A Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship

Duenpen Kochakornjarupong

B.Sc. (Computer Science)
M.Sc. (Computer Science)

UNIVERSITY
of
GLASGOW

A thesis submitted for the degree of Doctor of Philosophy (Ph.D.)

The SCRE Centre
University of Glasgow
July 2007

Abstract

The thesis addresses the issue of how to help staff in Universities learn to give feedback with the main focus on helping teaching assistants (TAs) learn to give feedback while marking programming assignments. The result is an innovative approach which has been implemented in a novel computer support system called McFeSPA. The design of McFeSPA is based on an extensive review of the research literature on feedback.

McFeSPA has been developed based on relevant work in educational psychology and Artificial Intelligence in EDucation (AIED) e.g. scaffolding the learner, ideas about andragogy, feedback patterns, research into the nature and quality of feedback and cognitive apprenticeship.

In particular, a number of issues in designing various patterns for forming the feedback have been investigated. McFeSPA draws on work on feedback patterns that have been proposed within the Pedagogical Patterns Project (PPP) to provide guidance on structuring the feedback report given to the student by the TA. The design also draws on the notion of andragogy to support the TA. McFeSPA is the first Intelligent Tutoring System (ITS) that supports adults learning to help students by giving quality feedback.

The approach taken is more than a synthesis of these key ideas: the scaffolding framework has been implemented both for the domain of programming and the feedback domain itself; the programming domain has been structured for training TAs to give better feedback and as a framework for the analysis of students' performance. The construction of feedback was validated by a small group of TAs. The TAs employed McFeSPA in a realistic situation that was supported by McFeSPA which uses scaffolding to support the TA and then fade.

The approach to helping TAs become better feedback givers, which is instantiated in McFeSPA, has been validated through an experimental study with a small group of TAs using a triangulation approach. We found that

our participants learned differently by using McFeSPA. Consistent with our hypotheses, the evaluation study indicates that 1) Providing content scaffolding (i.e. detailed feedback about the content using contingent hints) in McFeSPA can help almost all TAs increase their knowledge/ understanding of the issues of learning to give feedback, 2) Providing metacognitive scaffolding (i.e. each level of detailed feedback in contingent hint, this can also be general pop-up messages in using the system apart from feedback that encourage the participants to give good feedback) in McFeSPA helped all TAs reflect on/rethink their skills in giving feedback, and 3) When the TAs obtained knowledge about giving quality feedback, providing adaptable fading of TAs using McFeSPA allowed the TAs to learn alone without any support.

McFeSPA's implementation was intended to be sufficient to test the main hypothesis that TAs could benefit from an on-task computer system designed to support the TA to learn how to give feedback. The work with TAs suggests that a more complete implementation of the approach would be acceptable to TAs for real life use. Our analysis of the comments of the TAs provides the basis for further work so that a future version of McFeSPA could support all the needs of both novice and experience TAs.

The thesis makes an original contribution to the fields of AIED and ITSs - particularly the scaffolding approach that can help the TAs improve the quality of their feedback to students during marking Prolog programming assignments. The work also makes a potentially significant contribution to work on formative assessment using Information and Communication Technology (ICT), especially an innovative approach that provides opportunities for fostering reflective thinking by the feedback giver. Finally, this thesis contributes to the field of HCI by demonstrating an original approach based on an adaptive and adaptable interface in relation to helping the TAs use the system to provide quality feedback to students' programming assignments.

Acknowledgements

Only a page is hard to acknowledge many greate people that assisted and supported me to produce this work. I am taking this chance to indicate my deepest greatfulness to these people who have contributed towards the completion of this thesis. I attempt my best and apologise dearly for those whom I overpass.

I would like to articulate my sincere gratitude to my supervisor, Prof. Paul Brna. I am indebted to him who not only have given me intensive supervision throughout my study, but also all his advice for my various health problems during my study. Thanks for his time for critical reading, thanks for all his encouragement, valuable advice and all suggestions and thank you very much for his patient to my poor English. I also give thanks to Dr. Paul Vickers who was my co-supervisor in the initial year of this research.

I am also grateful to both Dr. Stuart Watt and Prof. Robert Matthew for serving as members of my committee and provided useful suggestions to improve my thesis.

Thanks to Pensri Amornsinlapachai who assisted and guided me in the initial prototype and thanks to all members in the former LTRG at University of Northumbria at Newcastle who suggested me to improve my initial prototype.

Thanks to the current and former SCRE stuff who assisted me commented to my materials for evaluation of my study, thanks to all evaluators and participants who provide helpful comments for improving of my future work, and thanks to Mr. James Slatcher who proofread my thesis for two rounds.

Thanks to all my Thai friends at University of Glasgow, University of Edinbugh, University of Northumbria, and my colleague in Thailand who have supported me and help me in any forms.

Thanks to Royal Thai government for financial support for nearly four years, and Mathematics department at Thaksin University for allowing me to be on leave to follow this PhD research.

Thanks to the formers School of Informatices, University of Northumbria at Newcastle for the facilities and staff provided towards this end.

I wish to thank granny (Khun Ya) Yaowarath Bunnak who taught etheric body for health and thanks to all her teachers (e.g. The Headmaster Dasira Narada, Ms. Barbara Ann Brennan, Prof. C.W. Leadbeater, and Dr. Robert Channy) for all the knowledges for helping other to enjoy good health.

These acknowledge would not be complete without special thanks to my mother who support me during Ken's birth and his growing during the early his year. I also give thanks to my younger brother for taking care of father, mother, and elder brother during my study, and for any form of assistance.

A special Thanks goes to my dearest best husband for helping in any assistance, especially looking after Ken deliberately.

Last but not least, supreme thanks to Buddha who is the best guide and the greatest teacher who points the way to nirvana (final release from the round of rebirth) to all people in the world. Especially, doing meditation help me to concentrate on my work, help release my tension while doing my work, and also help me much be patient to overcome any obstruction until today.

This thesis is dedicated to my both dearest families (Kochakornjarupong and Santhititham), my relations, and my foster mother.

Thank you all,

Duenpen Santhititham Kochakornjarupong

Declaration of Author's Copyright

I hereby declare that this thesis was composed by my own that the all work described therein has been done by myself and this work has not been submitted for any other degree or professional qualification except as specified.

Duenpen Kochakornjarupong
July 2007

Table of Contents

Table of Contents

PUBLICATIONS

1. Kochakornjarupong, D., and Brna, P. (2003). Towards Scaffolding Feedback Construction: Improving Learning for Student and Marker. In Y. S. Chee, Law, N., Lee, K., and Suthers, D. (Eds.), Proceedings of the 11th International Conference on Computers in Education, (ICCE 2003) (pp. 599-600). 2-5 December 2003, Hong Kong.
2. Kochakornjarupong, D. (2003). Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship. In Kinshuk (Ed.), Proceedings of the Doctoral Student Consortium of the 11th International Conference on Computers in Education, ICCE 2003 (pp. 5-8). 1 December 2003, Hong Kong.
3. Kochakornjarupong, D., Brna, P., & Vickers, P. (2005). *Who Helps the Helper? A situated Scaffolding System for Supporting Less Experienced Feedback Givers.* In Proceedings of the 12th International Conference on Artificial Intelligence in Education, (AIED 2005) (pp. 851-853), Amsterdam, The Netherlands, 18-22 July 2005.
4. Kochakornjarupong, D., &Brna, P. (2007). *How to Assess the Feedback Giver? Evaluating of a Scaffolding System for Supporting Less Experienced Feedback Givers.* In Proceedings of the 13th International Conference on Artificial Intelligence in Education, (AIED 2007), Marina Del Rey, California, USA. 9-13 July 2007.

List of Figures

List of Tables

Chapter 1

Introduction to the Research

1.1 Introduction

The aim of this research is to make a contribution to existing research on how to improve the quality[1] of the feedback[2] given to learners and at the same time address the problem of helping the feedback-giver to improve their feedback giving skills. This research addresses both facets in an effort to explore the role that theories of scaffolding and competence play in the design of a system for training teaching assistants (TAs) to analyse the feedback they give to their students. The approach taken in the thesis is that TAs should be supported and become more effective feedback giver. This thesis is consistent with the QAA Enhancement Theme[3] ("QAA Scotland," 2007) on assessment. It aims at enhancing the understanding of the TA's own performance. In addition, this research is complementary to the research themes suggested by Matthew (2004) i.e. 'Improving feedback to students (link between formative and summative assessment)' which is an assessment sub-theme of Enhancement Theme ("Enhancement Themes," 2007). Matthew (2004) reported that there were still many issues about 'Improving feedback to students' unsolved. Some of these are: 'Training of students for self and peer assessment'; 'Training of students to use feedback effectively'; 'Development of student abilities to recognise good and bad work'; 'Staff time to develop new feedback methods'; and 'Getting the

[1] Quality is how good or bad something is. In this thesis, we research what it means for something to be quality feedback drawing on the extensive literature about feedback. For the purposes of this thesis, 'good feedback' is synonymous with 'quality feedback'.

[2] The return of information about the result of a process or activity with the aim that the learner improves their understanding. This thesis is primary concerned with managing the feedback from teacher to students."

[3] The Enhancement Themes initiative is part of the Quality Enhancement Framework which is designed to support higher education institutions in Scotland to manage and enhance the quality of the student learning experience and to increase public confidence in the quality and standards of higher education. QAA is the Quality Assurance Agency for Higher Education; Quality Assurance (QA) means trying to meet the standard of higher education while Quality Enhancement (QE) means trying to improve practice rather than attain some minimum standard.

balance between formative and summative assessment'. So there are a number of issues related to improving feedback to students. Therefore, research in this area has extensive possibilities for improving teaching practice. In order to perform this research, firstly, we need to explain the nature of the problem (see Section 1.2) and after that we will explain the overall objectives of the research and plan (see Section 1.3). Thereafter, we will draw out the anticipated contribution of knowledge (see Section 1.4) and finally we will propose a synopsis for the thesis (see Section 1.5).

1.2 Problem Statement

Many educators agree that teaching and learning is more effective when reinforced by appropriate instructional material in the form of either feedback or suggestions to the learner (Brown & Knight, 1994; Askew & Lodge, 2000; Gibbs, 2006) in which assessment is also part of the instruction. In addition, assessing learner achievement, which not only involves providing a score, but also more detailed feedback in order to improve students' learning (Brown & Knight, 1994; Askew & Lodge, 2000), is a vital element in teaching and learning. Giving feedback, in fact, is not only useful for the learners, but it can also develop teachers' ability to assess students work (Race, 2001). From assessment, especially, learners desire more feedback (Gibbs & Simpson, 2003), particularly in adult learners who are more inclined to under take self-directed learning (Knowles, 1988), to develop their studies. In other words, some learners may be unhappy that they work on their assignment with considerable effort but obtain inadequate feedback (London, 1995). As a consequence, feedback, indeed, is a primary instructional strategy and also a powerful component in learning (Chai, 2003). However, there have been some problems in giving quality feedback.

Kulhavy and Wager (1993) argued that the result of studies conducted since the 1960s suggests the general finding that some feedback is better than no feedback. Furthermore, despite the fact that characteristics of quality feedback should be more detailed and related to individual learners, the result of giving feedback has not helped students much because quantity and quality of feedback

may not be enough to help students (Gibbs & Simpson, 2003). Beyond this, there are the following additional problems:

1.2.1 Class size

Even though a number of students enroll in higher education in order to obtain a higher degree, the problem of assessing a large number of assignments (Gibbs & Simpson, 2003) results in insufficient time for the instructors[4] who teach in higher education to give quality feedback. It is, furthermore, not easy to provide 'detailed feedback' to all learners in a large class (Brown & Knight, 1994). Providing feedback is, of course, a crucial part of learning and teaching; yet in a large class, it is difficult to give 'individual feedback' to every student according to Gibbs & Simpson.

1.2.2 Quantity of feedback

Most students require feedback that not only does not provide too many comments, but also gives more guidance with regards to their future improvement (Brown & Knight, 1994). Most tutoring systems provide feedback to the users (e.g. Ecolab (Luckin & du Boulay, 1999; Luckin et al., 2003), Cognitive Tutors (Anderson et al., 1995), and PACT Geometry Tutor (Aleven & Koedinger, 2000; Aleven & Koedinger, 2001), etc). With regard to the tutoring system if it provides too much feedback to users, they may not think by themselves and wait for external feedback to influence their thinking. In comparison, if the tutoring system does not provide enough feedback, users may be disappointed that they cannot see the correct answer (Smith, 1997). This can be poor feedback in which it is not suitable for the tutors to give such feedback to the students. Besides, condition 9 of Gibbs & Simpson (2003), concludes that students do not understand more detailed feedback from the teachers; they do not understand what the teachers attempt to explain to them; they receive feedback which is hard to interpret. There may be complicated feedback from teachers that do not properly understand their students preferred learning methods.

[4] An instructor is a university teacher ranking below assistant professor (Oxford Dictionary, Oxford University Press, 2001). In this thesis, we use "teacher" stand for "instructor".

Furthermore reflective practice assignments (Brown et al., 1997) of various types of assessment can result in time-consuming feedback marking, which can result in low quality feedback. In addition, providing each student with greater content in their assignment feedback affects teachers' workload in higher education.

1.2.3 Individual Differences

In practice, giving individual feedback is perhaps often inconsistent due to the differences in students' abilities and the amount of effort they put in to their study (Beck et al., 2002). In addition, if teachers give inconsistent feedback, this may confuse students.

1.2.4 Timing of feedback

In the case of giving delayed feedback, it may be detrimental if the learners have inadequate time to assimilate the previous lesson before going on to the next lesson (Race, 2001). However, provision of immediate feedback to students can cause confusion to the students when they obtain several errors and can not remember everything.

1.2.5 Content of feedback following student error

In fact, although many different approaches have been tried through trial and error feedback for learning purposes has not succeeded (Race, 2001). Traditionally, when giving feedback, most teachers only give feedback when students make mistakes, but they do not give feedback for correct answers (Race, 2001). Despite its importance, methods for providing feedback have not been very successful. Most methods have been established by a system of trial and error. In addition, Graesser, Person, & Magliano (1995) found from their studies that tutors do not devote much time to explaining errors. Due to the fact that it is difficult for the tutors to deal with the problems of distinguishing errors and misconceptions, the students are left to work out the answers themselves.

1.2.6 Positive feedback

One problem with giving feedback, particularly, with large class sizes, is that perhaps the teachers ignore the importance of feedback statements. Failure to understand the value of positive feedback may lead to students lacking motivation to follow the teacher's comments. On the other hand, this is especially so when teachers provide only positive feedback which does not include enough reasonable detailed feedback, for example, when a teacher gives "excellent" at first time of marking and next time he/she gives "very good" (Race, 2001).

1.2.7 Problems in giving quality feedback

Apart from the various problems of giving quality feedback outlined above, in condition 11 from the investigation by Gibbs & Simpson (2003), there are many reasons why poor feedback affects students' learning in the future, for example, returning feedback late to students; feedback may be unrealistic or unspecific; students do not known the answer from the feedback question or what to do from the feedback; feedback may discourage students; students do not follow feedback or know how to improve according to the feedback. When providing feedback, there have been several common errors:

- Not allowing the speaker or presenter the opportunity to comment on his or her work.
- Saying what you would do rather than listening to what the person says he or she did.
- Saying what was done is totally useless.
- Attacking the person rather than analysing the question (Brown et al., 1997).

Problems with providing feedback to students might be either oral or written. In this manner, it is interesting to take feedback on assignments into account particularly in large classes.

1.2.8 Problems in giving feedback on assignment

Nowadays, for almost all people teaching and learning at undergraduate level, lecturers[5] teach large groups of learners. To provide quality teaching, all lecturers should pay attention to, and follow closely, the learning development of each learner. In particular, lecturers should provide individual feedback or instruction or consult each student. Despite this, the problem of big classes still leads to inadequate time to assess students' learning (Tsintsifas, 2002) together with providing quality feedback i.e. marking and providing such feedback for each learner.

Teaching big classes affects assessment, which is especially true at the undergraduate level Universities often hire helpers as teaching assistants (TAs) or employ automated marking assignments. However, novice teachers or TAs lack the required training in giving quality feedback (Dennis et al., 2002). Although learning instruction is improved effectively, some research shows inconsistent results. This is because many teachers have not been trained to give feedback. However, there is very little research in this area (Kochakornjarupong et al., 2005).

1.2.9 Inadequate training in giving quality feedback

With regard to the marking of assignments and providing feedback that is necessary for teaching and learning at undergraduate level, many people who are requested to give feedback have inadequate training to do so. Providing feedback is an important task, yet there are still some teachers who argue that they have not been trained to improve their students' learning through feedback (Brown et al., 1997). Even if automated marking assignments systems or semi-automated marking assignments system such as SPROUT (Pardoe & Vickers, 1994; Rimmer et al., 1995), Ceilidh (Foxley et al., 1999), CourseMaster (Foxley et al., 2001; Higgins et al., 2002), and BOSS (Joy & Luck, 1998; Joy et al., 2000) can provide automated feedback to students, they have not taught tutors or TAs. Tutors or TAs might have a great deal of experience in programming language issues but they

[5] A lecturer is a person who gives lectures, especially as a teacher in higher education (Oxford Dictionary, Oxford University Press, 2001). In this thesis, we use "teacher" stand for "lecturer".

lack knowledge in giving quality feedback. This knowledge is a part of the pedagogical strategies in which they give feedback like the feedback that they received when students, according to epistemological beliefs (Hoffer & Pintrich, 1997; Hammer & Elby, 2000). That is the knowledge that the TAs have obtained from their teachers, to learn how to give quality feedback in order to provide more detail of explanation of feedback provision.

In fact, giving feedback is a fundamental part of learning and teaching. Thus, learning to give quality feedback should be a concern. In the next section we will propose the outline approach.

1.2.10 Outline Approach

Although the main issue includes both students and TAs, this research is primarily about supporting TAs. This involves both helping the TAs to be efficient and helping them to learn about how to provide quality feedback. Due to the fact that the scaffolding approach has been used successfully to support learners, we have selected this as an appropriate approach to teach TAs who like most adults have little time to learn anything- while engaged in marking students' scripts.

The scaffolding approach involves helping the learners to succeed in a way that they could not accomplish on their own (e.g. the Zone of Proximal Development (ZPD) (Vygotsky, 1978)). Scaffolding techniques have been applied effectively in Ecolab (Luckin & du Boulay, 1999; Luckin et al., 2003), Cognitive Tutors (Anderson et al., 1995), and PACT Geometry Tutor (Aleven & Koedinger, 2000; Aleven & Koedinger, 2001), etc). As a consequence, it is likely to be useful for helping the TAs to learn to give quality feedback. When the TAs achieve mastery, they will not require any support from the system. The system 'fades' away by reducing the amount of support which the TAs require to improve their learning. Not only do our approaches lie in the scaffolding approach, but they also employ cognitive apprenticeship (Collins et al., 1989); feedback pattern (see Principles of McFeSPA in Chapter 5); providing quality feedback (see Chapter 3); contingent help approach (Wood, 1999); and andragogical model (Knowles et al., 1998).

However, there is little research on the scaffolding approach applied in computer-support to help novice teachers learn to give quality feedback e.g. semi-automatic/automatic marking systems rarely provide any support to train the novice teachers to give quality feedback. We have been unable to detect a system in our research that either supports or trains the novice feedback giver. In addition, a number of systems scaffold students to learn in a particular context but they have not helped adults learn in giving feedback to improve their feedback giving. To do this, we have chosen to work on the problem faced by TAs in the real situation of marking programming assignments in large classes to explain students' errors. In this manner, TAs are likely to be inexperienced in communication skills even though they have programming skills.

1.2.11 Summary

In conclusion, it is, of course, perhaps a truth per se that feedback is essential for learning, yet several problems, as mentioned above and are also associated with the investigation of Doig (1999), entail provision of quality feedback. Therefore, the aim of this research is to improve the quality of the feedback given to learners and at the same time address the problem of helping the feedback giver to improve. In order to meet this aim, we desire to ask some further questions, these are: How does computer-support help TAs learn to provide quality feedback? How does computer-support help the TAs to improve their feedback giving skill? Can a computer-support system promote better help seeking activities providing feedback for TAs? i.e. TAs can use the system alone without any support. According to the effectiveness of scaffolding software as mentioned above, we have chosen to use a scaffolding framework (see Chapter 5) to improve TAs skill of giving feedback. Therefore, in order to help the feedback giver learn to provide quality feedback, the scaffolding framework could help them learn feedback giving skills in appropriate situations.

1.3 Overall objectives, and plan

The aim of this research is to develop a scaffolding system to help novice teachers or teaching assistants (TAs) improve the quality of feedback to students on their

programming language assignments and to improve the teacher's own understanding of how to give quality feedback. The plan and time table of the research can be seen in Appendix J. In the following we propose the overall objective and outline the limitation of the research.

1.3.1 Objectives and Requirements of the Research

In this thesis, we will design and implement a system which features with interesting properties in which we propose a set of tools that will provide an innovative method of improving the quality of feedback and of helping the teacher to learn how to give better feedback. We then test whether it is really interesting empirically. The prototype design, it involves the design and implementation of a semi-automated marking system (see Chapter 5-7) to extract error types (e.g. design, implementation, style, see Chapter 4) sorted by critical errors/weaknesses and common errors, it will annotate these errors and then utilise feedback patterns (see Chapter 3). Further, we provide scaffolding support to help TAs/new teachers choose which of the several forms of feedback pattern is better to use to generate the feedback for the students. For the evaluation, we use the system with scaffolding turned off to help three TAs give feedback by using pre-provided mock-up student scripts and three TAs using the full system with the same mock students' scripts. Consequently, we test the hypothesis that the TAs who use our system could give better feedback to the students' scripts as judged by a range of measures, for more details see Table 1.1.

Table 1.1 Research gaps, Hypotheses, Methodology

Research Gaps	Hypotheses	Conceptual Approaches	Methodology	
			Design & Implementation	Evaluation
1. There is little research in training novice teachers to give feedback.	1.1 TA can learn to give quality feedback by giving feedback to students.	-	-	1.1 Reviewing and examining literature (see Chapter 2)
	1.2 Modeling of quality feedback from a mentor could train the TA to give quality feedback.	-	-	1.2 same as the methodology of 1.1
	1.3 Suggestions from teachers could train the TA to give quality feedback.	-	-	1.3 same as the methodology of 1.1
	1.4 The TA could learn to give feedback from student comments.	-	-	1.4 same as the methodology of 1.1
	1.5 The TA could learn to give feedback from teacher comments.	-	-	1.5 same as the methodology of 1.1
2. There is little research in the design of a framework for analysis of student performance to be	2.1 Design issue could be types of student programming weaknesses.	-	-	2.1 Reviewing and examining the literature (see Chapter 4)

Research Gaps	Hypotheses	Conceptual Approaches	Methodology	
			Design & Implementation	Evaluation
a domain in training the TA to give quality feedback to the students.	2.2 Implementation issue could be types of student programming weaknesses.	-	-	2.2 same as the methodology of 2.1
	2.3 Style issue could be types of student programming weaknesses.	-	-	2.3 same as the methodology of 2.1
				Interview expert lecturers in marking programming assignments (see Appendix A)

Research Gaps	Hypotheses	Conceptual Approaches	Methodology	
			Design & Implementation	Evaluation
3. There is little research in employing computer-support to help novice teachers to provide quality feedback.	3.1 Computer-support could **help the TA deliver feedback to the learner as an authoring tool** (type of scaffolding: **Functional scaffold** help the user to interpret/ use software)	3.1 Using Carroll's SBD (Scenario Based Design) (Carroll, 1995, 2000) and HCI design (Benyon & Imaz, 1999) to design and implement interface e.g. choice of feedback message (weakness message or positive feedback), managing feedback message by editing/adding/deleting, and organising feedback; editing and generating feedback report; and user customisation. We propose a sub-testing of usability evaluation by using HCI evaluation methods, according to the ISO 9241 standard, part 12, that defines usability in terms of effectiveness, efficiency, and satisfaction (Kirakowski, 2000) i.e. User satisfaction questionnaire	3.1 Interview with an expert in giving feedback 3.2 Interview with an expert in teaching programming 3.3 Interview with a lecturer in teaching Prolog programming 3.4 Design Scenario of the environment for electronic marking and providing quality feedback 3.5 Design and Implementation of the initial system according to the scenario based design in 3.4 3.6 Redesign and Reimplementation of the initial system to develop the pilot system	3.1 To test **Usability** in initial system by testing the system with specialists in usability by collecting data from semi- structured interview (Robson, 2002), questionnaire, and observation (for redesigning the initial system to be the pilot system) 3.2 same as 3.1 and testing with an expert in giving feedback 3.3 Empirical work (after redesign/reimplementation) 3.4 Analysis of the results

Research Gaps	Hypotheses	Conceptual Approaches	Methodology	
			Design & Implementation	Evaluation
4. There is little research from employing computer-support to help novice teachers improve quality of feedback giving skills.[i]	4.1 Implementation of scaffolding approach in computer-support could **help the TA increase knowledge/ understanding in issues in learning to give feedback.** (type of scaffolding: **Content scaffold** e.g. help/hint)	4.1 Theory of Help Seeking and Help Design (Aleven et al., 2003), Scaffolding, and Contingent Tutoring (Wood, 1999; Wood et al., 1999; Wood, 2001; Aleven et al., 2003), andragogical model (Knowles, 1988, 1990), framework of Learning Environment (Collins et al., 1989), instructional design framework (Herrington & Oliver, 2000) to design and implement rule of feedback pattern, rule of hint, rule of dialogue response.	4.1 Design Scenario of the environment for scaffolding/help the TA to learn/train to provide quality feedback. 4.2 Design and implement rule for feedback pattern, rule of tutor's hint, rule of quality feedback, and rule of dialogue response. 4.3 Redesign and Reimplementation of the initial system to develop the pilot system	4.1 To test TA **Learning** to give quality feedback by testing the system with specialists by collecting data from structured interview (Robson, 2002), questionnaire, and observation (for redesigning the initial system to be the pilot system) 4.2 Empirical work (after redesign/reimplementation, the data will be collected by observation of the TAs behavior *; pre- & post-test; structured interview; and questionnaire 4.3 Analysis of the results

Research Gaps	Hypotheses	Conceptual Approaches	Methodology	
			Design & Implementation	Evaluation
	4.2 Implementation of scaffolding approach in computer-support could **help the TA reflect/rethink his/her skills in giving feedback.** (type of scaffolding: **Metacognitive scaffold)**	4.2 same as the methodology of 4.1 & Skill Meter (Corbett & Trask, 2000)	4.4 same as methodology 4.1 - 4.3	4.4 To test TA **Learning** to give quality feedback by observation the TA behaviour via the log data goal (video-recorded –avi file); structured interview; questionnaire; and analyse the result.
	4.3 When the TA obtains knowledge of giving quality feedback, implementation of **fading computer-support could allow the TA to learn alone without any further support** - known as **adaptable fading**.	4.3 same as the methodology of 4.1 & adaptable fading (Jackson et al., 1998)	4.5 same as methodology 4.1 - 4.3	4.5 same as the evaluation of 4.2-4.3 * (of 4.2) e.g. counting events such as errors; assessing the quality of the output that the TA generates with the aid of the system we are evaluating

In other words, our methodology consists of: design and implementation of a system to perform some automated feedback (e.g. using Visual Basic, Prolog) and analysis of student solution (e.g. Prolog); design and implement a system for marking assignments (e.g. using Visual Basic); design and implement a system for advising teachers (a) Active -like dialog to remind the teacher, b) Passive (e.g. offer choices for processing feedback message and generating feedback report)); Integrate all above.

1.3.2 Limitations of the Research

Our system does not involve how students interact with the TAs/new teachers. Furthermore it is out of the context of our system to consider student questions related to the feedback which they receive. Beyond employing our system, other approaches are generally, to use FAQ (Frequently Asked Question) or an Answer Garden which is dynamic system that popular via E-mail; however, our system does not involve online feedback. In addition, our system does not design standard questions for both programming domain and feedback domain, not specific face-to-face between TAs and the system but emphasises high quality feedback. In terms of providing support, it is not easy to guarantee that we have a complete set of techniques to help provide the best quality feedback because the kinds of support that we can provide must be capable of helping the learner and the feedback giver to improve their performance on current and later tasks. Besides even though our system does not give any score, it provides an approach to a semi-automated assessment helping markers to learn to give feedback i.e. helping TAs by scaffolding, contingent help, help seeking (temporary help). Therefore we do not focus much on the metric based automated assessment because our system is another dimension of formative assessment that focuses on the students' type of error/weaknesses.

1.4 Anticipated contribution of knowledge.

This chapter contributes to the issue regarding how to provide help to the learners to improve their learning from their feedback. This work contributes to the field of Artificial Intelligence in Education (AIED) which focuses on how to train

people to give quality feedback in the situation of marking assignments applied to programming teaching. We propose the contribution according to the following.

1.4.1 Major Contribution

A new scaffolding system which can help novice teachers or teaching assistants (TAs) improve the quality of feedback to students in the situation of marking programming assignments and to improve the teacher's own understanding of how to give quality feedback i.e. it will contribute to TAs directly, and students indirectly.

1.4.2 Minor Contribution

A framework for scaffolding systems that will help people learn something from the system and that can be applied to contexts other than training teachers to give quality feedback e.g. design of McFeSPA could help people in online assessment, supervision assessment, and student assessment. In addition, the approach can be applied in training supervisors to give feedback to employees.

1.4.3 Summary

We believe that using a scaffolding system to help TAs learn to give feedback can help them gain experience, and improve their skill and approach to giving quality feedback; however, our system is an advance in helping TAs to do better. This should be particularly helpful for TAs who have not learnt to give feedback.

1.5 Synopsis of the thesis

This thesis consists of nine chapters. Chapter 1 proposes the general problems to be addressed and the objective of the scaffolding system through the anticipated contribution of knowledge. This chapter highlights the main problems in giving feedback. Chapter 2 presents the nature of giving feedback and the characteristics of good feedback. This chapter highlights several issues in relation to the following; how do people learn to give good feedback and how do computer systems help teachers provide feedback? Chapter 3 presents the feedback design

which relates to several factors to underpin the scaffolding system for help giving quality feedback. This chapter highlights the analysis of the level of feedback content for constructing the model of giving feedback from the system to TAs directly and to students indirectly. Chapter 4 proposes a programming domain for training TAs to give quality feedback as a framework for analysis of student performance. This chapter highlights a framework for classification of types of weaknesses. Chapter 5 presents the investigation of several scaffolding systems through system design based on several interesting approaches. This chapter highlights the system's architecture and principles for training TAs in giving better feedback. Chapter 6 explores scenario-based design of scaffolding systems in order to develop interface. This chapter highlights the design situation of a TA marking an assignment with a semi-automated marking system. Chapter 7 presents the implementation of the system and reports the usability evaluation. This chapter highlights the evaluation results with discussion of the results to improve the system for the evaluation in the next chapter and further work. Chapter 8 reports the evaluation of the system's learning environment. This chapter highlights the results of the evaluation and discussion of analysis of the results regarding the next version of the system. Chapter 9 concludes all research achievements and proposes the contributions of the research. This chapter highlights the theory and implications for future research work.

How do people learn to give good feedback?

2.1 Introduction

In order to answer the questions in Chapter 1, this chapter we will investigate the range of research that investigates how to help educators learn or practice giving feedback in order to improve their skills for giving better/quality feedback to students in the future. Providing teachers with practice to upgrade their skills is like teaching teachers to improve their feedback giving skills. In general, considering providing feedback that is crucial for teaching and learning at undergraduate level, many people have little training to give the required feedback (Kochakornjarupong et al., 2005). Most research involves training teachers in learning, teaching and assessment (Nicol, 2000) but there is no feedback training. Providing feedback is an important task. In addition, several educators have researched how to give good/quality feedback to students in higher education with regard to innovative assessment (Bryan & Clegg, 2006), yet there have been some teachers who argue that they have not been, yet there have been some teachers who argue that they have not been trained to improve their students' learning through feedback (Brown et al., 1997). How can an inexperienced teacher become a good teacher? From observation of other teachers, and reflection on others teaching methodology. For these reasons, we should know which is good feedback to provide learners with and which feedback is poor and should be avoided. Further, from the available literature, it seems possible to suggest the following ways of helping to provide feedback in accordance with the following.

2.2 What counts as good feedback?

Kulhavy & Wager (1993) argued that the result of studies conducted between the 1960's and the early 1990's in general suggest that some feedback is better than

no feedback. According to Brown et al. (1997), the goal of feedback, in fact, is to assist someone to develop what he or she is doing in which it should be beneficial and sufficient to the recipient. They also suggested that feedback should be specific, accurate, timely, clear and necessary to encourage a person in order to change their thinking and then improve their experience.

Bangert-Drowns & Kozma (1989) proposed effective feedback in which it should include giving correct answers together with positive feedback, give correct response, give an assessment of student answers, give the reason for students' incorrect answer, support student to achieve correct answer, advise special approaches to students.

In general, providing feedback can help students know how to improve their next piece of work (Brown & Knight, 1994; Askew & Lodge, 2000). Clarke (2000) reported the example of giving effective feedback e.g. "You worked very hard on this". In addition, giving feedback not only is more useful for the learners, but it can also develop teachers' assessment techniques (Race, 2001). There are several kinds of effective feedback which we will explore in the nature of feedback (see Section 2.2.1) then we shall review the characteristics of good feedback (see Section 2.2.2) followed by techniques of giving good feedback (see Section 2.2.3) and then by Effective assessment with good feedback (see Section 2.2.4) and finally we will discuss issues related to the marking assistant and good feedback (see Section 2.2.5).

2.2.1 Nature of feedback

The term "Nature of feedback", or reinforcement, refers to the foundation of a number of instructional principles used in the context of programmed instruction (Deterline, 1962; Markle, 1964). These principles included "use prompting". i.e. providing hints is recommended in order to "shape" the learner's behaviour by selectively reinforcing the correct response. The other principles concerned the choice of an appropriate "step size" i.e. how much information to present at once, and how often feedback or reinforcement should be provided. Generally, "nature of feedback" references issues connected with quality, quantity, and language used.

The topology of Tunstall & Gipps (1996) suggested that the content of feedback should include evaluative or descriptive strategies: *evaluative feedback strategies are giving rewards and punishments, and expressing approval and disapproval e.g."well done", "good boy", "Brilliant"; descriptive feedback strategies are telling children they are right or wrong, describing why an answer is correct, telling children what they have and have not achieved, specifying or implying a better way of doing something, and getting children to suggest how they can improve such as the "go back and check your work" command, the "tell me how" invitation, the "what would make this better" question.* Even though the strategies of Tunstall & Gipps (1996) are for the primary classroom, they might still be useful to adapt for giving feedback at the undergraduate level. Many teachers agree that it is good to provide evaluative feedback and follow this with descriptive feedback to describe why an answer was right or wrong or how it might be improved (Hargreaves et al., 2000).

Basically, the kinds of feedback covered by the term "nature of feedback" (e.g. using prompting) are not of an adequate quality because of their lack of detail. We want an answer to how we provide quality feedback – so studying the characteristics of good feedback in what follows could help us select the types of quality feedback to use in our research.

2.2.2 Characteristics of good feedback

In order to provide quality feedback, there are several factors to take into account as follows.

2.2.2.1 Asking key questions

In order to provide constructive or focused feedback and encouragement, there should be key questions to ask in the tutorial. These are *(a) What are you trying to do? (b) What have you been doing? (c) What problems are you having? (d) What are you going to do next?* (Brown et al., 1997). Similarly, considering giving feedback on assignments, the teacher should ask some questions to the learners when the work is returned. Brown and colleagues (p.4) revealed that there are three valuable questions to ask when providing feedback. Those are

- What were you trying to do?

- How did you do it?
- Why did you do it in that way?

Furthermore, from the reviews of Chi, Siler, Jeong, Yamauchi, & Hausmann (2001) in relation to effective feedback, teachers may give students prompt explanation via feedback e.g. *"Can you explain this in your own words?" "Explain why you believe that your answer is correct or wrong?" "What did we learn in class about this particular topic?"* Beyond this, it should be vital to give feedback information to students to tell them how to focus on further steps (e.g. providing hints to students to complete the tasks) in which it is concerned with the process of learning (Suhonen et al., 2001). These may remind the learner to focus on their work. Nevertheless, providing 'asking key questions' to students might be inadequate to either improve or revise their assignments. Thus, providing feedback by asking key questions alongside providing hints that depends on the types of problem in order to guide the students to update their assignments, could help them improve their learning development.

2.2.2.2 Specification of the important error

Noonan (1984) found that the process of giving feedback depended on correct answers rather than the type of error that learners performed. Thereafter, Kula & Wager (1993) reported that there is no research supporting the concept that feedback following errors causes students distress. In that case, it is obvious that not only feedback can help the learner to correct errors but can also reinforce motivation. For this reason, a large number of studies tried to combine motivation with feedback (e.g. from the investigation of Methaneethorn, Vickers, and Brna (2004)). Giving a feedback response as only correct/incorrect (Knowledge Of Response: KOR) is less useful to learners than providing feedback with the correct answer (Knowledge of Correct Response: KCR), Kulhavy & Wager also found that adding further information to feedback messages has no consistent effect on instructional performance. This is because feedback may depend on "learner's knowledge" and "time of responding", according to Kulhavy & Wager.

Brown & Knight (1994) discovered that students require feedback that does not provide too many comments and also gives more guidance to be useful for their improvement. Nevertheless, it is necessary to provide feedback

especially on important or reoccurring errors to save time in debugging errors and teachers should not give too much feedback because students have to put aside more time to be able to understand them (Race, 2001). It is clear therefore that not only do the teachers waste time giving feedback on every error but also the learners may not pay attention to all those comments. Beneath this idea of focusing on important errors, Gibbs (1992) attempted to provide effective and efficient feedback by providing more relevant and interesting feedback to the learners. Considering important errors, in findings from the marking system for CourseMaster (Higgins et al., 2002), most students appreciate a system that gives important errors only. According to their experience, giving feedback on every error could be detrimental to the students learning system. Therefore, the quantity of feedback should be controlled by the teachers in accordance with the students' ability. Furthermore, in order to provide feedback following errors, Intelligent Tutoring Systems are designed to help the learner solve bugs and misconceptions that normal tutors may not support because skilled tutors seldom give knowledgeable feedback (Putnam, 1987; Lepper et al., 1990; McArthur et al., 1990).

2.2.2.3 Thinking students' action

In general, teachers may consider students' action before providing feedback. Laurillard (2002) categorized feedback into intrinsic and extrinsic feedback, extrinsic feedback, for example, *'very good', 'should try harder'* etc. Effective quality feedback of this kind is imitation of intrinsic feedback, for example, *'you have offered good evidence for your arguments here?' 'You would have achieved a better introduction to this essay by including some historical background to the field'* etc (p.126-7). In order to attain the goal, comments of this kind of feedback should relate to the students' action. If providing feedback on students' essay writing is extrinsic feedback, tutors could supply more valuable content as intrinsic feedback. Therefore, providing intrinsic feedback is telling the learners what they should do in which it is individualised, private, formative feedback that help the learners to construct their understanding between theory and practice. In addition, Nicol & Macfarlane-Dick (2006) proposed seven principles of good

feedback practice, based on 'Seven principles of good practice in undergraduate education' (Chickering & Gamson, 1991)

1. help clarify what good performance is (goal, criteria, expected standards);
2. facilitates the development of self-assessment (reflection) in learning
3. delivers high quality information to students about their learning;
4. encourages teacher and peer dialogue around learning;
5. encourages positive motivational beliefs and self-esteem;
6. provides opportunities to close the gap between current and desired performance;
7. provides information to teachers that can be used to help shape teaching (p. 205).

They maintain that if formative assessment and feedback are used with regard to their seven principles of good feedback practice, then formative assessment and feedback might be the basis for further work on supporting students' development of their own learning (self-regulation of learning). Beyond this, it would be constructive to tell the learner to write on the bottom of their assignment about which feedback they prefer to receive from the tutors before handing into the tutors (Gibbs, 1992). However, it might be difficult for computer-support to distinguish the learner preference unless the system provides a choice for them to choose then next time the tutors can provide appropriate feedback which is relevant to the learners' requirements.

Furthermore, giving good feedback should reflect the students' action. Hatton and Smith (quoted in Brown et al., 1997), suggested that reflection might be defined as "deliberate thinking about an action with a view to its improvement". They discriminated four kinds of reflection that are obvious in the essays of students depending on their writing skill. These are

- Descriptive writing in which no reflection is evident
- Descriptive reflection in which some reasons, based on personal judgement, are provided
- Dialogic reflection in which a student explores possible reasons and approaches which may be rooted in their reading of the relevant literature
- Critical reflection that involves exploring reasons and approaches and the underlying assumption and concepts. (p.30).

With regard to learning to give feedback, a computer-support system can be a tool to help teachers reflect on their actions while giving feedback to students. That is the system can interact with the novice teachers or TAs by providing/reporting their progress of feedback giving. This is to help them reflect on their performance on giving feedback.

2.2.2.4 Individual feedback

Ersoy (2001) pointed out that informing a learner about his or her performance i.e. giving feedback, can increase motivation. In the same way, feedback is most effective during learning and can improve both theoretical development and application. Giving individual feedback can reduce the gap between teachers and students in which they could receive effective, high quantity feedback, thinking about the learners' feeling when they receive their assignments back (Race, 2001). Hence, giving feedback to individual learners may be useful for each learner to improve their learning. Furthermore, it is vital that teachers should be careful to give effective feedback to learners. Denton (2001) suggested that good feedback should include the learners' name and avoid writing the same comment on students' work.

Besides, in teaching aspects in higher education in the UK ancestrally (e.g. Oxford University and Cambridge University) teachers often give individual detailed feedback (e.g. face-to-face tutorial, immediate or oral detailed feedback) on assignments. Even if this is a provision of quality feedback, Gibbs & Simpson (2003) hypothesized that it is central to student learning to provide frequent assignments with written detailed feedback. Nevertheless, it is difficult to give such feedback to all students in large classes.

2.2.2.5 Elaborative or detailed feedback

Graesser et al. (1995) found that tutors did not devote much time to explaining errors, due to the fact that it is difficult for tutors to deal with the problem of feedback to distinguish bugs and misconceptions, and so let the students try to overcome the problems alone. For this reason, Intelligence Tutoring Systems are designed to help the learner to solve bugs and misconceptions that normal tutors

may not support because skilled tutors seldom give knowledgeable feedback (Putnam, 1987; Lepper et al., 1990; McArthur et al., 1990)

According to Moreale et al (2002), Whitelock et al. (2003) and Denton (2003) having a helper to assist teachers or instructors –teachers in the undergraduate level- give more feedback, such as elaborative or detailed feedback, or quality feedback, can help the students develop their learning. In the study of Fernandes & Kumar (2004), they examined the effect of providing minimum feedback –do not explain why an answer is wrong, and detailed feedback – do explain why the answer is wrong. They claimed that textbooks did not provide clear explanations like minimum feedback in which it does not tell the learner why they answered incorrectly. They also found that the result of providing minimum feedback may damage the student learning process; on the other hand, detailed feedback may help students improve their learning, i.e. help them to learn better. One student made the following observation regarding written feedback, "I like the feedback because it helps me understand what I did wrong. Lots of examples help me understand things easier, and this helped show me what I was doing wrong." (Fernandes & Kumar).

2.2.2.6 Feedback including hint

In terms of marking for feedback, it may include hints, for example, choosing few positions in which the learners may be weak in order to tell them how to improve their skills rather than promote all errors (Brown et al., 1997).

2.2.2.7 Feedback loop

Beneath the provision of a feedback loop, if the learners who obtain their feedback do not improve their practice on the next assignment, the tutors should do the following feedback loop;

- Collect all comments or feedback for the next assignment.
- Tell the learner to write all their suggestions on their work on how to improve their skills and if they follow their suggestions, it should influence their grade (Gibbs, 1992).

Feedback should be exchanged between receiver and sender in order to develop teaching. Therefore, it will be useful to give feedback in two ways

(MacDonald, 2000). That is teachers provide feedback for students to improve their learning then students give feedback to teachers to improve their teaching.

Nevertheless, instructors should be aware of the time response i.e. they should return the students task quickly before moving to the next lesson. On the one hand not only does effective feedback concern instructors' response, but also the students' response. If students hand in the revision of their task late, it could affect the instructors' response, taking up needless time. Although feedback response from students to teachers is useful for teachers to improve their feedback to the students, in our context we pay special attention to feedback from the teachers to the students and feedback dialogue response between the system and the TAs.

2.2.2.8 Feedback summary

In terms of feedback summary, Gibbs (1992) suggested that informing feedback to all learners is to remind them of their main errors in order to develop their skills before handing in the next assignment.

2.2.2.9 Timing of feedback

Race (2001) argued that teachers should give feedback to the learners in an appropriate time frame. It is not useful if they provide feedback after students have finished their exams because students do not have enough time to practice before the next exams. Therefore teachers should give feedback after they finish teaching or after returning marked assignments to the learners. It should be an early assignment and should not be the last assignment of the course. Further, Gibbs (1992) suggested that if the tutors do not have enough time to provide immediate feedback, they should mark and return the assignment to the learners as soon as possible in order to return their next assignment in time. Thus, providing feedback on students weaknesses from their assignments would be helpful for them to improve their learning before taking the final exam.

2.2.2.10 Continuous Feedback

Giving good feedback might be continuous. Sadler (1998) suggested that learning with formative feedback should be real feedback. Considering giving feedback on

assignments, teachers should give continuous feedback according to each assignment or sub-task. This can generate a students' motivation; however, they may overlook given feedback. Thus, if the teachers return the assignment without giving any grade or score to the learners, it would increase student motivation without raising anxiousness regarding their scores, and then they could pay attention to the given feedback message.

2.2.2.11 Positive feedback

Positive feedback may be concerned with, for example, commenting favourably on the ability, intelligence, effort or achievement of the student. Giving positive feedback can be encouraging for learners (Race, 2001). Condition 6 of Gibbs and Simpson's study (2003) states, "the feedback focuses on students' performance, on their learning and on actions under the students' control, rather than on the students themselves and on their characteristics". Related to this, Mueller & Dweck (2004) found that children who were praised for ability (or intelligence) after success eventually led to a negative effect for their achievement motivation when they performed poorly later. Their conclusions were that it is better to praise for effort as well as for ability. Praise is positive feedback (or encouraging messages) to students. On the other hand, negative feedback is a message to the student either explicitly or implicitly points out student's weaknesses (or failure) in the student. Even information for improving learning may be taken as (implicitly) negative.

We need to be careful about the perceived effect of feedback i.e. before giving positive feedback to each student, teachers should consider the motivation of each individual student depending on his/her understanding of the student's emotional and cognitive states in order to determine what can be said positively about the student's performance and what might be taken as negative feedback. The aim is to provide positive feedback which is believed to help in the development of student's 'intellectual performance' (Dweck et al., 2004). In other words, the teacher gives positive feedback to students after success, then next time they fail the teacher still gives them positive feedback (though different in content e.g. first give 'Excellent', next time give 'Good'). In this manner, the students' feeling is negative rather than positive. In addition, giving negative

feedback increases the risk of demotivation (Wootton, 2002). Consequently, giving positive feedback before negative feedback, or information for improving learning, and following this with positive feedback reduces the risk of demotivating the student. Such a 'sandwich' of positive feedback, information and then positive feedback would be helpful for the learners to improve their learning and reduce the risks of demotivation (Eckstein et al., 2002).

2.2.2.12 Advance feedback

Before any assignments, teachers should give the learners advanced feedback that highlights the most common mistakes made by learners, then the teachers could give feedback to the learners by not referring to the previous feedback which they have not been given (Race, 2001).

2.2.2.13 Restatement correct answer feedback

Considerable research suggests that feedback would be very useful when it restates the correct answer instead of telling right or wrong (e.g. Butler & Winne (1995)).

As a consequence, this has several characteristics of good feedback, it can be argued that providing good feedback, followed by a number of problems as stated above as well as prioritising the provision of the good feedback could help the learners improve their life long learning.

2.2.2.14 Summary: Characteristics of poor feedback

Although provision of good feedback should rely on Section 2.2.2.1-2.2.2.13, giving some good feedback may be difficult because of the problems of giving good feedback. In other words, they are like characteristics of poor feedback. In fact, providing feedback is a crucial part of learning and teaching even if in large classes when it is difficult to give individual feedback to every student. As mentioned in Chapter 1, describing a number of problems from giving good feedback.

According to the section on the problems of giving good feedback (see Chapter 1), we should avoid these so they do not appear on the teachers' feedback before they are returned to the learners. In addition, even if giving individual

feedback can reduce the gap between teachers and students in which they could receive effective, quantity feedback, thinking about the learners' feelings when they receive their assignments back (Race, 2001), it increases the teachers workload substantially, especially in big classes. Besides, in order to reduce the problems of giving 'positive feedback' stated in Chapter 1, teachers should give more detailed feedback and give the reasons on their work (Race, 2001).

It is vital for teachers to know which poor feedback to avoid giving to students. With regards to tutoring systems, if the tutoring system provides too much feedback to users, they may not think by themselves and always wait for the system to respond for them. By contrast, if the tutoring system provides too little feedback, users may be disappointed that they cannot see the correct answer (Smith, 1997). This may be considered poor feedback as it may not be the best way for the tutor to give such feedback to students.

Commonly, the source objective to send feedback involves receiver behavior, opinion, value or action (London, 1995) such as some learners may be unhappy that they work on their assignment with abundant effort, but they obtain inadequate feedback; however, it is not easy to provide feedback in large classes with more detailed feedback to all learners (Brown & Knight, 1994).

Giving individual feedback perhaps encourages inconsistency due to the difference of students' ability and the difference of their effort. Giving consistent feedback can stimulate students' interest rather than giving inconsistent feedback or no feedback followed by detailed feedback, they should consider the previous feedback they provided to the students (Brunot et al., 2000). Nevertheless, if each teacher gives inconsistent feedback, it may result in students' confusion.

In fact, providing feedback is extremely important when learning is still by trial and error (Race, 2001). Normally, most teachers give only feedback when students make a mistake, but they do not give feedback to students who are right (Race, 2001). This is often found in feedback giving situations where it may harm and reduce learners' motivation when some of them have not completed the learning process. Therefore, this is like Answer Until Correct feedback (AUC) (Ross & Morrison, 1993). In this manner, this would be helpful if the teachers consider an appropriate level of help for the learners depending on their contingent errors.

2.2.3 Techniques of giving good feedback

Wager & Mory (1993) observed that different feedback depended on different types of learning and also found that feedback is always related to a response generated by a question (p.70). They also recommended using questions and feedback associated with the stages of the information processing (p.71) (see Chapter 3) which is a technique to teach TAs to give feedback according to the given situation i.e. giving feedback from the error result of program analysis. Denton (2003) suggested that preparing feedback before giving it to the student could help the teachers provide feedback to students quickly. From condition 7, "The feedback is timely in that it is received by students while it still matters to them and in time for them to pay attention to further learning or receive further assistance", in the study of Gibbs & Simpson (2003), they found that the giving of feedback should be done during the course. If teachers provide feedback after finishing the course, it might not be useful for students. Gibbs and Simpson also suggested the approaches to help students read feedback in condition 10 of their research -"feedback is received and attended to", for instance, giving feedback by no mark; giving assignment by self-assessment or peer assessment; giving the second assignment by understanding of the first assignment. From these techniques, giving feedback with no mark is very interesting to adopt in our context.

Giving feedback to learners might depend upon errors they have made. A better approach is to use relations between elements in a domain to built a knowledge representation technique (Smith, 1997). Then build the relation by using the fundamentals of providing feedback in which this depends upon the structure of knowledge representation technique and the types and size of domains can be flexible (Smith).

When supplying feedback, there is no general rule in which it depends upon the domain and the circumstance to require feedback. Therefore, Smith (1997) suggested some useful approaches from his literature: optimal path, authoritarian, issue based feedback, device based feedback, use of primitive operation only and the use of high order operations.

- Optimal path: VanLehn's (1996) ANDES uses this approach if students do not follow the path, they will receive immediate or negative feedback. If students do not pass any process, the system will help learners by help-request from a why-question then a hint-message will be provided. The disadvantage of this approach is that students have to follow all definite paths (Smith, 1997).

- Authoritarian: This approach was used in the LISP tutor (Anderson Reiser, 1985) which is like an optimal path.

- Issue based feedback: In this approach it is difficult to define a domain because learners have various skill levels. However, the system may specify that learners have inappropriate skill, then informs the learner indirectly, then gives immediate feedback.

- Device based feedback: This approach provides a flexible approach to system exploration and allows the user to complete problems using any possible solution to fulfill a goal.

- Use of high order operations: This approach is based on the system giving immediate feedback; however, its disadvantage is some immediate feedback may prompt students to guess the answers.

VanLehn (1996) suggested giving minimal feedback, emphasizing that teachers should provide only feedback that will help students to solve their previous errors. Smith (1997) proposed five help commands of domain feedback to provide appropriate feedback according to users' request: help, commands, hint, why, and how. Considering quality feedback, there are assessments to support student learning, they are: quality and time of feedback (e.g. giving enough feedback, often giving feedback, giving enough information in any detail); quality of giving feedback emphasized in learning by receiving feedback from teachers rather than students' learning by themselves; feedback link to assignments' objective and criteria to grades (Black & Wiliam, 1998; Gibbs & Simpson, 2003). However, quality of the following feedback depends upon students' responsibility to improve their learning. There are the examples of feedback questions: "Can you tell me a little more about that?"; "What happened after that?"; "What did you think about that?"; "Why is that?"; "Why do you think

that is?"; "I'm not sure what do you mean...?"; "Why didn't you...?"; "I'm not sure, I quite understand" (Smith, 1997). Feedback questions are quite open questions by nature, which might lead to difficulties for the learners to answer some of the questions. For more detail on techniques of giving good feedback, see Chapter 3: Feedback Design.

2.2.4 Effective assessment with good feedback

Generally, teaching and learning should be completed when there is effective assessment. In order to improve learning, it is vital to provide more detailed feedback to students (Race, 2001). In higher education teaching, even though feedback is the main part of teaching, it is ignored from the teaching process (Ramsden, 1992). It is disagreeable for students when they receive their assignment back without any feedback in which they receive only a mark or grade and also there is seldom extended feedback that concentrates on students weak points, as this is difficult to give to students (Rawles et al., 2002). This is perhaps because they are afraid to explain the reason why the students make the mistakes which result in low scores.

Beyond this, giving feedback is a part of reflective learning in which it is a central skill of assessment (Brown et al., 1997). According to effective teaching strategies, there should be no barrier between teaching and assessment to provide feedback to the learners (Ramsden, 1992). It is not an easy task to write comments on students' work for quality teaching. Especially, for large groups of learners, most tutors may give instant or immediate feedback on students' assignments beyond their errors with a summary of recommendations and a description of further reading; thus, assessment should also serve as a feedback function for teachers (Ramsden, 1992). However, we should understand most teachers may rush to mark assignments so they can return the students' work in time before the next lesson without considering important errors or specific feedback. Giving detailed feedback on every error may reduce learners' interest in the previous lesson due to inadequacy of time to take in the feedback and fully understand it (Gibbs & Habeshaw, 1992). In this manner, giving priority to quality feedback should be noted.

Larson, et al. (1986) found that the components of feedback such as timeliness, specificity, frequency and sensibility, involved giving both positive and negative feedback, which is less influential than assessing feedback. Similarly, feedback could be given together with assessment. Ramsden (1992) suggested one of six key principles of effective teaching in higher education, namely "appropriate assessment and feedback". For example, education in Australia emphasizes students' improvement on quality of feedback.

Furthermore, providing feedback relates to effective assessment. One of Ramsden's fourteen rules for better assessment in higher education states "never assess without giving comment to students about how they might improve" (Ramsden, 1992). Nevertheless, it should depend on each learner's improvement and their previous skills. Thus, the aspect of assessment would involve a social, personal and historical record of each individual learner (Brown et al., 1997).

With regards to learning styles, there are the four phases of the cycle and the corresponding learning styles of the learning cycle of Kolb (quoted in Brown et al., 1997) including activity, reflectors, theorists and pragmatics, "Reflectors do not like to be rushed. They prefer to learn through assimilating information, reflecting upon it and their experience and reaching decisions in their own time". Even though giving instant feedback is an important aspect of Computer Based Assessment (Higgins et al., 2002), in our context, due to the learning cycle, it is best to provide a short delay before returning students work, for example, few days after submission.

A large number of researches fail to consider timeliness. Indeed, teachers should give feedback one or two days after submission. The earlier feedback is returned, the more effective it is for students who will still remember what comments they received and should improve next time around (Race, 2001). In this aspect, computer-support could help the teachers to provide quality feedback in a short period of time.

In addition, from condition 7 in the study of Gibbs & Simpson (2003) feedback should be returned during the course. If teachers provide feedback after finishing the course, it might not be useful for students. Nevertheless, it could be

useful for them in the future; thus, we argue that giving effective feedback could help students to learn by themselves in a lifelong learning capacity.

Of course, effective assessment might rely on good feedback. How can teachers be helped to give good feedback to a large class? According to the following paragraph employing a marking assistant would be helpful for them.

2.2.5 Marking Assistant for good feedback

For almost all teaching and learning at undergraduate level, each tutor may teach large groups of learners. In order to provide quality teaching, all tutors have to pay attention to, and follow closely the development of learning for each learner. In particular the teacher is required to provide individual feedback or instruction and assist each student. Appropriate and common methods of supporting the development of learning for each student can be achieved by providing feedback either after teaching or after marking an assignment.

In the case of large classes, most teachers are often unable to support either high quality or appropriate feedback for each learner. In order to alleviate this problem, many universities employ marking assistants, for example either automated marking assignments or hire senior students as teacher assistants (TAs) to help the lecturer. To manage the teachers time spent on marking assignments and providing feedback, they may assign a team of postgraduate tutors or experts from more senior years to mark assignments instead. For example, in the Department of Philosophy at Leeds University, the 'tutor' gives 5-minute feedback sessions to each student as well as written comments according to (Brown et al., 1997). In this manner, it should nevertheless depend on either the School/Department or University policy to employ marking assistants.

However, TAs, often do not have the experience to provide quality feedback, they are like new teachers. Can we help the TAs to help the teachers? Many teachers are too busy to give feedback and they have problems finding ways to give consistent feedback (VanLehn et al., 2003). As a result of a large number of students or teachers being too busy to give feedback to all their student assignments, the TAs are assigned to provide strong support to the learners. For

example, in one-to-one tutoring students achieve more understanding, greater motivation, and can work quicker (Slavin, 1987).

2.2.6 TAs have to be trained

Various people who have marked learnt how to give feedback (Elawar & Corno, 1985); however, they have not learnt how to give effective feedback i.e. many teachers are seldom trained to give effective feedback (Brown et al., 1997). This is perhaps the cause of ineffective teaching and learning problems in higher education. In fact, most TAs are like new teachers i.e. they are peer students who are little older than students themselves, and as such it is extremely rare for them to be trained to teach by giving feedback via scaffolding approach and greater explanation to the learners (Fitz-Gibbon, 1977). Brinko (1993) also investigated the requirement in the cognitive processes of consultants and how teachers make decisions within the feedback session: how they decide which information to feedback to the faculty client, why they structure their sentences and phrase their comments as they do, why they choose certain words over others, how they decide to frame a problem, how they decide to name a problem, and how they offer solutions and how they use silence (Brinko). For this reason, it is very interesting to research this issue.

Besides, training the tutors to give good feedback could be a good approach to provide them with the skills on how to evaluate other people's work and feedback. Brown et al. (1997) presented four approaches of tactics for modular assessment.

- Provide feedback on every assignment, especially, the best two or three assignments;
- Tell everyone what they must do and submit the two best assignments for final evaluation;
- Assign the first assignment to be the model of feedback in which the learners can improve their next work;
- Students must submit brief assignments before the final submission then the tutors will mark and supply feedback alongside evaluation grades.

Thus employing a marking assistant could help the teachers mark, as both novice teachers or TAs who lack experience of giving good feedback, and could assist the teachers if they are provided with training, so how can they be trained? They could learn from several situations as discussed in the following section.

2.3 How do people learn to give good feedback?

Observations from giving feedback to students, and the comments from mentors could be a useful way to help people learn how to provide feedback? What are the criteria that markers desire to know which will be explored in the following sections?

2.3.1 Giving Feedback to students

Kumar (2003) found that producing feedback can help each learner improve himself; however his system did not evaluate the quality of feedback. In terms of learners, TAs can be students who learn to give feedback and this can apply to our context for establishing feedback to help the TAs learn to give feedback. In this section we will consider ways to give feedback to students, and ways of learning to give good feedback schemes.

2.3.1.1 The way to give feedback to students

Giving feedback to students may involve face-to-face, asynchronous communication or synchronous communication (e.g. the use of instant messaging). As a result, we can deduce to apply such approaches on how to help teachers to give feedback to students effectively.

On the other hand, when teachers try to give feedback on traditional methods of assessment, Rawles et al. (2002) found that it is difficult to provide quality assessment to students, i.e. giving detailed or specific feedback. They reported that it is a weak form of giving feedback to students. They observed restrictions in giving feedback forms to students so face-to-face tutoring or employing technology or software tools to support the teachers to give feedback could help them provide more detailed or specific feedback to the learners.

2.3.1.1.1 Face-to-Face Tutoring

According to Bloom (1984), "One-on-one tutoring allows learning to be highly individualized, and consistently yields better outcomes than other methods of teaching". In historical teaching and learning, there are a number of examples of one-to-one tutoring in which teachers and students can communicate closely (e.g. (Wasik & Slavin, 1993)). This results in highly effective learning. However, currently, by economy, the number of students in each class has increased greatly (Mohan, 1972; Cohen et al., 1982; Bloom, 1984). In this manner, it is necessary to have senior students to act as tutors to help students' learning. However, they have not been trained to teach in an effective way (Graesser et al., 1995; Graesser et al., 1999).

2.3.1.1.2 Mediated technology

In general, it is accepted that technology is the most suitable tool for delivering feedback which may be via either asynchronous or synchronous communication as the following section explain;

2.3.1.1.2.1 Asynchronous Communication

Generally, asynchronous communication allows learners time to reflect on a topic before posting a reply. There are several different types of asynchronous communication tools that teachers can use to provide feedback to students e.g. Frequently Asked Questions (FAQ) that provide an answer to a number of similar questions; Answer Gardens (Ackerman & Malone, 1990; Ackerman & McDonald, 1996) that require more intelligence than FAQ in which some questions that the system can not answer will be sent to the appropriate expert and returned to the user in the network of Answer Gardens; Electronic Feedback via E-mail (Denton, 2001b, 2001a, 2003); Electronic-Tutor Marked Assignment System (e-TMA) (Moreale et al., 2002; Whitelock et al., 2003); in College Courses (Collins-Brown, 2001). Although those systems can provide answers or feedback to the learners, they do not provide any tools for the novice teachers or TAs to learn to impart quality feedback in the area in which they require the most assistance.

2.3.1.1.2.2 Synchronous Communication

Basically, synchronous communication allows learners to reflect the discussion from different locations at the same time. There are several difference types of synchronous communication tools that teachers can use to provide feedback to students e.g. chat[6], desktop videoconferencing (DVC) and GroupWare[7], according to (Salter, 2002). In addition, peer assessment systems are a new technology, developed by Bhalero & Ward (2001), can enhance assessment practice with reference to seven principles of good feedback practice (Nicol & Macfarlane-Dick, 2006). Technology offers the possibility of assessing online discussion, and supporting the rethinking of how to regenerate concept of assessment in higher education (Nicol & Miligan, 2006). Technology provides tools for students to reflect on their performance and also help the teachers improve their teaching. Specifically, the tools described by Nicol & Macfarlane-Dick (2006) are not designed to help teachers reflect on improving their feedback giving. However, aspects of the tools could be helpful for designing a system to help the novice teacher reflect on their performance in learning to give good feedback.

2.3.1.1.3 Example Feedback Tools

There are several software tools to help the teacher to give feedback in various domains, for example, in science marking (e.g. Ms.word-excel-marking (Denton, 2003)), in essay marking (e.g. TMA (Thomas, 1998); e-TMA (Moreale et al., 2002; Whitelock et al., 2003)), in mathematics marking (e.g. Online Exercise System (Bryc & Pelikan, 1999; Sapir, 1999), in Programming marking (e.g. SPROUT (Pardoe & Vickers, 1994; Rimmer et al., 1995), Ceilidh (Foxley et al., 1999), CourseMaster (Foxley et al., 2001; Higgins et al., 2002), BOSS (Joy & Luck, 1998), AssesmentMaster (Suhonen et al., 2001)). These tools are described in more detail in the next section. Although these tools can help the teacher to evaluate the students learning, they do not supported training of novice teachers or TAs to help them learn to provide quality feedback.

---

[6] Chat is used to describe two or more people using the Internet to conduct a discussion in the real time (Brna, Irvine, Duncan, Karamanis, 2001)

[7] This tool allows the participant to discuss more activities than text based discussion and also allow the user share white board to share document and other applications (Brna, Irvine, Duncan, Karamanis, 2001)

2.3.1.2 Learning to give good feedback schemes

Giving good feedback schemes could be a pilot for novice markers to follow regarding the following sub-sections.

2.3.1.2.1 Fostering students self-critical thinking

In general, there are still some tutors that give feedback to foster students to improve their self-critical thinking via their assignments and then return the assignments to them (Ramsden, 1992). One method of providing good feedback may be to give feedback to learners on the same assignments, so they can examine their improvement and then award a final mark to grade them (Ramsden, 1992). Thus, it should be a better way to evaluate how the learners improve their learning from receiving feedback.

2.3.1.2.2 Employing semi-automated marking

Nevertheless, there are still some students who prefer traditional marking and do not like automated marking systems that do not provide detailed feedback or comments after assessments, giving only pass or fail instead (Ramsden, 1992). If there are any errors with the system, it may cause students to fail when it is not their fault. Thus, the human markers should be involved in a form of automated marking called semi-automated marking.

2.3.1.2.3 Advice strategies

Beneath providing feedback, Gibbs (1992) asserted that when assessing students' work teachers should:

- *Encourage students to judge their own work*, for example, most students may overlook their errors on their work before submission, thus the tutors should tell them to evaluate their work before final submission.
- *Give feedback promptly*, for example, the tutors should give instant feedback to the learners because if it is delayed, it may be a new subject when it is returned which may cause a lack of interest for the learners.

However, it could be argued that practice programming each lesson may be related to each other so that the learners could gradually drill programming skills and understand how to debug their errors. In this practice, if we give student

feedback promptly, they could find the same errors again because they have not recognised how to correct their errors. Thus, it is better to give important errors alongside prompt feedback. *(3) Be positive in your feedback*, for example, tutors should provide positive feedback to the learners who work well in order to encourage and advise them for next piece of work.

2.3.1.2.4 Written feedback

Giving useful feedback might be more helpful for students when teachers give large assignments and return a great deal of feedback, quickly and with sound advice while giving feedback from teaching, doing laboratory, etc. in which this does not give feedback via assignment, for example, oral feedback or informal feedback, is less helpful (Gibbs et al., 2003). Therefore, giving written feedback should be considered more valuable than oral feedback; nevertheless, if students do not choose to read feedback, they may not improve their learning. Therefore, one way to help them read feedback is to make sure that written feedback pays special attention to individual learner performance.

2.3.1.2.5 Feedback on assignments

Condition 8 of Gibbs & Simpson (2003) states, "Feedback is appropriate to the purpose of the assignment and to its criteria for success", there is some evidence; theory and empirical experience to support this condition that giving feedback on assignments in order to correct errors; explanation improves understanding; suggesting next learning step to create more learning; improve skill from practice rather than from content; encouraging students' reflection in assignments and continuing studying.

2.3.1.2.6 Giving indirect feedback

From the study of Graesser et al. (1995), it could be suggested that tutors did not give further comment regarding student errors. Nevertheless, Graesser (1993) also analysed the feedback to student contribution by tutors at all quality levels: error-ridden, vague, partially correct, and completely correct. From the results of the findings (Graesser et al., 1995), they might not guarantee that feedback can help students to solve error-ridden and vague student contributions. Besides, tutors seldom acknowledge the contribution was an error-ridden contribution.

Of course, we believe that providing polite, indirect comment to error-ridden answers plausibly appropriate to students rather than rough, direct negative feedback. This also has the potential to encourage students to improve their learning, as reported by the research on skilled tutors by Lepper et al. (1990), McArthur et al. (1990), and Lepper et al. (1993). In addition, there are two observations of major difficulties to help students solve their errors by Graesser (1995). Firstly, students do not know how to expose and repair their errors to improve the metacognitive skill of self-regulating their knowledge (Collins & Brown, 1988; Schoenfeld, 1988; Scardamalia et al., 1989; Bangert- Drowns et al., 1991; Scardamalia & Bereiter, 1991; Merrill et al., 1992). Secondly, students are less confident when they receive negative feedback. The way to give soft and indirect conduction when students confront errors, bugs and misconception is better than direct negative feedback and reparation in which those tend to be students' weaknesses. In order to support this, there is some evidence that skilled tutors use a soft indirect way rather than a rough direct way (Fox, 1991, 1993; Lepper et al., 1993).

Hence, it might be better to give indirect feedback to novice teachers or TAs and at the same time to teach them to give indirect feedback. Nevertheless, those studies could be improved by including a structure of giving feedback. In fact, even if Graesser (1995) analysed feedback to students, not to teachers, there appears to be some potential to apply this to help TAs to give feedback to students.

2.3.1.2.7 Giving feedback for contribution

Good tutors should provide students with feedback to help them obtain the quality of their contribution; however, even though giving feedback is provided by skilled tutors and Intelligent Tutoring Systems, detailed feedback is very complicated to achieve (Graesser et al., 1995). Gibbs & Simpson (2003) suggested approaches to help students read feedback in condition 10 of their research, for instance, giving feedback by no mark; giving assignment by self-assessment or peer assessment; giving the second assignment by understanding of the first assignment. It seems to be a good way to allow the students to resubmit their assignments in order to provide feedback for contribution.

As a consequence, these are the issues which TAs should be concerned with while providing feedback on students' assignments.

2.3.2 Approaches of promoting improvement

Most educators emphasized giving feedback to students (Denton, 2001b, 2001a; Moreale et al., 2002; Denton, 2003; Whitelock et al., 2003); however, they have not focused on feedback to the teachers in which there are very few studies about giving feedback which emphasises training teachers (e.g. written feedback on students homework (Elawar & Corno, 1985; Chi et al., 2001), providing feedback of human tutoring (Chi et al., 2001)). In this section, we will explore how they help both in training teachers and students to learn in order to employ their approaches for training teachers to give feedback? i.e. the way to give feedback to teachers to provide better feedback to students. In order to perform this we will start with Modeling of good feedback from Mentor (Section 2.3.2.1) that consists of Naturalistic Tutoring and Interactive Tutoring (Section 2.3.2.1.1), Reflective Practitioner (Section 2.3.2.1.2), Cognitive Apprenticeship (Section 2.3.2.1.3) which includes Cognitive Apprenticeship Framework (Section 2.3.2.1.3.1), Situated Cognition and the Culture Learning (Section 2.3.2.1.3.2), Legitimate Peripheral Participation (Section 2.3.2.1.3.3), and Situated Learning in Adult Education (Section 2.3.2.1.3.4) then continue to Suggestions from teachers (Section 2.3.2.2) which consists of Training Strategies (Section 2.3.2.2.1), Naturalistic tutoring protocol (Section 2.3.2.2.2), and Prompt/Scaffolding (hint, suggest) (Section 2.3.2.2.3) there after continues to Learning from student comments (Section 2.3.2.3) and finally Learning from teacher comments (Section 2.3.2.4).

2.3.2.1 Modeling of good feedback from Mentor

Modeling of good teaching skills may be a part of the solution of our context i.e. seeing somebody give good feedback or showing TAs good feedback examples for them to adapt the mechanism. A good module for giving feedback can be a prototype for the teacher to learn how to give feedback, for example the result of examining natural human tutoring (Chi et al., 2001), the approach of cognitive apprenticeship (Collins et al., 1989), situated learning issues (Brown et al., 1989),

and the issue of Legitimate Peripheral Participation (Lave & Wenger, 1991) which could help the TAs to give more guidance.

2.3.2.1.1 Naturalistic tutoring and interactive tutoring

Chi et al. (2001), carried out two studies, a study of naturalistic tutoring and a study of interactive tutoring. In the study of naturalistic tutoring, they used unskilled tutors with no experience tutoring and training per se. and were told to tutor students naturally, no format was provided for them, in which to observe tutoring of unskilled tutors, thereafter they achieved tutoring protocol. From the result of examining natural human tutoring in which they trained teachers to give more guidance, the substantive statement is categorize into:

- Giving explanations
- Giving direction (either positive or negative) feedback, followed by a short corrective explanation if the feedback is negative, such as "No, when it went through..." –negative response then explain the correct answer
- Reading text sentences aloud
- Making self-monitoring comments (in which the tutors commented about their own instructions such as "I don't know if this will help you" or commented about the tutors' own understanding of the materials, such as "I don't know why they put [that line of text] in there, it just kind of confused me")
- Answering questions that students asked
- Asking content questions (such as "Which is the upper and which is the lower chamber of the heart?" –ask learners from the content of lesson.
- Scaffolding with generic and content prompts.
- Asking comprehension –gauging question (such as "Is this starting to stick?").

The categorization of 1-4 is non-interactive move, the categorization of 5-8 is interactive move, the categorization of 1-6 is self-explanatory, and the categorization of 7-8 is clarification (give more example). Beyond this, they

revealed that there are several kinds of guiding activities for scaffoldings. Those are

- pumping for "what else"
- hinting (e.g. "So, it's kind of leaving out the lungs here?"
- fill-in-the-blank kinds of request (e.g. "OK", so, it come from …")
- highlight critical features
- decomposing the task
- executing parts of the skill
- providing physical props or cue cards
- describing the problem so as to orient the student to the important features
- comparing the current problem with a previously solved problem
- maintaining goal orientation or reminding the student of some aspect of the task
- completing the students' reasoning step or "spicing in" (or jumping in and providing) the correct answer when the student commits an error, without acknowledging that an error has been made
- initiating the beginning of a reasoning step or a task
- asking a leading question (e.g. "And when do you think it goes?)"
- redirecting the student
- providing an example.

According to tutoring protocol and guiding activity for scaffolding, it could be helpful to apply some to scaffold the TAs to give feedback to students in our context.

In the studies of Chi et al. (2001), they had three hypotheses from the first study from which their studies did not cover the evidence to support all hypotheses even though there was the evidence to support T-hypothesis, tutors' move, from the review of Chi et al.; there was also evidence to support S-hypothesis, the students' response, for useful response of students in the tutoring context; and there was the evidence to support I-hypothesis that interactive responses involve learning rather than non-interactive responses. Nevertheless, in practice, they assumed that it was possible to do S-hypothesis rather than T-hypothesis, i.e. student response is more important than tutor scaffolding. The

result of the I-hypothesis study found feedback did not help the student respond constructively. This resulted in teacher feedback that did not relate to learning. They presumed that this was because student responses were not systematic. Due to the fact that the results of the first study were not clear, they conducted the second study –interactive tutoring.

From the second study, if a classroom is interactive rather than didactic, teachers may not control the tutoring dialog as much as prompting. As a result, teachers should start their dialog first; however, Chi et al. (2001) found that students started their dialog first. Besides, the number of teachers' statements from the first study was more than student responses; in contrast to the second study, the number of students' statements was more than teachers' statements. To conclude, from the second study Chi et al. found they were successful in decreasing teachers' explanation and feedback in order to give an example of scaffolding comments for teachers to use to scaffold students instead. However, most teachers used scaffolding prompts rather than context-free prompts, by comparison. They revealed that this was because giving scaffolding prompts is easier as teachers only provide scaffold to student's response in order to complete the teaching detail correctly. The benefit of giving scaffolding to students is to help students understand the study concept of the lesson. Chi et al. (p. 517) concluded that there were three effective results from their research:

- an interactive style is more inspiring and creates a pleasant learning process
- guidance is better provided by prompting or scaffolding than giving a single explanation
- hints in the form of advice work well for an interactive style of tutoring.

Therefore, the studies of Chi et al. could be useful to adopt in this thesis, for example, we could guide the TAs to learn how to give good feedback in which it is vital to next generation Intelligence Tutoring System. These appear good ways to give feedback to students and apply to teachers to learn to give feedback even if the studies of Chi et al. have not concentrated on how the teachers learn to give better feedback and their approaches are focused on giving feedback to students,

and they focused on scaffolding prompting rather than an interactive style. Therefore, providing scaffolding prompts could be useful to apply in our context.

Chi et al. (2001) reported that unskilled teachers used scaffolding prompt rather than context-free prompt because it is easier to deliver to students. It could be helpful to apply to our context a teaching system that could be programmed to give context-free prompts equal to or more than scaffolding prompt by guiding the TAs' thinking, as well as providing context-free prompt in the form of 'key questions' with regard to the learner's assignment. This should provide a better way to give balance. Furthermore, the conclusion of Chi et al. results could be helpful to apply to our context. In the first study, if the teaching system teaches the TAs to give feedback interactively, the TAs may enjoy learning how to give better feedback. In the second study, a teaching system should prompt/scaffold rather than teach all processes of giving feedback. As the TAs may not remember all the processes they should learn from real situations. In the third study, the results hinted that in an interactive style of teaching students received better feedback.

2.3.2.1.2 Reflective Practitioner

Schön (1983) offered an approach to epistemology of practice based on a close examination of what some practitioners – architects, psychotherapists, engineers, planners, and managers – actually do, he also collected a sample profile from occupations focusing on situations where junior employees are trained. In his analysis of these cases, he described the assumption that capable practitioners usually know more than they can say. They mostly exhibit a kind of knowing-in-practice silently so he presumed that it was possible to construct a test model of knowing from the actual performance protocol because practitioners frequently reveal their capacity of reflection in the middle of an action to deal with the unique, uncertain, and conflicted situation of practice. In addition, he asserted that the role of practice for the professional depended on technological changes (e.g. in medicine, engineering, business management and education) and also practitioners frequently confuse the values, goals, purposes, and interests (e.g. when teachers are faced with pressure for increased efficiency in the context of contracting budgets, institutions demand that they rigorously "teach the basics", to

encourage creativity, build citizenship, help students to examine their values) "Each view of professional practice represents a way of functioning in a situation of indeterminacy and value conflict, but the multiplicity of conflicting choice among multiple approaches to practice or devise his own way of coming them", according to Schön (1983).

In terms of the dominant epistemology of practice, there are three components to professional knowledge:

- An underlying discipline or basic science component upon which the practice rests or from which it is developed;
- An applied science or "engineering" component from which many of the day-to-day diagnostic procedures and problem solutions are derived;
- A skill and attitudinal component that concerns the actual performance of services to the client, using the underlying basic and applied knowledge (Schön, 1983, p.24).

These components could be helpful for the TAs to practice giving feedback to students.

In terms of Knowing-in-action, Schön (1983) stated that there was nothing in common sense to make us say that know-how consists of rules or plans which we entertain in the mind prior to action. Although we sometimes think before acting, it is also true that in much of the spontaneous behavior of skillful practice he revealed a kind of knowing which did not stem from a prior intellectual operation. It seems that learning by doing results in more experience when the learner has enough practice. In addition, according to Schön (p.54), he explained the characteristics of knowing in action which has the following properties.

- There are actions, recognitions, and judgements which we know how to carry out instinctively; we do not have to think about them prior to or during their performance;
- we are often unaware of having learned to do these things; we simply find ourselves doing them;
- In some cases, we were once aware of the understandings which were subsequently internalized in our feeling for the stuff of action.

In other cases, we may never have been aware of them. In both cases, however, we are usually unable to describe the knowing which our action reveals.

With regard to reflection-in-action, it is thinking about doing something while doing it in which Schön (1983) called "reflective conversation with the situation". He also pointed out that reflection on the tacit understanding which have grown up implicitly in practitioner's action around the repetitive experiences of the specialize practice, understands which he surfaces, criticizes, restructures, and embodies in the further action that can allow the practitioner to experience. It is this through process of reflection-in-action which is central to the "art" by which practitioners sometimes deal well with situations of uncertainty, instability, uniqueness, and value conflict (p. 49). In terms of reflecting-in-practice, Schön noted that it was considering what a practice was and how it was like and unlike the kinds of actions in which a practitioner's reflection can serve as a corrective to over learning (p.61). Although reflective-in-action is an extraordinary process, Schön observed that it was not a rare event in which for some reflective practitioners, reflective-in-action is the core practice. Nevertheless, because professionalism is still mainly identified with technical expertise, reflection-in-action is not generally accepted as a legitimate form of professional knowing (p. 69). Consequently, despite this, TA's should be allowed the opportunity to practice giving feedback under unpredictable situations. Schön also reported that most practitioners may find uncertainty in their technical experts, and as a result they are unsure what to do; therefore study of reflection-in-action is critically important. In order to solve such problems, developing an epistemology of practice which places technical problem solving within a broader context of reflective inquiry, shows how reflection-in-action may be severe in its own right. Further if we link the art of practice in uncertainty and uniqueness to the scientist's art of research it could increase the legitimacy of reflection-in-action and encourage a broader, deeper, and more rigorous use, according to Schön (1983).

In addition, Schön (1983) proposed the different constant that various practitioners bring to their reflection-in-action. These are the media, languages, and repertoires that practitioners use to describe reality and conduct experiments; the appreciative systems they bring to problem setting, to the evaluation of

inquiry, and to reflective conversation; the overarching theories by which they make sense of phenomena; the role frames within which they set their tasks and through which they are bound to their institutional settings. According to such constants, we should consider them in order to design a system to train the TAs in which computer-support could help the TAs reflect their action on how to provide good feedback.

Even in the study of these sorts of reflection, crucial both to professional development and to the epistemology of practice, there have been limits to reflection-in-actions in which the practitioner may not frequently think about what they are doing while doing it. Furthermore, reflection-in-action does not depend on a description of intuitive knowing that is complete or faithful to internal representation, according to Schön (1983). For example, considering novice teachers, a reflective teacher requires a kind of educational technology which does more than extend her capacity to administer drill and practice in which an educational technology could help students to become aware of their own intuitive understandings, to fall into cognitive confusions and explore new directions of understanding and action, according to Schön (p. 333).

In summary, the idea of reflective practice is an alternative to the traditional epistemology of practice. It leads to new conceptions of the professional-client contract, the partnership of research and practice, and the learning system of professional institutions; in a sense both similar to and different from the radical criticism, to a demystification of professional expertise; the scope of technical expertise is limited by situations of uncertainty, instability, uniqueness and conflict, according to Schön (1983).

Thus, employing a computer supported system as a tool to help the teachers to provide quality feedback and at the same time learning to give good feedback could support them in using educational technology. The system might inform the TA how they improve on their learning to use tools of the system and he/she might reflect on this information

2.3.2.1.3 Cognitive Apprenticeship

Novice teachers often learn to mark from mentors, they then evaluate how they mark in order to improve their marking. In this manner, it is in line with the modeling method of cognitive apprenticeship approach (Collins et al., 1989) that is useful for training.

The studies of Collins et al. (1989) clarifies some of the implications for the nature of the knowledge that students acquire through a proposal for the retooling of apprenticeship methods for the teaching and learning of cognitive skills. Their studies specifically proposed the development of a new cognitive apprenticeship to teach students the thinking and problem-solving skills involved in school subjects in the domain of reading, writing, and mathematics. From their studies, they argued that those domains are foundational for learning and communication and for engaging cognitive and metacognitive processes for learning and thinking. And also those domains are well suited for teaching methods modeled on cognitive apprenticeship. In addition, they examined pedagogical practices, the structural features of traditional apprenticeship, detailing what would be required to adapt these characteristics to the teaching and learning of cognitive skills which consist of modeling, coaching and fading. In terms of coaching, it is the provision of scaffolding which is the support, in the form of reminders and help. When the learners can achieve their target skill then the master reduces (or fades his participation), providing only limited hints, refinements, and feedback to the learner (Collins et al., 1989).

In terms of traditional apprenticeship, Collins et al. (1989) explored the provision of a conceptual model to success in teaching complex skills with three related reasons which are providing advanced organisation for the learners in their first attempts to execute a complex skill; provide sense of the feedback, hints, and corrections from the master during interactive coaching sessions; support the learner to learn independently by their own performance.

According to Collins et al. (1989), in terms of cognitive apprenticeship, it emphasises two issues, which are the methods aimed initially at teaching the processes that experts use to handle complex tasks; learning through-guided-experience on cognitive and metacognitive, rather than physical, skills and

processes. Cognitive apprenticeship requires techniques to encourage the development of self-correction and monitoring skills. Thus, traditional apprenticeship focuses on teaching skills in the context of their use while cognitive apprenticeship is extending situated learning, which is learning in real situations, to diverse settings so that students learn how to apply their skills in varied contexts.

2.3.2.1.3.1 Cognitive Apprenticeship Framework

Collins et al. (1989) proposed a framework for designing a learning environment to be characteristics of ideas in learning environments in which there are four dimensions: content, methods, sequence, and sociology.

A) **Content** – there are four categories of expert knowledge (p. 477) a) Domain knowledge –in an appropriate situation- consists of conceptual, factual, and procedural knowledge; b) Heuristic strategies comprise effective techniques and approaches for accomplishing a task; c) Control strategies are the decision of how to select various possible problem-solving strategies; how to decide when to change strategies; reflection on problem-solving; monitoring, diagnostic, remedial components; and d) Learning strategies – knowledge about how to learn beyond general strategies.

B) **Methods-** for the design of teaching methods to help students acquire and integrate cognitive and metacognitive strategies. Collins et al. (1989) succeeded in using their model in reading, writing and mathematics, based on cognitive and metacognitive strategies to centre their teaching around activities to convey these explicitly to students. Their method consisted of modeling, coaching, scaffolding, articulation, reflection, and exploration. Modeling is an expert model task. Coaching is both observing students while they carry out a task and offering hints, scaffolding, feedback, modeling, reminders, and new tasks. Scaffolding is the support that a teacher provides to help/suggest to a student how to carry out a task. Articulation is any method of getting students to articulate their knowledge, reasoning, or problem-solving processes in a domain. Reflection enables students to compare their own problem-solving process with those of an expert, another student, an internal cognitive model of expertise. Exploration is a method that students use to solve problems on their own.

C) **Sequencing** – The principle of sequencing is that it allows students to build a conceptual map –a conceptual model encouraged by expert modeling.

D) **Sociology** - Apprentices learned skills in the context of their application to realistic problems, within a culture focused on and defined by expert practice.

Cognitive apprenticeship framework is useful for teacher training in the electronic learning environment. Thus, this model might be helpful to use in order to help the teacher learn to give good feedback.

There are several cognitive apprenticeship software programs available to help the learner to learn, for example, Smalltalker (Chee, 1995), and COMPANION (Hilem & Futtersack, 1994); however, the implementation of scaffolding and fading are difficult instructional methods because they require a teacher or the system to be sensitive to the specific desires and difficulties of students engaged in task performance at any particular point in time, according to Chee (1995). In other words, in his system when the student requests any help, they can click the **"I'm stuck"** button. In this case, this system uses scaffolding mode like a help system rather than adaptive scaffolding.

As a consequence the implementation of scaffolding approach is interesting, thereafter, there are a number of researchers that implement such an approach e.g. Emile (Guzdial, 1995); Ecolab (Luckin & du Boulay, 1999; Luckin et al., 2003); SE-Coach (Conati & VanLehn, 1999)). Furthermore, a scaffolding approach would be useful to cooperate with contingent tutoring and computer-support (Wood, 2001). In this manner, we decided to use a scaffolding approach to implement our context.

2.3.2.1.3.2 Situated Cognition and the Culture Learning

Situated learning theory is a precise model applied in several contexts in every situation. With regard to teaching practice, learning in a real situation could help the learners learn and use tools to better support understanding. Thus, the knowledge of learning how to use a tool could help the apprentice to practice effectively (Brown et al., 1989). They studied two examples of mathematics instruction and thus proposed the components of situated learning as conceptual knowledge of understanding in a real situation, a product of the activity, context, and culture –social interaction and collaboration in the culture of the domain –that

the knowledge is developed and used. Furthermore, they reported that indexing knowledge would be easier to understand for learners. Similarly, the TAs should learn to use tools indexed knowledge of quality feedback to help them provide feedback in authentic situations while marking assignments i.e. marking authentic assignments together with giving good feedback. Thus, these components would be useful in order to help the teacher learn to give good feedback.

2.3.2.1.3.3 Legitimate Peripheral Participation

Legitimate Peripheral Participation refers to how newcomers become integrated into a community of practice. In this manner, learners are learning in a situation. In addition, Lave & Wenger (1991) put forward ideas of apprenticeship such as learners as apprentices; teachers and computers as masters; cognitive apprenticeship, apprenticeship learning and life apprenticeship. They also studied five apprenticeships, namely Yucatec midwives, Vai and Gola tailors, naval quartermasters, meat cutters, and non-drinking alcoholics. From such studies, they offered the theoretical framework of legitimate peripheral participation and remarked that it should comprise conflictual forms of everyday practice, of motivation, and of the development of membership/identity into objects of analysis. Further, it was evident that no one was certain what the term meant, they stated that the synonym of apprenticeship is "situated learning".

In terms of apprenticeship, "it had become yet another panacea for a broad spectrum of learning-research problems, and it was in danger of becoming meaningless", according to Lave & Wenger (1991). From apprenticeship to situated learning, apprenticeship as models of effective learning in the context of a broader theoretical goal. For situated learning, in particular, situated activities resulted from differing interpretations of the concept, in terms of situated, it refers to some people's thoughts and actions located in space and time, or depends on the meaning of a social setting, according to Lave & Wenger (p.32). Situated learning refers to a transitory concept, a bridge, between a view according to which cognitive processes are primary and a view of social practice is the primary, generative phenomenon, and learning is some of its characteristics. Thus, legitimate peripheral participation is proposed as a descriptor of engagement in social practice that entails learning as an integral constituent.

The concept of regeneration is how a newcomer moves from periphery, to full participation, to supporting newcomers. In other words, the community of practice is a social organism that exists so people can enter and leave with little effect. As a consequence, a mentor's experience could support learning technology. In this case, computer-support may not only be able to provide some facility for giving good feedback, but also provide some aspects of mentoring (Lave & Wenger, 1991). This framework might be helpful to use in order to help novice teachers as newcomers learn to give good feedback.

Wenger's (1998) later work presented a theory of learning in social practice. The main idea of this work focused on communities of practice which is discussed in terms of community, social practice, meaning, and identity. He moved from describing activities in terms of legitimate peripheral participation towards examining activities in terms of the tensions between several "dualities".

He described Communities of Practice in terms of four dualities': between participation and reification; designed and emergent issues; identification and negotiability; and local and global practice. The duality between participation and reification has attracted the most interest. Importantly, he describes the relationship of these concepts as a logical one i.e. everything has both explicit and tacit knowledge; and everything also has both formal and informal process.

> *"Explicit knowledge is thus not freed from the tacit. Formal processes are not freed from the informal. In fact, in terms of meaningfulness, the opposite is more likely. To be understood meaningfully as a representation of a piece of physics knowledge, an abstract reification like $E=mc^2$ does not obviate a close connection to the physics community but, on the contrary, requires it. In general, viewed as reification, a more abstract formulation will require more intense and specific participation to remain meaningful, not less." (Wenger, 1998, p. 67).*

Wenger's theory can be applied to any conceptual framework for thinking about learning since learning usually involves a "process" of social participation.

> *"Whenever a process, course, or system is being designed, it is thus essential to involve the affected to communities of practice" (Wenger, 1998, p. 234).*

In our context, a novice TA is a member of a community of practice - a practitioner. He/She can get to learn through reflection what it means to give good feedback. He/She can also participate (a legitimate peripheral participation) in marking assignments with the help of a computer support tool. He/She can use the

computer support as a tool to practice learning to give good feedback. The tool could help him/her reflect on his/her action. With regard to reflection-in-action (Schön, 1983), this is like the TA is thinking about how to give good feedback to students while he/she is learning to give feedback in the situation of marking assignments.

2.3.2.1.3.4 Situated Learning in Adult Education

Stein (1998) defined situated learning in adults as to create the conditions in which participants will experience the complexity and ambiguity of learning in the real world. He also proposed four elements of situated learning which are

- content;
- context;
- community of practice;
- participation.

In terms of content, it consists of fact and the process of the task but emphasises high order thinking process for reflective thinking, for example, dialogue with learners, negotiates the meaning of contents, instructors provide opportunities for learners to cooperate in investigating problem situations and apply content close to their environment. For context, it is building an instructional environment sensitive to the tasks learners must complete to be successful in practice (setting for examining experience). In terms of community of practice, it regards the shaping of learning i.e. opportunity for interaction. This element comprises the joining of practice with analysis and reflection to share tacit understandings and to create shared knowledge from experiences among participants in a learning opportunity, and the body of knowledge created by an individual entering an area of inquiry. For participation, it is providing the learner with the meaning of the experience then an interchange of ideas between each learner with the material of instruction. This approach is similar to cognitive apprenticeship approach in which the learners observe mentor/expert and use their acquired knowledge. In addition, it would be useful to apply this approach to tools to teach novice teachers or TAs to give feedback from the context of the system by analyzing the knowledge of expert and requirements from the students because the things that students require are the things that the TAs often do not provide. Therefore, in our context, participants could consist of TAs, students, and experts

who can provide some aspect for designing a teaching system but the users will be the TAs directly and indirectly the students.

2.3.2.2 Suggestions from teacher

Suggestions from the teachers might be helpful in providing some aspects for designing the teaching system to train the TAs to give quality feedback according to the following sub-sections.

2.3.2.2.1 Training Strategies

In the studies of Elawar & Corno (1985) teachers provided written feedback on students' mathematic homework and were trained to give written feedback on specific errors, tell the students their weaknesses and give suggestions on how to improve. In their studies, there were three tests for training teachers

- the experiment group –whole-class treatment, provide normal practice –no training for providing specific written feedback- and marking with no comment;
- half-class treatment group, provide practice –training the teachers for giving specific written feedback- and marking with full feedback;
- 3). The other treatment group were provided practice and marking with no feedback.

These studies were for training the teachers to give written feedback. In the training of giving specific written feedback, it used a combination of lecture, demonstration, open question, and simulation exercises in which the trainer provided a four-question algorithm to ask themselves whilst reviewing students homework for them in order to think about the feedback they gave to the students, those questions were:

- What is the key error?
- What is the probable reason the student made this error?
- How can I guide the student to avoid the error in the future? and
- What did the students do well that could be noted? (p. 166).

From their results, the teachers improved a small amount in the skill of giving feedback. However, despite the good algorithm, the trainer did not provide

guide/hints to help the teachers training. In addition, this empirical study was concerned with mathematics marking for school children not for adult learners in the context of marking computer science assignments. Thereafter, Black & Wiliam (1998) reported that most teachers require instruments (tools) to improve feedback response to students. In this manner, using computer-support could help them learn/improve providing feedback.

Clarke (2000) suggested training should be improved for new teachers in the next course in order to improve giving feedback on assignments, and provided the following practical strategies:

- clear objective
- tell the learner to work according to the learning objective
- tell important weak point to learners to reduce the unknown gap
- tell the learner how to improve their current errors
- don't write too much on the bottom of students' work
- give students time to digest teacher feedback and wait to see how they improve.

These strategies are of interest to adopt in our context.

Training to give feedback might be considered as providing more detailed feedback. Detailed feedback should consist of various elements. There are three elements of feedback. Those are

- 1design goal
- evidence about present position to provide feedback
- some understanding of a way to close the gap between the two (Carnell, 2000).

In addition, there are more effective ways of presenting feedback. Cousins & Leithwood (1986) found that the qualities of the information source for presenting feedback consisted of sophistication, credibility, relevance, communication quality, content and timeliness. These would be the dimensions of feedback that TAs should consider before giving feedback to the learners.

Besides, it should prove to be an efficient approach if practitioners exchange their reflections and learning for giving feedback in order to achieve the

best practice, and successful improvement of providing feedback (Thomas et al., 2000). However, this approach may be out of our context that would allow the TAs to learn to give feedback to individuals within the system. From the importance of a tutoring strategy, feedback in an educational way has to be a constructive and open system between learning and additional editing for future learning (Kawachi, 2002). Although, there is improvement of effective learning instruction, some research has inconsistent results because many teachers have not been trained to give feedback, furthermore there is very little research in this area (Brinko, 1993). Accordingly, it is interesting to carry out research in training novice teachers to give quality feedback with computer-support, which includes aspects from experts for giving such feedback and providing levels of help to support giving quality feedback.

2.3.2.2.2 Naturalistic tutoring protocol

Considering naturalistic tutoring, Graesser et al.(1995) examined dialogue patterns using two examples of naturalistic tutoring with unskilled tutors, namely graduate students tutoring research methods to undergraduate students, and high school students tutoring algebra to 7th grade students. They found that peer students could help students to solve problems better than instructors. However, this conflicts with the belief that expert tutors should provide a better learning outcome. Graesser et al. conjectured that tutors might lack training to be experts and they desire to be practiced in domain knowledge and tutoring strategies before being experienced tutors. When those tutors complete their training, they should have a good skill level in conversational dialogue. There are a great deal of researchers that demand to discriminate dialogue pattern during tutoring to relate with learning outcome; however, their work is not in-depth qualitative analyses of tutorial interaction (Graesser et al.). For this reason, Graesser et al. investigated naturalistic tutoring protocols. These protocols were clear learning components which focused on current pedagogical theories and Intelligent Tutoring Systems. The learning components of Graesser et al. consist of:

- Active student learning;
- Sophisticated pedagogical strategies;
- Anchored learning in specific examples and cases;

- Collaborative problem solving and question answering;
- Deep explanatory reasoning;
- Convergence toward shared meanings;
- Feedback, error diagnosis, and remediation;
- Affect and motivation.

They argued there are some learning components that are underdeveloped, flawed or non-existent in normal tutoring. As a consequence, tutors may require training to implement those components by expert tutors and Intelligent Tutoring Systems therefore helping tutors increase the learning outcomes.

Graesser et al. (1995) researched the active student learning component and found that students asked questions in a tutoring setting more frequently than in a classroom setting; nevertheless, students in their studies were low in "active student learning". Therefore, students may need to be trained in asking good questions to reflect their weak knowledge. In comparison, TAs are like novice teachers, to be good tutors, they may desire to be trained in asking good questions to expert tutors to make up for their lack of knowledge in teaching. To reach "active student learning", Intelligent Tutoring Systems are required to define particular transferring control strategies to students. There are a large number of computer systems (e.g. (Woolf & McDonald, 1984; Clancey, 1987; Woolf, 1991; Graesser et al., 1999) which include the starting dialogue to prompt students to ask questions, answer questions, create examples and control students' learning environment. However, even though the studies of Graesser et al. provide good learning components, they do not give any structure for teaching unskilled tutors, particularly, on giving feedback out of the classroom.

Graesser et al. (1995) proposed the tutor controls dialogue in the form of curriculum script and a five step "dialogue frame". As follows: Tutor asks question; Student answers question; Tutor gives short feedback on the quality of the answer; Tutor and student collaboratively improve the quality of answer; Tutor assesses students' understanding of answer. Tutors use polite conversation with students to give indirect student errors. Due to the fact that it seldom developed the learning component, the analyses of Graesser et al. are in-depth with regards to the gap of potential learning mechanism that exist during normal

tutoring. Their project has been discriminated into unskilled tutoring, skilled tutoring, and Intelligent Tutoring Systems, in order to identify what skilled tutors do, how unskilled tutors fail to train effective tutors. Unskilled tutor failings can be applied to enhance Intelligent Tutoring Systems to imitate human tutors. However, these might not be enough for unskilled tutors to learn from skilled tutors. There are still other factors to learn, for example, learning environment, learning from student comments and reaction, etc. Although the solution of Graesser et al. emphasized student learning rather than teacher learning, we can still apply some components to teach teachers to teach students. At this point, critics of this position might argue that they have not structured a module of giving feedback or how to apply scaffold modeling. Subsequently the "dialogue frame" of Graesser et al. (1995) was adapted by the studies of Chi et al. (2001).

Chi et al. (2001) adapted the following five steps of the "dialogue frame" of Graesser et al. (1995) to be a "tutoring frame" into their studies; tutor asks starting question; provide a beginning answer by student; provide short feedback by tutor (confirm the answer is correct or not; scaffolding and elaborative feedback for students' answer by tutor (taking 5-10 turns); evaluation of students' understanding by tutors. The studies of Chi et al. and Graesser et al. for structuring interactive feedback provide a module of giving feedback, but at this point they have not addressed how teachers learn to give good feedback. Feedback provision for the novice teacher is very interactive and would demand adaptation to give feedback and explain students' weaknesses to reflect different constraints of the interaction.

2.3.2.2.3 Prompt/scaffolding (hint, suggest)

Providing an example of "scaffolding prompt" (Chi et al., 2001) could help teachers learn to give feedback. Nevertheless, even though the studies of Chi et al. (p.507) emphasized scaffolding to guide prompting a student rather than giving direct feedback on a student's response, it might be useful to adapt to give good feedback to the TAs to give feedback to the students. Their methods consist of:

- Open-ended (e.g. "What's going on here?" "Any thing else to say about it?" "Could you explain or put this in you own words" "What do you think?")

- Content-free (e.g. "What are you thinking about?" "What do you think?" "What does this sentence mean?" "What does this sentence tell us?")
- Deep scaffolding prompts (e.g. "Do you have any ideas/ thought on why that might be the case?" or "Could you connect what you just read to what you have read before?"

Providing a list of prompts as guidance from teachers' comments and they can practice by stimulating the learners to respond (guide them to the next step). Chi et al. (p.508) reported two steps to help teachers to give feedback to students:

- Teachers read a description on what prompting is and what it is not, that it is effective, and hypotheses for why it is effective
- Teachers read several excerpts taken from a pilot study involving tutors prompting students, to get a sense of what a prompting dialogue looks like.

These may not be enough to help TAs to learn to give feedback to students because the studies of Chi et al. scaffolded students, not teachers, but their idea may apply as to how to improve feedback by prompt/scaffolding to teachers to learn to give better feedback. Their approaches provided feedback to students even though they might give a little feedback to teachers.

Kullman (1998) observed that giving feedback to apprentice teachers consists of "non-directive behavior" (e.g. tell the apprentices non-directives such as "Why did you …"), "developmental behavior", and "collaborative behavior". He pointed out that apprentice teachers desire either a mentor or experienced teacher to "tell them what was expected of them", to "demonstrate knowledge", to "show good examples", to "criticise the bad things", and to "tell the truth in an encouraging way". They desire an expert teacher to give them an example of giving feedback. Penny et al. (1996) found that "students appeared to conflate the mentor's counseling, teaching and assessment roles…most students suggested that mentors and tutors should tell them what they should do so that 'mistakes' could be avoided in the future (p. 62), they also found that most students saw tutors demand for critical reflection…as a form of assessment (p. 67)". From a non-directive approach, Kullman reported that teachers should not ask students questions without providing guidance. This result tells us that a teaching system

should guide unskilled teachers to give feedback to students as well as extra feedback that includes more explanation of how to avoid errors.

The methods of Chi et al. (2001) have not scaffolded teachers, so when do teachers know which is the best feedback to give to students. Due to the fact that their methods focused on feedback to students in which this may concern very little feedback to teachers, we desire to know how do teachers improve? How do we know which feedback in the feedback protocol given to teachers is good/better feedback? Beyond this, we should learn how to provide quality feedback from other sources according to the following sections.

2.3.2.3 Learn from student comment

Studying student's feedback could help teachers learn how to improve providing useful feedback to students. Student comment may be positive comment or negative comment in which it may be an example of feedback for teachers to learn how to give useful feedback for each learner. As a consequence, teachers should know which type of comment might be useful to improve their feedback next time. In addition, Gibbs et al. (2003) reported the results of Assessment Experience Questionnaire (AEQ) for science courses (e.g. Physics, Chemistry, etc) from two universities (University A & University B) in which teachers from the University A give considerable feedback to students for a large assignment while at University B, teachers give little feedback to a small assignment. In order to provide a large number of students with quick feedback, University A has to use a number of resources. The results of the questionnaires from the majority of students in University A indicated that the students can achieve benefit from feedback in order to improve their assignment; some students from University B seldom received feedback or recommendations when they misunderstood or obtained late feedback that was useful for the next assignment; some students complained that sometime feedback did not help them learn better or improve their learning or feedback did not give information how to improve their learning (Gibbs et al., 2003). These studies showed that we should pay attention to detailed feedback that includes an explanation of how to avoid error and improve students learning.

With regard to providing teachers with a good feedback example, they should be given instances of positive comments from students that can be studied to discover how each teacher as a learner can improve his/her learning. For instance, Saitio (1994) found that students preferred feedback from authentic teachers rather than non-teacher feedback. Jackson (1995) found that students preferred to see feedback from teachers. In addition, students preferred that teachers told them how to use feedback in order to improve their meta-cognitive control (Sadler, 1998).

Thus, these results show that quality feedback should be concerned with more explanation or guidance to the students' misconception as well as human teachers should participate in providing feedback on automated marking feedback tools. In addition, the results of students comments as mentioned above could imply to adapted feedback mechanism.

2.3.2.4 Learn from teacher comment

Learning by giving feedback between teachers (Gibbs, 1978)), (Muda, 2000)), (Back, 1999)), (Zohar, 2000)), (Brown, 1973) could help teachers to give better feedback. Providing feedback information to the teachers might be helpful or not useful and may come from students thinking or praise given by other teachers. Feedback from students as mentioned above is quite useful while feedback from teachers to teachers may be via face-to-face, or mediated technology as per the following sub-sections.

2.3.2.4.1 Face-to-Face

Provision of face-to-face feedback is the traditional way of giving feedback among teachers. It might be providing feedback between peer teachers or by the result of interviewing, as conducted by Gibbs & Simpson (2003). They found several results from interviewing teachers in order to improve teachers' ability at giving feedback and changes to appropriate assignments in order to give appropriate feedback to students. Defining limited assignment, for example, give students an assignment by reading the question of the assignment then returning to study which can help students to complete their assignment rather than read all lecture notes then do the assignment; some students did not read teacher feedback

from assignments because it is not useful for the next assignment; some students read feedback from assignments in order to prepare for an examination; however, some teachers do not return assignments back to students; some students pretended to know and understand the feedback given to them. Therefore, the feedback should be helpful for both students and teachers and in giving each assignment it should be a continuous practice, for instance, students should understand the first assignment before doing the second assignment. According to such results, it could help us in designing the TAs system.

2.3.2.4.2 Mediated Technology

The role of computers for providing feedback influences not only the learners, but also the teachers. For example, asynchronous communication (Conferencing On the Web: COW (Bonk et al., 2001)) for pre-service teachers were employed between three groups of people i.e. between pre-service teachers and pre-service teachers; between pre-service teachers and mentors; and between pre-service teachers and instructors. COW was employed among these people in order to discuss some sort of problematic situation that pre-service teachers were confronted with, and then help the pre-service teachers solve such a problem. In this study, peer feedback was very conversational and opinionated while feedback from instructors/mentors was emphasised by high level questions, as well as providing examples and case specific feedback. Even though some results of this study showed effectiveness for Conferencing On the Web (COW) for the pre-service teachers in the discussion, some results showed the problems with tools for structuring case feedback. In such cases, the pre-service teachers revealed that they were unhappy with the lack of model types and quantity of case feedback. However in the case of this study it is merely training the pre-service teachers to use COW -not training them to provide quality feedback, this might suggest some idea of how a TA system interacts with the TAs alongside interactive-dialogue.

Developing giving feedback to teachers may use interactive feedback (system-TAs) in our context. Thus, employing these approaches to build computer-support could not only enable some facilities of giving good feedback, but also provide aspects of mentoring.

2.4 How do the computer systems help teachers provide feedback?

Employing computer-support to help the teachers provide feedback on students' assignment could be useful to promote students learning e.g. giving feedback in science marking systems (Section 2.4.1), in students essay writing marking systems (Section 2.4.2), in mathematics marking systems (Section 2.4.3), in computer science marking systems (Section 2.4.4) in which we will explore such systems in the aspect of how to help teachers provide quality feedback to students in the following sub-section.

2.4.1 Science marking systems

Electronic Feedback is software developed by Denton (2003) for giving consistent feedback to students' lab report and essay writing. This system helps the marker prepare feedback messages such as general comments by the system; providing grade comments; and standard comments by the marker prepared in advance before giving feedback including the student's name. Even if electronic marking assistants might be useful to improve giving feedback to the learner, Denton argued that it is not part of a teacher's routine. He found that if we define a feedback statement before marking the assignment, it will save time marking students' work and tutors can mark quickly. Further, the results of his students who received feedback via e-mail stated it was an efficient approach to achieve their actions on the assignment (e.g. "It is a helpful method of marking as it enables you to see how and why mistakes were made..." and "It offers a more in-depth description of how you have gone wrong" (Denton, 2003)). In addition, the results of employing his software by the markers showed positive responses including that they can return more feedback of a higher quality, and in a shorter period of time. Nevertheless, even though his system could help the learner to improve their learning, it did not help novice teachers; or TAs learn to give quality feedback.

2.4.2 Essay marking systems

Beyond the software of Denton (2003) as mentioned earlier, which can help the marker provide feedback to the learner, the literature of Dennis, Mills, Smith, & Tucker (2002), reported that the existing essay marking systems that they can assess via the web such as PEG (Page et al., 1997), E-Rater (Burstein & Marcu, 2000), Betsy(Rudner, 2002)) are extremely unreliable to use in students work alone, but they are useful to assist the teachers marking essays by error analysis of the students essay. These systems still demand a human marker to work alongside the system to assess students' tasks. Some essay marking systems are used as commercial software tools, for example Intelligent Essay Assessor: IEA (Chung & O'Neil, 1997; Landauer et al., 1998) for students in essay writing training.

In addition, e-TMA (Electronic Tutor Marked Assignment) for marking assignments, developed by the Open University, although not an automatic marking essay (Thomas, 1998), can add comments by itself and also can add crossing-out/modifying text (track change of Ms.word). Thereafter Moreale et al. (2002) investigated the parameters associated with marking assignments by postgraduate tutors via e-TMA and found : time, level of assignments and, a set of surface metrics. Besides, they proposed that automatic marking essays are often suited for postgraduate students because they can provide topic-related content such as "Does it match the question?" In this study, although it has provided some benchmark metrics to start to construct a monitoring system that includes both readability metrics and content heuristics from both tutors and students, problems that arise include

- Teachers/tutors have to mark and give feedback by themselves with no guide.
- In e-TMA tools, marking via a screen, tutors can open/close the e-TMA file if they have not finished marking, but the system will not respond to teacher's marking.
- No feedback to tutors for giving feedback to students (to help tutors learn to give good feedback).

Furthermore, the results of giving too many comments did not help the students obtain a high score. In sum, in spite of the system assisting the teachers

to give feedback to students, it did not provide any support for training the novice teachers to learn to provide good feedback.

Later, Whitelock et al. (2003) designed feedback model for tutors' written feedback on the students' essay assignments for analysing tutors' comments via e-TMA. Their model is based on Bales' (1950)'s 'interactional categories'. Bales explicitly introduces a 'socio emotive' role. Whitelock et al. are unlike many other previous researchers in this area. They wanted a scheme which included this aspect. In Whitelock et al.'s model, tutors maintain students' motivation by providing feedback with a 'socio-emotive' element and 'task-oriented' contribution. Then the system categorises the tutor feedback (Whitelock et al., 2003). Whitelock and colleagues found that the tutors stimulated the student's next response by using questions and employed Bales' categories to suggest the problems in some part of the students' essays. In addition, their finding suggested that the system could help the tutor give appropriate feedback to students. Although this model was not used to help tutors provide quality feedback to students, the analysis of the tutors' comments could be helpful to apply in training TAs to learn to give good feedback.

2.4.3 Mathematics marking systems

There are several mathematics marking systems, e.g. Online Exercises System, this system by Bryc and Pelikan (1999) can help teachers provide hints together with the question of the assignment, but does not inform the learner what was right or what was wrong. Furthermore it does not provide any help for the markers in giving good feedback; For the WebTester and the Linear Algebra WebNotes (Sapir, 1999), this system can explain errors and report what was right and what was wrong; however, it did not provide any support for training the novice teacher to learn to provide good feedback.

2.4.4 Computer Science marking systems

Automatic marking systems in computer science are frequently designed to assess students programming assignments, for example SPROUT, Ceilidh, BOSS, and CourseMaster (see Chapter 4). Furthermore, CourseMaster has the property of

diagram assessment (e.g. circuit design, software design) in which those systems can help the teacher with marking. In order to describe this, we will explore such systems in the next chapter.

2.4.5 Summary: How do computer systems help teachers provide feedback?

Despite employing either semi-automated or automated marking systems, these systems do not provided any aspects to train the novice teachers to give quality feedback. In addition, it is interesting to research how to train novice teachers to give good feedback in computer science, especially in program marking, because even though a compiler can provide some warning or error messages to students, it is not easy to understand for novice students. Even tutors who have experience in programming, still have little knowledge of communication skills to explain the errors. In this manner, we will explore such aspects in the following chapter and have also addressed this problem in the methodology in Chapter 1.

2.5 Summary

To conclude, how can we develop a system to help the TAs learn or practice giving feedback so that they can improve giving better/quality feedback skills to students in the future? There is little research which analyses or classifies the types of feedback as well as retrieving appropriate feedback in the issue of Artificial Intelligence (AI) in Education such as expert system techniques, underpinning educational psychology for either TAs or novice teachers to help them make decision about choosing suitable feedback for the learner. So far, to our knowledge, automated assessment rarely helps them to learn (Dennis et al., 2002). In other words, there are a lack of tools, if any, which provide a good experience in giving quality feedback especially with regards to marking programming assignments. In addition, in our context, we emphasize assessment which tutors provide feedback on students' practice assignments and our aim is to help teachers and learners to develop their own knowledge. In other words, our thesis scaffolds for new teachers or teaching assistants more than learners. As a consequence, we hypothesise that a scaffolding system could help the TAs learn

to give better quality feedback, a scaffolding system could help the TAs reflect/rethink their skills of providing feedback, and fading of scaffolding system could allow the TAs learn alone without any support. However, we cannot guarantee that TAs will be satisfied with the system, as this depends on "people behavior" (Norman & Draper, 1986). In fact, it is difficult to implement the scaffolding approach according to the empirical study of using computer-support for adult learning by using cognitive apprenticeship approach (Chee, 1995). Thus, how can we train TAs to give quality feedback?

In this chapter, we investigate how people learn to provide quality feedback. In order to manifest how we can design quality feedback adopted in our system, in the next chapter focuses on how to design feedback to employ in our context.

Chapter 3

Feedback Design for quality of the feedback

3.1 Introduction

The previous chapter discussed the analyses of student's weaknesses, in this chapter, we explore the feedback design used to underpin the scaffolding system for help in providing quality feedback to the TA, and less directly to the learner.

In general, feedback is one of the potential components in the learning process (Dick & Carey, 1990). Most educators also know that learning will be more powerful when it is reinforced by suitable instruction and quality feedback (Brown & Knight, 1994; Moursund, 2002). This brings us to the meaning of feedback. Feedback has been defined as any information that follows a response as both motivating the learner to try to do better; and, as providing the knowledge for learners so that they can correct or improve their answers (Skinner, 1958; Sales, 1993; Brown & Knight, 1994). It can also be any messages that the learner gets in response to a correct answer for a question (Buscemi, 2003). In this thesis, the meaning of feedback in education is "Providing scaffolding for Teaching Assistants (TAs) or novice teachers to improve the quality of the feedback in lifelong learning" TAs are like adult learners who have self-directed learning abilities (Knowles, 1988; Kerka, 1999). What knowledge of giving feedback do the TAs need to learn and what kind of feedback should they provide to students? In this thesis, general research about feedback is applied to the issue of how to provide feedback during assignment marking, which will be explored throughout the rest of the chapter.

3.2 Learner's motivation

Kulhavy & Wager (1993) claimed that when learners produce an error, response feedback allows the error to be corrected. They argued that the result of several studies conducted between the 1960's and 1993 suggest the general finding that

some feedback is better than no feedback. In addition, although feedback is a part of the instructional process in the classroom, it is also used in computer-based instructional programs (e.g. CAI (Computer Assisted Instruction)). Various researchers such as Ross & Morrison (1993) have categorized feedback into several types (e.g. KOR (Knowledge Of Response), KCR (Knowledge of Correct Response), AUC (Answer Until Correct)) that not only can help the learner to correct errors, but it can also reinforce motivation. For this reason, a large number of studies try to combine motivation and knowledge about the student's answer into programmed instruction. With regard to the system of program instruction, as can be seen from Figure 3.1, KOR feedback explains to the learner that his or her answers are correct or incorrect but it does not describe the correct response. In this process students will respond only once. According to Figure 3.2, KCR feedback is KOR with the correct response added. In this process it responds to students only once. Besides, Hancock, Stock, & Kulhavy (1992) reported that feedback is more effective when it restates the correct answer, instead of simply verifying ("right" or "wrong") the learner's initial response". Furthermore, AUC feedback as shown in Figure 3.3 is KOR with opportunities added for the learner to continue selecting answer choices until they get the correct answer. This process allows responses from the learner for several rounds, depending on the system. If the students are unable to respond with the right answer, they might not have the opportunity to learn the right answer; thus the quality system which employs AUC feedback should have a finite round of looped responses to correct the student answers. In addition, Elaborative feedback is extra feedback beyond KOR, KCR and AUC such as rewriting the correct answer or adding text extracted from one or more sources; however, from the research of Ross & Morrison (1993), the most expensive kind of response is to provide elaborative feedback. Nevertheless, if there is an appropriate pattern for giving elaborative feedback, it can help the teachers to provide such feedback easily.

Hudspeth (1993) noted that elaborative feedback is progressively more detailed, specific, and complex ideas that can then be acquired more easily as derivations or elaborations of the more general content; therefore, employing detail to a sequence of content could provide the learner with a progression of anchoring knowledge that subsumes, integrates, and organizes the more detailed

or complex knowledge. Even though these are categories of feedback derived from a study of CAI, they point out that many software designers and developers do not seem to devote much attention to the role of feedback. In spite of the difficulties in developing a semi-autonomous system, it would be possible to establish an adaptive system to help teachers provide students with feedback - "The non modelers believed that it was impossible for computer models to be extensive enough to provide the adaptive feedback required", according to Lajoie (2000). Research related to CAI feedback includes evidence that the learner's name may support the motivation of the learners (Sales, 1993; Buscemi, 2003; Denton, 2003). For this reason, feedback is an extremely important part of teaching and learning. Not only CAI researchers, but other researchers also found, from their researches, that students need to be able to access feedback to be sure that they are working along the right lines (Reushles et al., 1999; Gibbs & Simpson, 2003b). In this thesis, considering feedback provision; therefore, it would motivate the learners (both TAs and students when TAs are the learners who learn to give quality feedback while the students are the learners who are given feedback by the TAs) to learn depending on several factors from many researches, according to the following.

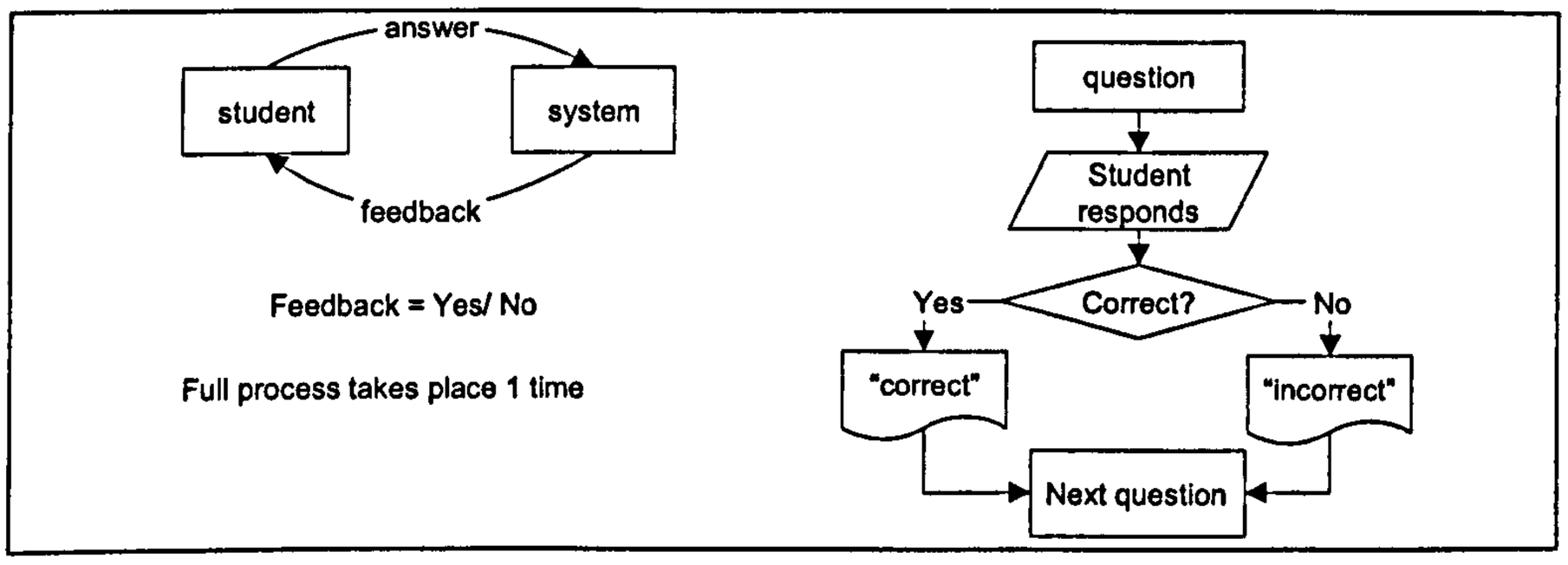


Figure 3.1 KOR diagram, slightly adapted from the original figure of the procedure used in three types of feedback (Ross & Morrison, 1993)

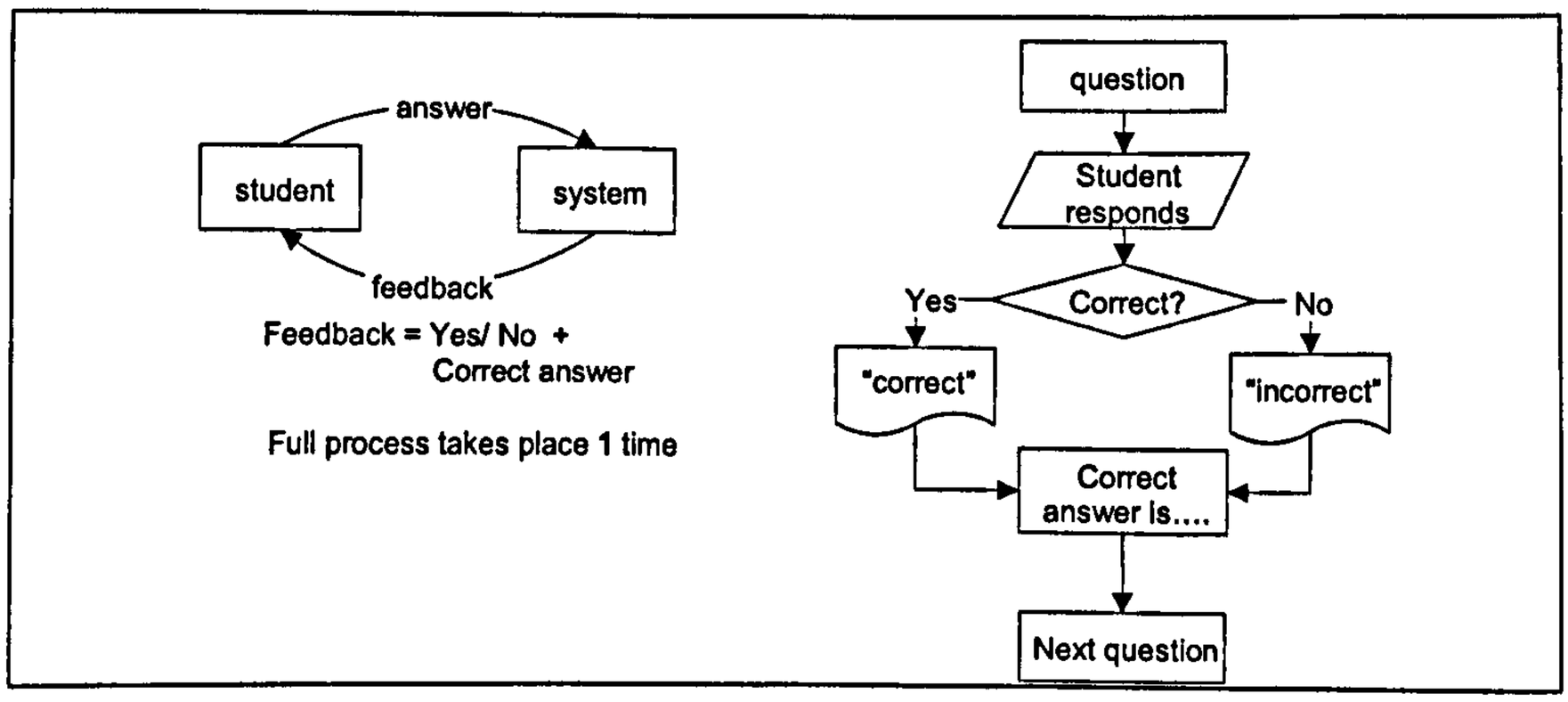


Figure 3.2 KCR diagram, slightly adapted from the original figure of the procedure used in three types of feedback (Ross & Morrison, 1993)

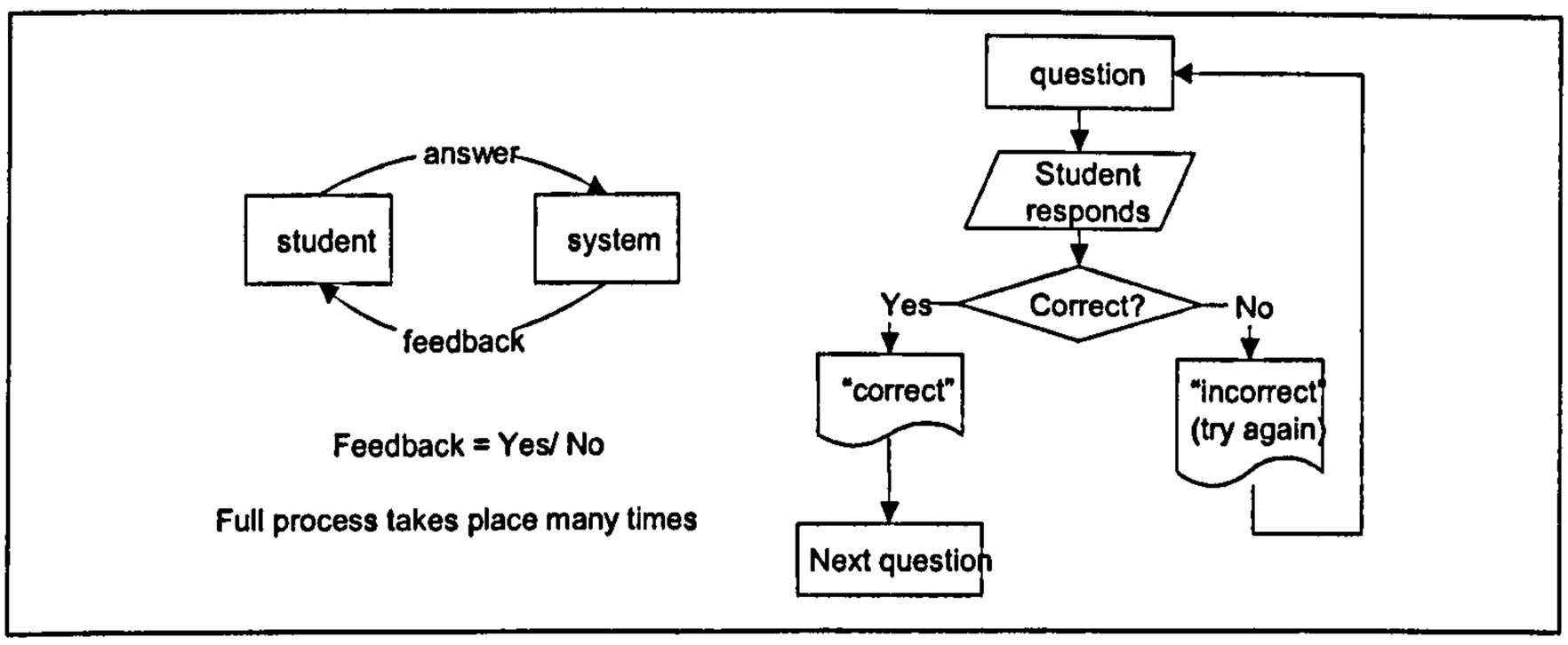


Figure 3.3 AUC diagram, slightly adapted from the original figure of the procedure used in three types of feedback (Ross & Morrison, 1993)

3.3 Individual Difference

Not only can feedback be provided by a human tutor, but it may also be provided from other sources. Each learner has individual differences such as skill level, and learning style. With regards to learning style from different sources of feedback, Draper (1999) proposed that there are three necessary categories of feasible source of feedback:

- the learner by themselves;
- the environment, for example, the compiler provides an error message;
- the human tutor.

Although there are three sources of feedback, he insisted that feedback from the learner themselves is inadequate to support their learning. As mentioned earlier, we, therefore, contend that a human tutor can contribute more to the learners than

the environment alone i.e. in providing quality feedback to an individual we should not only consider learners skill level, but also help from the tutor alongside the learning environment.

There are two separate sources of feedback: intrinsic and extrinsic. "Intrinsic" feedback is that which comes as a natural result of the action; the feedback is intrinsic to the action, while "extrinsic" feedback, does not occur within the situation but as an external comment on it: right or wrong, approval or disapproval, according to Laurillard (2002). With respect to andragogy (Knowles, 1990), the theory of adult learning, adult learners need both intrinsic and extrinsic feedback. However, Laurillard specified only feedback from teaching, which did not include feedback that results from an assignment. From providing feedback, nevertheless, we should consider the level of cognition of the learners when the teachers provide them with feedback. In this case, Heift (1998) distinguished categories of feedback corresponding to three learning levels: expert, intermediate, and novice. Firstly, expert learners need general feedback. Secondly, intermediate learners need precise location and type of error so the teacher should refer to the exact source of error. Finally, the most detailed feedback message is most suitable for the novice learner. Moreover, she considered that feedback also depends on the students' previous performance history. If we consider feedback by learning level and intrinsic and extrinsic, most expert learners have high intrinsic feedback, while novices need more external feedback. This is supported by the literature of Chai (2003) in which he reports that feedback does not support all learners because it can depend on learner behaviour. In particular, feedback relied on learners. From the experiment of Lhyle & Kulhavy (1987), learner groups who received different feedback more than one time could detect an error better than groups that received the same feedback. Regarding quality feedback, therefore, it should include intrinsic and extrinsic feedback and the level of each learner. Also, we believe that effective feedback should provide for the learner in different ways and more than one time.

3.4 Quality of feedback

In general, any tutors can provide feedback to the learner; however, how can we

know that it is quality feedback? Quality feedback should consist of content and efficiency. First of all, we consider the quality of detail in the content. From the study of Draper (1999), he pointed out that there are three types of quality feedback:

- explain to the learner what was "right";
- explain to the learner what was "wrong";
- explain to the learner how to right the wrongs without "wronging the rights".

In other words, the content is a dimension of feedback.

Due to different kinds of perception, teachers should provide different levels of feedback to the learners. Draper (1999) proposed that different kinds of feedback should be divided into five levels:

1) give information about the outcome, for example, right or wrong, success or failure;

2a) ask the students what the result was;

2b) ask the students what the right answer was;

3a) give the students the right answer. It should be noted that the teachers do not want to know if students will understand or not;

3b) analysis of which section of input was wrong;

4a) tell the students what would make the answer right;

4b) tell the students which method they used;

5) describe what's wrong about the student's output.

His classification from one to five can help the learners from low level one to high level five – i.e. he sees the different levels of student attainment as being closely associated with the kinds of feedback they need. This means that feedback at level one provides less support for the students than feedback at level five because the teachers provide more explanation about the students' mistake on level five. In order to help the learners improve their learning, this entails a level of help and feedback that can be adaptive in the following.

3.4.1 Contingent help and adaptive feedback

Generally, learners need help after making errors rather than iterate their mistakes. For this reason, it is necessary for tutors to provide the learners with contingent help, scaffolding them to achieve their Zone of Proximal Development (ZPD) (Vygotsky, 1978) as well as in order to help them solve the problems initially then when they gain more knowledge to manage the problem and finally fading through reducing the support (e.g. Ecolab (Luckin & du Boulay, 1999; Luckin et al., 2003)). For example, teachers could give students feedback and tell them to find the answers from a new question that is similar to the previous question. Learners then may not require further help because they may not make any errors (Wood et al., 1999).

In the Development of Contingent Tutoring Systems (Wood et al., 1999), the level of help is classified into 5 levels of hints:

1) The problem, and encouragement to solve it. Feedback comes here too.
2) Quick vague cue.
3) Less vague cue. If materials are involved, select appropriate ones.
4) Almost the answer or materials need only a final push.
5) The answer or demonstration of solution.

Providing contingent feedback help is similar to adaptive feedback. To put it simply, feedback can also depend on factors related to the student's state. This introduces the idea of adaptive feedback in personalizing the feedback and will include various factors. Various researches have found that the efficiency of quality feedback should relate to adaptive feedback which depends on various variables (e.g. achievement, motivation, and attitude). Sales (1993) reported the achievement of efficiency of feedback that it can achieve extended learning from differentiation error type and the content of the lesson during presentation or incorrect response. While research about motivation depends on a large number of variables, e.g. attitude, effort, time on task and achievement, which are all related. However, further results from other research in this area are still required. The last one is attribution which should define learner's perception and performance on a task together with a feedback message to specify the learner's success and his/her effort. For example, feedback should include more elaborative information which

tells the learner why her/his answer is incorrect and the possible ways to solve problems (Sales, 1993). By that means, providing effective contingent help and adaptive feedback can be done by considering each learner's achievement, time on task, effort, and attitude. This method could be useful in a design learning environment by helping TAs to give quality feedback to students.

3.4.2 Positive and Negative Feedback

Feedback should be provided in a positive way. Foote (1999) regarded positive ways as positive ability e.g. "You are very good in math"; positive effort e.g. "You have worked hard"; positive conduct e.g. "Look how neatly Debbie's paper is"; positive in general e.g. "That's very good". Providing quality feedback is extremely crucial for the learner, and as can be seen, general Intelligent Tutoring System (ITS) could not provide feedback to the learners as well as a human programming tutor (Draper, 1999). Teachers may, nevertheless, use software tools to help them make decisions before they give the students feedback. Thus, the students should receive quality feedback. However, the quality of feedback should relate to the time spent on providing the feedback. We will consider the timing of feedback in the next subsection.

Even though from the perspective of Draper (1999), there are five types of feedback, they are discriminated clearly between positive and negative feedback, he also alleged that positive feedback is more useful than negative feedback as it provides a sense of power and hope for the learners; thus, positive feedback may improve the quality of feedback. Furthermore, some teachers believe that both negative and positive feedback can motivate the learners to improve their abilities. Therefore, research is needed to verify this notion. In addition, Foote (1999) pointed out that feedback can be classified into eight types: Positive Ability, Positive Effort, Positive Conduct, Positive in General, Negative Ability, Negative Effort, Negative Conduct, and Negative in General, in which the feedback from her research is correlated between positive and negative feedback. For this reason, we assert that positive feedback should be more encouraging to the learners. In addition to positive feedback, quality feedback should support the learners through explanation or by encouraging them to find the reasoning that supports

their answers. Providing quality feedback on students programming assignments, consequently, should consider both positive and types of quality feedback.

3.5 Timing of feedback

Dempsey & Wager (1988) classified timing of feedback into immediate and delayed. In terms of immediate feedback, it is the provision of informative corrective feedback to a learner or examinee as quickly as the computer hardware and software will allow during instruction or testing. Types of immediate feedback are:

- item-by-item;
- learner-controlled;
- logical content break;
- end-of module;
- break by learner;
- time-controlled (end of the session).

Delayed feedback is the provision of informative, corrective feedback to a learner or examinee after a special programming delay interval during instruction or testing. Types of delayed feedback are:

- item-by item;
- logical content break;
- less than one hour (end-of-session);
- 1-24 hours (end of session);
- 1-7 days;
- extended delay;
- before the next session

In addition, Draper (1999) pointed out that there are two kinds of feedback timing as mentioned earlier that are the same. He reported that delayed feedback generally provides more benefits and that this depends on the kind of knowledge (declarative or procedural) being delayed via the feedback (type of feedback e.g. notification or elaborative), the type of error (e.g. critical error or non-critical error) to which the feedback is a response to the current skill level of the learner. Draper (1999) reported that with procedural knowledge, elaborative feedback,

non-critical error and low skill level, delayed feedback will be more efficient than immediate feedback. It can therefore be argued that immediate feedback may always be inappropriate for learners. However, Cook, Burnett, & Boom (1997) argued that although immediate feedback did not particularly help users to debug common errors, it can support users in debugging certain situations.

Huitt (1994) reported that the result of several researches into the uses of computers in giving feedback indicates that elaborated or delayed feedback is better than instant or immediate feedback in terms of the achievements made by learners. Despite the findings of Huitt most computer-based instruction is expected to be in the form of immediate feedback from many web course tools, these tools provide a reason why the student's answer is correct or incorrect (Sugrue, 2000) they do not however explain how to correct errors or how to avoid repeating the errors. Although the instructor can give feedback to each student, s/he should have much experience in providing quality feedback to each learner. In addition, there is little literature which classifies feedback and informs the feedback to individual students. To give individual feedback directly to students either by a program or a type of software is not easy. In such cases, most intelligent tutoring systems still inform feedback messages via human tutor (Sugrue, 2000). Thus, there is a strong argument suggesting that quality feedback should involve delayed response to learner errors. For example, if we give students immediate feedback, they cannot remember all their errors. For this reason, they may be unable to improve their incorrect answer in situations in which there is limited time. Besides, generally, immediate feedback may be unable to report certain errors such as warning-errors that involve marking programming language assignment directly to the learner.

Steinberg (1991) reported that providing immediate feedback might be depended on the learning situation. His empirical results reported that giving immediate feedback is suitable for a practice situation or an experimental situation. In comparison, delayed feedback is suited for testing situations. In terms of delayed feedback, it can help learners in self error discovery and to find or solve solutions by back tracking to the previous point, according to Steinberg.

As a result, in our context, we propose that providing delayed feedback is suitable for the learners to improve their learning while providing immediate

feedback is suited for the novice teachers who use a scaffolding system to learn to give feedback. In order to relay quality feedback to the learner, we should consider the quantity of feedback in the following.

3.6 Quantity of feedback

In relation to teaching programming languages, as with other subjects, it is necessary to provide feedback to the learners. It is not only human tutors, but also intelligent tutoring systems, that should provide clear final feedback after finishing some phase of teaching or tutoring. In other words, it is quantity of feedback or final recommendation. Heift (1998) found that the final recommendation for providing feedback is necessary for the learner. Those are

- feedback should be precise;
- it is not useful to provide more than one error message at a time;
- describe a specific error briefly.

The learner can easily understand describing the clarity of feedback in brief. Furthermore, any brief presentation can be clear to comprehend to the listener i.e. we should present a summary of the student's errors if the student performs the same types of errors and informs the students that there are more errors like this. This could help the students remember their errors and help them not to repeat themselves next time. When considering the submitting of student's programming assignment, there are situations where the response of the tutor may not be directly answering "correct" or "incorrect"; however, they may respond to the learners with "acceptable" or "unacceptable". The correctness here means it depends on right or wrong; acceptable or unacceptable, etc depending on the tutors' decision. This could happen for several rounds in order to receive the feedback. The amount of feedback could reduce in the later rounds and the students could make fewer errors after obtaining the feedback. In other words, tutors should give, for example, an opportunity to the learners to resubmit their assignments via their asking questions for hint/guide until the due date or final submission such as the conversations below:

Student: Is that good enough?

Teacher: Your assignment does not include comments on each control structure.

Or

This might be made better if you think about a meaningful variable.

Or

I think.... / I felt .... / You should have ...

Or

Don't you think it would have been better to...? / Why didn't you...?

Or didn't you realise/notice that...?

Student: Is that good enough to submit?

Teacher: Right, there are some variables which you haven't used in your program.

Or

You don't understand something or something in your program doesn't work for you.

Student: Is that acceptable?

Teacher: Well, your program is written in the wrong style. Can you improve it?

System design in our context can be achieved by allowing the TAs to ask the students questions on their weaknesses. Then students can answer their questions by resubmission of their revised assignment. However, questions from students are not in our context.

Jang, Kim, & Baek (2001) in their study of the design of feedback content proposed that effective feedback will be successful when it consists of verification (telling right/wrong) and elaboration (telling hint/detail and answering until correct). Jang, et al. (2001)) reported that many types of feedback depend on

- the method of presentation,
- the type of presentation,
- the time of presentation, and
- the amount of information presented.

In terms of elaborative feedback, it is good for less intelligent students (e.g. providing photograph, picture, diagram, animation, teacher's explanation via audio files). In terms of verification feedback and KCR feedback, this is suited for more intelligent students. Furthermore, they suggested a level of feedback which can be associated with the level of feedback of Draper (1999) according to our

analysis in Table 3.1. This could be useful for designing giving feedback to the TA via a scaffolding system. The feedback to be given from the TA to students will be presented in the next chapter.

Table 3.1 Analysis of the Level of Feedback Content

Feedback Content	Verification	Elaboration	Comment
No feedback (Jang et al., 2001)	-	-	Neither verification nor elaboration (give only score)
KOR (Ross & Morrison, 1993; Jang et al., 2001)	√	-	Response correct/incorrect depends on student's answer. This level is associated with feedback level 1- information of outcome- of Draper (1999)
AUC(Ross & Morrison, 1993; Jang et al., 2001)	√	-	Remains the same test until student gives the correct answer. This level is associated with feedback level 2a of Draper (1999)
KCR (Ross & Morrison, 1993; Jang et al., 2001)	√ (individual item verification + correct answer)	-	*Ask what was right* (from level 2b of Draper (1999)
Describe the right answer *(Draper, 1999)*	√	√	Verify correct with the correct answer, and Verify incorrect with the correct answer. This level is associated with feedback level 3a of Draper (1999)
***TC*: Topic Contingent** (Ross & Morrison, 1993; Jang et al., 2001)	√ (item verification)	√ (General elaboration which depends on the topic)	Depends on the topic. If a student answers incorrect then he should find topic (read) that contains correct information or add more details to the system to find the correct answer System provides answer but it depends on the student to seek such information as elaboration available (e.g. link to further reading in CourseMaster) This level is associated with feedback level 4a of Draper (1999)
***RC:* Response Contingent** (Ross & Morrison, 1993; Jang et al., 2001)	√	√ (item-specific elaboration)	Give the reason why the student's answer was either correct or incorrect. This level is associated with feedback level 3b of Draper (1999)
***BR*: Bug-Related** (Ross & Morrison, 1993; Jang et al., 2001)	√	√ (address specific errors)	- Elaboration depends on bug-libraries /rule sets (common learner error) - No correct response but helps students to find errors for self correction -This level is associated with feedback level 4a of Draper (1999)

Feedback Content	Verification	Elaboration	Comment
***AI*: Attribute Isolation** (Ross & Morrison, 1993; Jang et al., 2001)	√ (item verification)	√ (highlight the central attributes of the target concept)	Focus learner on key components of the concept to improve general understanding of the phenomenon (give the main key for students' understanding) This level is associated with feedback level 5 of Draper (1999)

The level of feedback as mentioned in Table 3.1 could be useful for training the TAs to provide feedback to the learners. Furthermore, the feedback level of RC, BR, AI could be useful as a model of providing feedback to the students. Beyond this, quantity feedback should be clear feedback that is easy to understand for the learner. In addition, we should add warning errors relating the clearness of feedback provided to the learner when marking a programming assignment.

Wager & Mory (1993) observed that different feedback depended on different types of learning and also found that feedback is always related to a response generated by a question (p.70). They also recommended using question and feedback associated with the stages of information processing (p.71) as can be seen from Table 3.2. This suggestion inspires ideas to design the TA system to teach the TAs to give quality feedback. Accordingly, in order to inform feedback to the learner clearly, we should have patterns of interaction in the following.

Table 3.2 Recommendations for the use of questions and feedback related to stages of information processing (Wager & Mory, 1993)

Function of the question	Function and type of feedback
1. Gain Attention	1. Arousal, create cognitive dissonance, Open-ended questions with no feedback.
2. Inform the learner of the objective.	2. Create an expectancy for performance. Rhetorical questions and didactic answers could be used to inform the student of the objectives.
3. Stimulate the recall of prerequisite learning	3. Bring related knowledge into short-term memory and confirm present knowledge. Questions eliciting analogies with right-wrong or conditional feedback could be used to test the understanding of prerequisites. Pretests may serve the same function (with or without feedback).
4. Present the stimulus material	4. Socratic dialogue to have the student deduce what information is needed. Feedback takes the form of confirmation of responses, and probing questions to guide inquiry.
5. Provide learning guidance	5. Questions provided for modeling component parts of the skill being learned. Feedback should show correct analysis by the student.
6. Elicit the performance	6. Questions recall learned skill or components of the learned skill to test for misunderstanding. Feedback should be remedial, directed at the misunderstanding if possible.

Function of the question	Function and type of feedback
7. Provide feedback	7. Feedback should be chosen to fit the purpose that the question is serving in the instructional process. It is possible that the type of feedback should vary with the learner's performance and confidence. Give knowledge of correctness and remediation for incorrect answers.
8. Assess performance	8. The purpose is to inform the student of progress toward mastery. Feedback should inform the student of the adequacy of his or her performance.
9. Enhance retention and transfer	9. Provide for spaced practice of the newly learned skill in an authentic situation. Immediate feedback as to correctness would seem most appropriate.

3.7 Pattern of interaction

The model of interaction of feedback would be achieved when it consists of many components. In this thesis, we consider the following three components

- the level of detail;
- setting;
- sequencing and filtering.

3.7.1 The level of detail

Feedback not only supports students to point out specific problems with their work, but also fosters motivation and can assist the teacher to take care of the reaction of the student. Providing feedback to the learner may be specific to the problem or general to the kind of problem, and so on, according to the categorization of Rehwinkel (2003). From the level of feedback detail, it also relates to the research of Cook et al. (1997) in which there are three factors that affect feedback which are: type of problem (specific/general), type of user (individual/group/class, low/high achievement), and type of bug (important/common error). Considering the error type, we also classify type of bug into three types that are design, implementation and style problem (see Chapter 4). In addition to the effect of factors of feedback, it influences learners to improve their learning. Accordingly, considering the level of feedback in Table 3.1, it could be useful if applied in a training system for TAs to give quality feedback and also for the learners to improve their learning

3.7.2 Setting

Setting should consider if we would provide feedback to individuals or a group of learners. In addition to setting the class of the learner, we should also contemplate corrective or evaluative feedback (Rehwinkel, 2003). Figure 3.4 and Figure 3.5, show how the diagram of corrective and evaluative feedback combine with the feedback giving level (Draper, 1999). Both diagrams can be applied to training the TA to give quality feedback by scaffolding system and can also be applied in giving feedback to students. Figure 3.4 is adapted from the original KOR and KCR figure of the procedure used in three types of feedback (Ross & Morrison, 1993). This figure consists of providing timing i.e. immediate feedback and gives the level of detail for quality feedback i.e. in the case of incorrect error, giving information about the outcome followed by asking the learner what the result was. Figure 3.5 is adapted from the original AUC figure of the procedure used in three types of feedback (Ross & Morrison, 1993). This figure consists of providing timing i.e. immediate feedback and gives the level of detail for quality of feedback i.e. in the case of an incorrect answer, giving the learners the right answer will be followed by providing information about describing what's wrong about the learner's output.

3.7.3 Sequencing and filtering

Most feedback that is received by learners is either intended to help the learner improve or correct an error. Some feedback may be given for guidance to the learners to find the answers from the previous lessons. This should help the learners to develop their skill. Heift (1998) stated that student errors should be ranked by the priority of the error. Teachers, however, should emphasize just the important errors. Because of contingent errors, it is unnecessary to inform the students about all the errors at one time. Moreover, teachers should avoid redundant and misleading feedback. In other words, they should avoid providing multiple error reports to the learners. Inevitably, there will be situations in which there are multiple errors. What is needed is a filtering model such as the one that is a part of the system of Heift that can handle these errors and extract the contingent error. In this case, an error priority queue is used to determine the

order for presenting the instructional message to the learners (Heift, 1998). As a result, this approach can give feedback to the learner by considering the principle or important errors.

According to the characteristics of quality feedback (see Chapter 2), the prioritising of providing quality feedback could help the learners improve their lifelong learning. For example providing quality feedback as the sequence of these: Advance feedback (in the classroom), positive feedback; important error (as negative feedback); positive feedback again; restate correct answer feedback; thinking about students' action; individual feedback; asking key question; providing hint; feedback summary; feedback loop; continuous feedback and timing of feedback. Learning to think and communicate with a diagram would make the task easier (Brna et al., 2001). Thus, representation of the relation of giving feedback knowledge would be easier to help people understand various factors for provision of feedback issues via our feedback ontology diagram. Therefore, we propose the feedback ontology according to our investigation as shown in Figure 3.6. The feedback ontology was developed by the author. This was done by starting with the sources of feedback (Draper, 1999). The ontology was grown from this source. Other sources were taken, analysed and then linked together "by hand" based on the literature about feedback in this chapter. Apart from the knowledge of giving feedback, in order to achieve more knowledge of giving effective feedback content (McKendree, 1990), we also present a diagram from reviewing McKendree's work, as can be seen from Figure 3.7

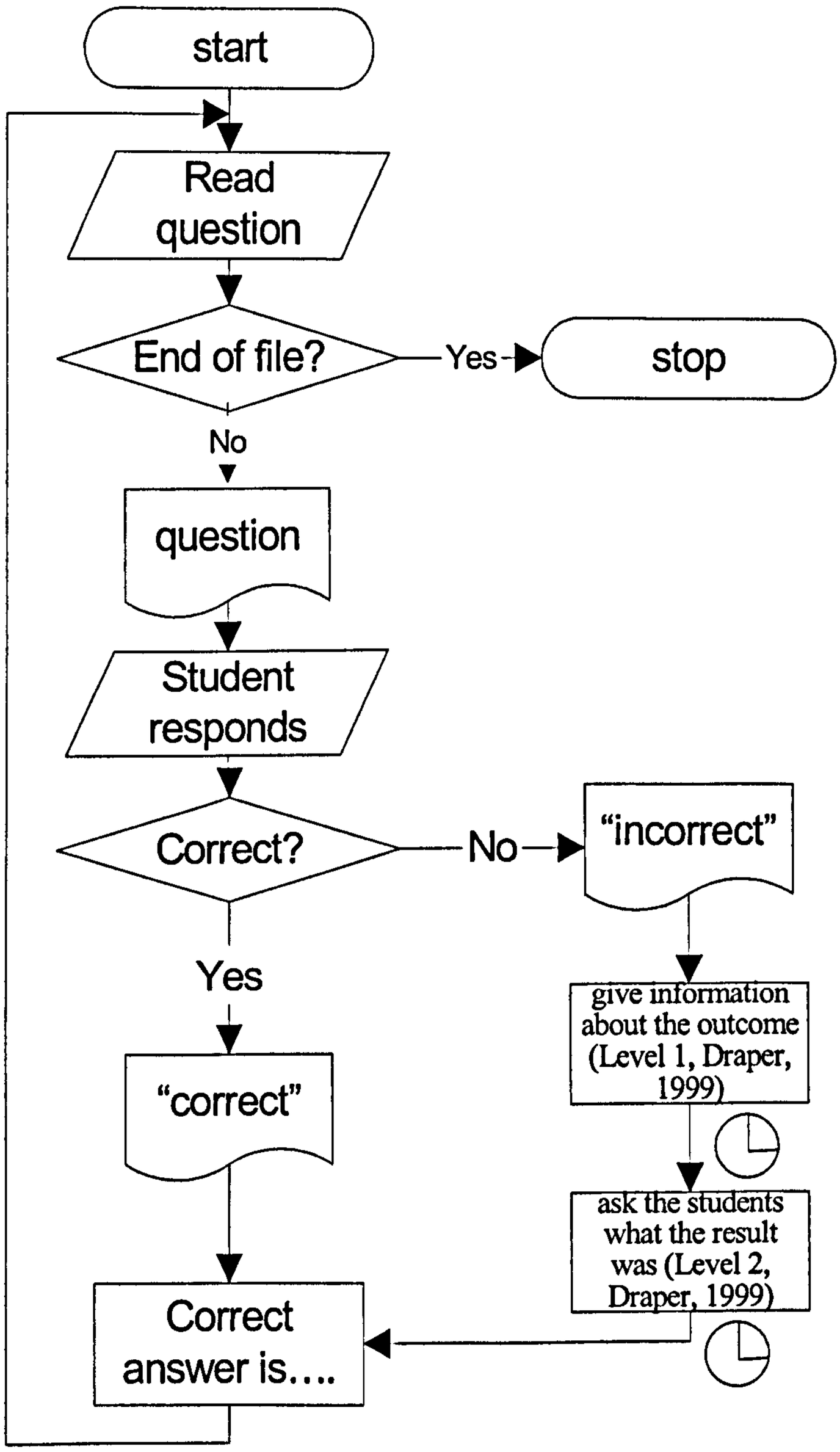


Figure 3.4 Combination of KOR, KCR, immediate and evaluative feedback and level of feedback

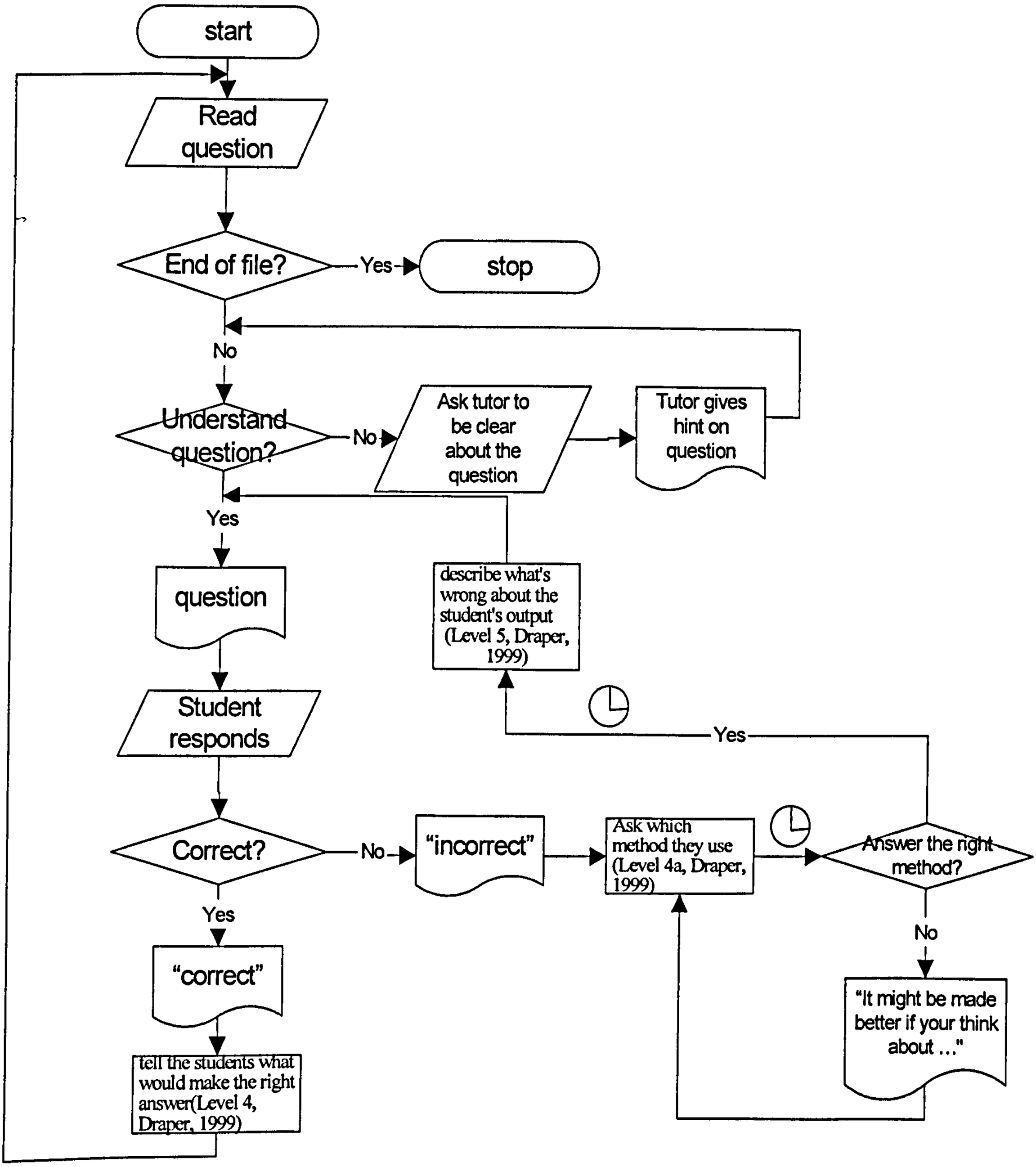


Figure 3.5 Combination of AUC, immediate and corrective feedback and level of feedback

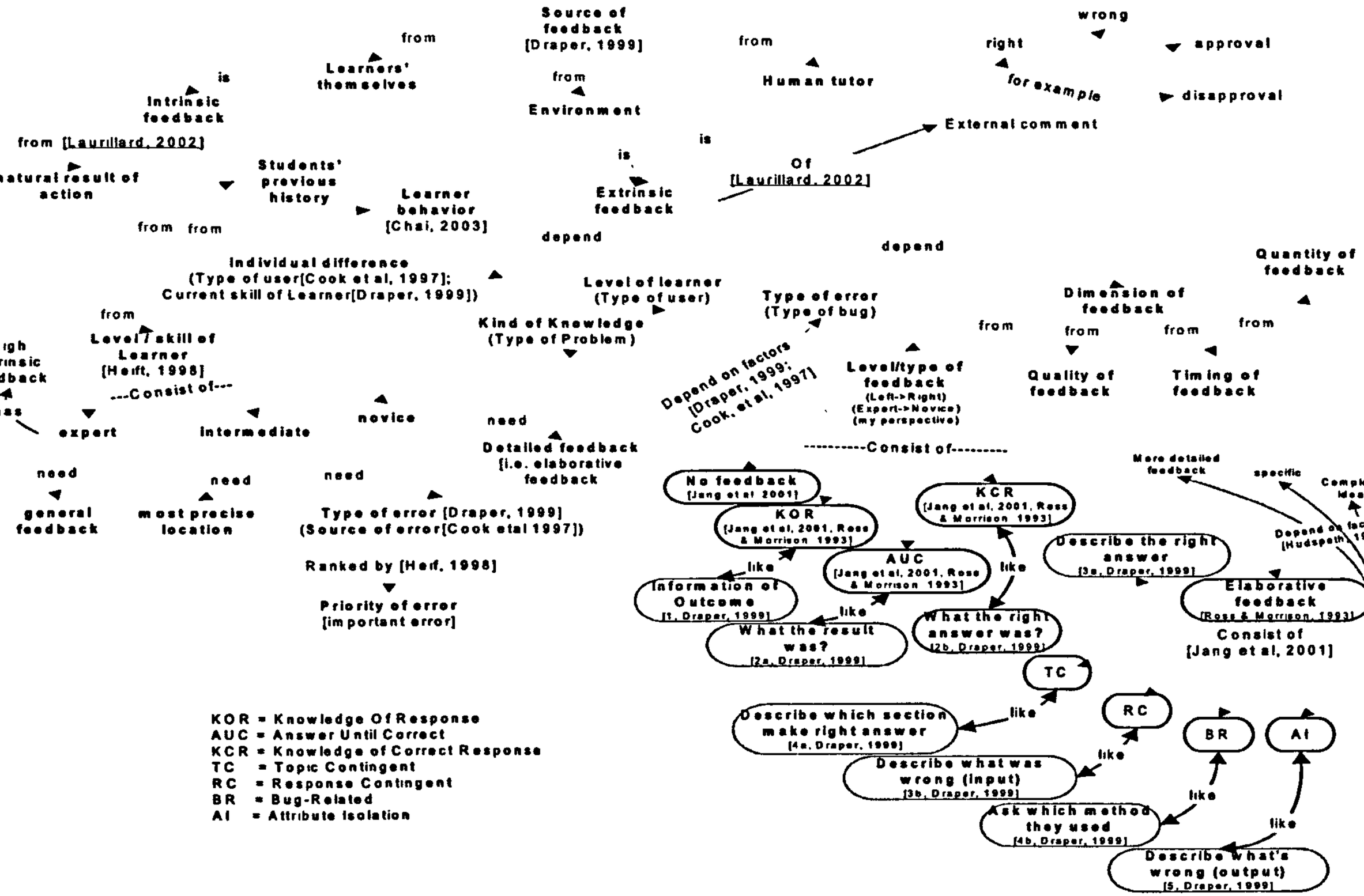


Figure 3.6 Feedback ontology

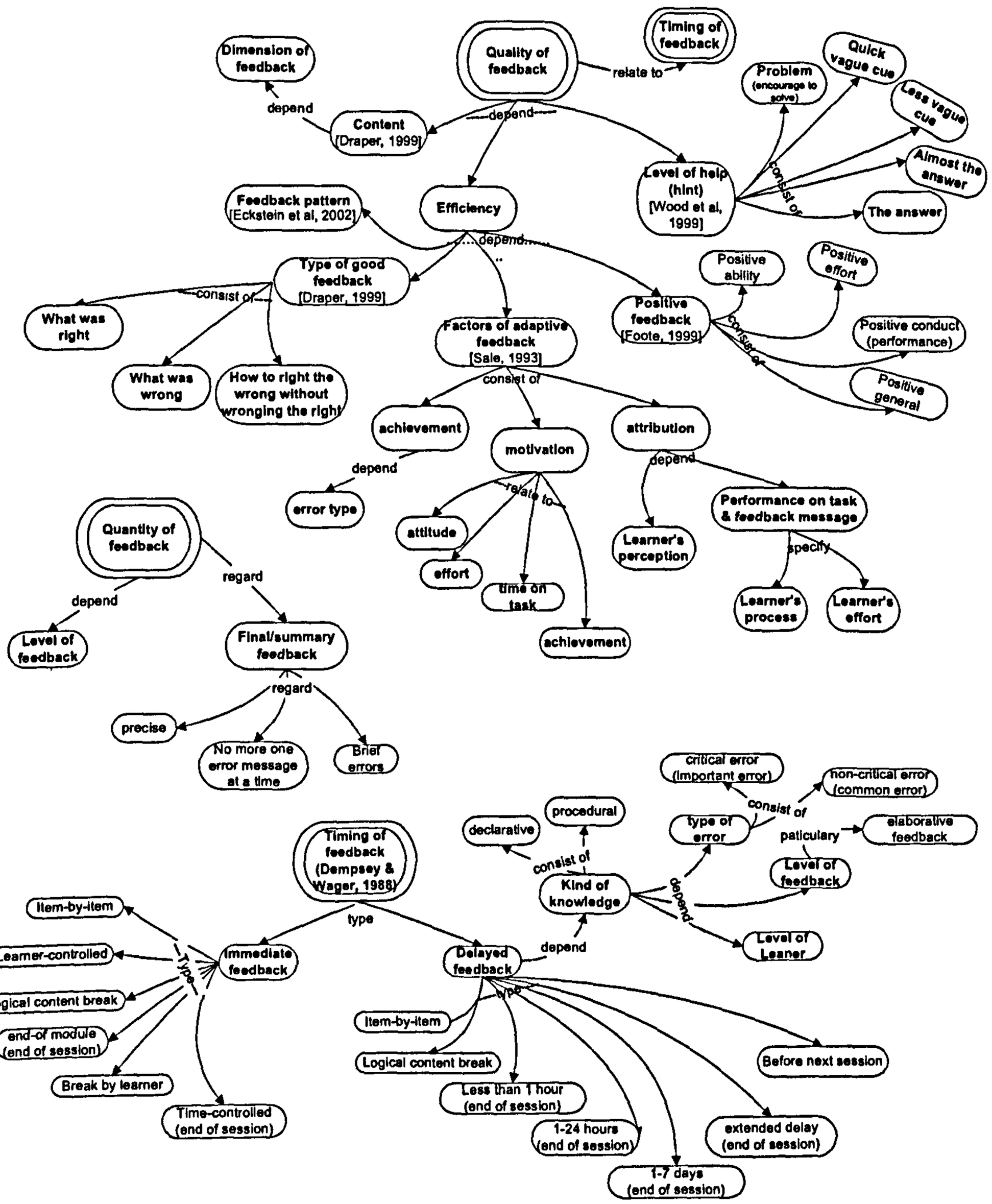


Figure 3.6 Feedback ontology (contd.)

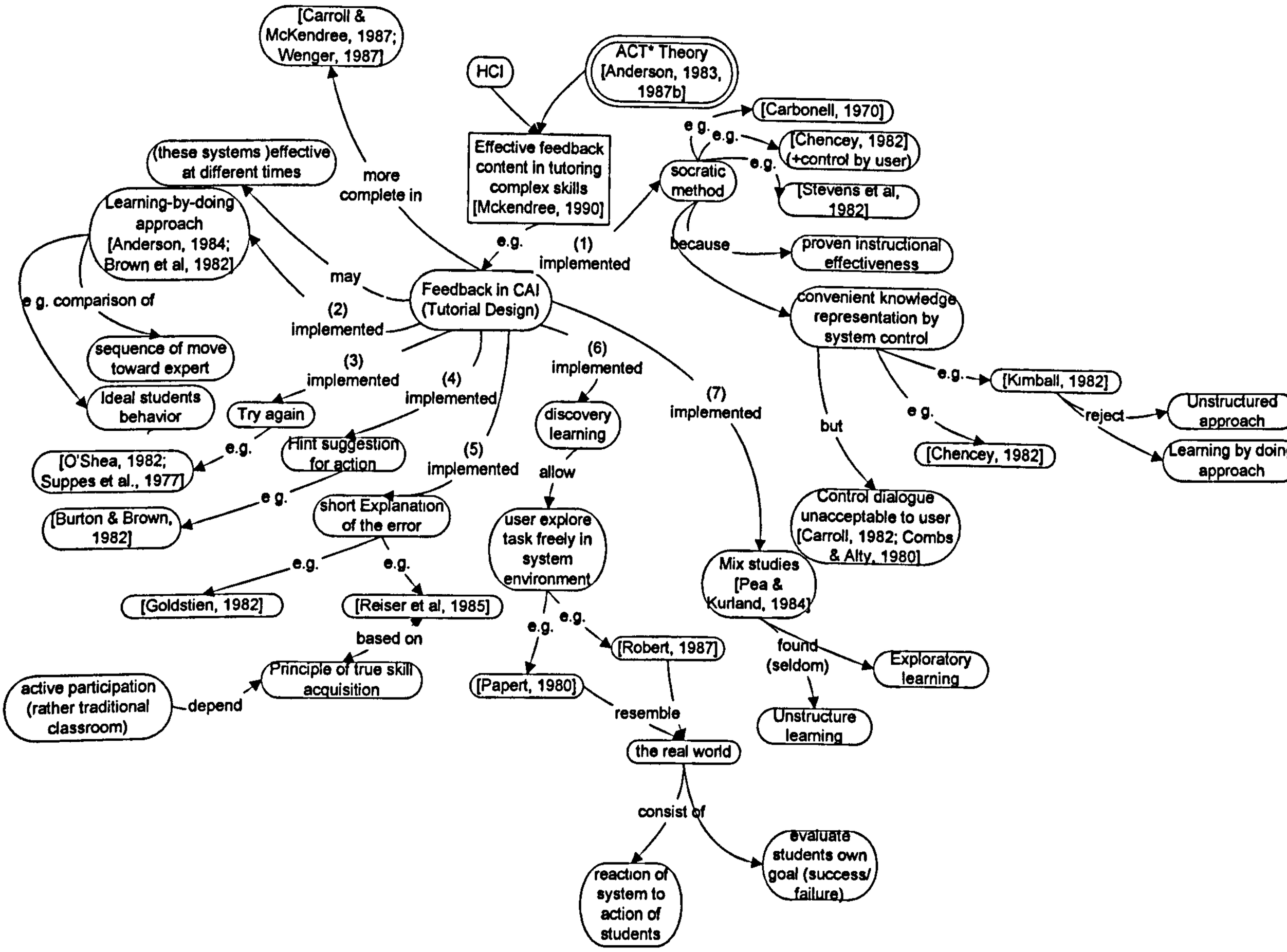


Figure 3.7 Effective feedback content diagram

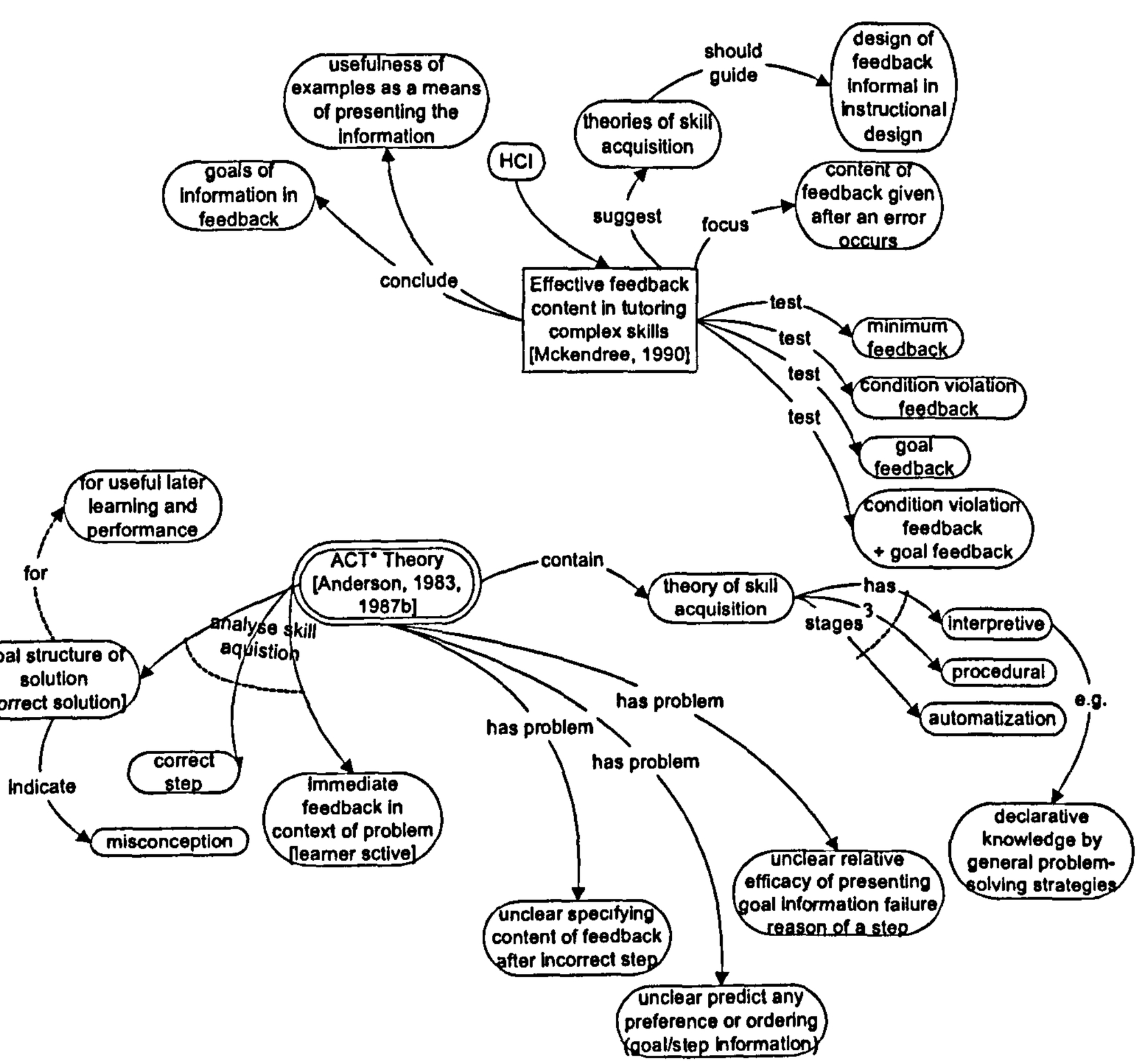


Figure 3.7 Effective feedback content diagram (contd.)

3.8 Discussion

With regard to different feedback as proposed in this chapter and from the literature of the practice of giving feedback to improve teaching (Brinko, 1993), there are a number of patterns for giving feedback but no empirical study (e.g. Feedback is more effective when negative information is "sandwiched" between positive information.) (Brinko, 1993). In addition, Eckstein, Bergin, & Sharp (2002) suggested that giving quality feedback should start and end with positive feedback in which the suggestions for improvement are sandwiched between these reinforcing comments as positive feedback. This also relates to the review of quality feedback of Draper (1999). Therefore, we argue that providing quality feedback should start and close with positive feedback. Besides, the analysis of feedback content and ontology of giving feedback will be employed in design principles of giving quality feedback (in the next chapter).

With regard to the content of feedback when the error happens as an explanation, Corbett & Anderson (2001) recently explored a formal research topic from early in the 20th century. They reported about when an error should be an offer of advice. They provided the explanation and response to the correct answer to the learner, but they did not mentioned how to train the teachers to provide quality feedback to the students. This brings us to the new system design. Namely, our system designs are apart from early automated instruction. The system is also different from general intelligent tutoring systems that only provide feedback and advice on a sub-goal in complex problem solving tasks (Corbett & Anderson, 2001). There is evidence shown that a group that receive different feedback more than one time can detect errors better than groups that receive the same feedback (Lhyle & Kulhavy, 1987) so it would be interesting to train the TA to give different feedback to students with regard to the history of student's performance and allow the student to resubmit the same assignment. Even though we examined several kinds of provision of quality feedback, we do not consider visual feedback or audio feedback. Furthermore, we also do not discuss the issue of providing audio feedback. Thus, according to our analysis from the previous chapter (see Chapter 3 to 4), the analysis of student's erorrs/weaknesses can be the components of system design to help the TAs learn to provide quality

feedback that would be sufficient to give feedback to students on their programming assignments in our context. Consequently we also classified error/ weakness type according to the priority of the error type. Thus, we will need to support a scaffolding of the TAs in a number of different ways. In the next chapter we present a framework for analysis of student weakness to be a domain of our system for learning to provide quality feedback.

Programming domain for training TAs to give quality feedback: A framework for the analysis of student performance

4.1 Introduction

In order to research providing scaffolding for the TAs, we have to select a domain. We have chosen learning to program because

- although automated marking systems for computer science assignment can provide the initial feedback (e.g. Course Master (Foxley et al., 2001; Higgins et al., 2002)) and include detailed feedback, automated systems are, by their very nature, unable to provide individual analysis and advice to students in the same way as a human marker or tutor can in an semi-automated system.
- We intend to add features that the automated marking systems can not perform but semi-automated ones can
- We have some experience of the domain ourselves.

An analysis of this domain provides the basis for the kind of feedback we can provide both to the TA and the learner.

The previous chapter reports how people learn to provide feedback, but how can we help them learn from that? In this chapter, we provide the framework used to underpin both the scaffolding system for the learner and, less directly, the help given to the TAs. There are many ways to structure a domain for training the TA to learn to give feedback. The structure of our domain relates to the life cycle of programming skills (e.g. design issue, implementation issue). This chapter presents experts' and TAs' perspective of giving feedback, giving feedback on a programming assignment, a framework for classification of types of weaknesses, pattern of error/weakness message, and discussion according to the following.

4.2 Experts' perspective of giving feedback

We worked alongside three different lecturers with experience in giving feedback, so we shall refer to them as expert tutors (see Appendix A). According to the domain topic, different experts have different methods. Expert A gives feedback to students according to the main problem students encountered in the submitted assignment. Expert B provided us with ideas in preparing detailed feedback in advance. This is consistent with the characteristics of giving quality feedback (see Section 2.2.2 in Chapter 2). Expert C emphasizes his feedback with analysis of the problem, design and implementation. Even though expert A and expert C mark programming assignments, they do not emphasize the content of the domain of the nature of programming for the students. We are aware that different experts have different background/skills/approach in giving feedback. Each lecturer has different criteria according to their personal experience (Ala-Mutka & Järvinen, 2004). Thus we structured the domain in terms of topics, programming skills or some other way.

4.3 TAs' perspective of giving feedback

We discussed giving feedback with two TAs. TA A is an experienced TA who always indicated every error and provided correct answers to students' assignment and sometimes provided feedback regarding important errors. Even though he has been trained to give feedback by the training program provided by the School, he still needs to learn how to give good feedback to the students. TA B is a novice TA who has never been trained to give feedback and also requires training to learn how to give good feedback to the students. According to the characteristics of giving good feedback outlined in Chapter 2, TA A does not always provide good feedback to students because he always indicated every error to the students' assignments. We are aware that different TAs have different background/skills in giving feedback, and based on the interviews with two TAs and conversations with lecturers, it would seem that they rarely receive sufficient training. Thus employing a system to support and train the TAs to give good feedback could help them achieve giving quality feedback skills.

4.4 Giving feedback on programming assignments

Marking systems as mentioned earlier (see Section 2.4 in Chapter 2) may help the teachers provide feedback to students. In our context, we are interested in how people learn to give feedback in different ways as to extract some characteristics from a real situation (e.g. giving feedback on programming assignment). In order to learn how to give feedback from such a situation, we need to know the literature and analysis of how computer-systems give feedback on programming assignments in the following:

4.4.1 How do computer systems give feedback on programming assignments?

In general, in terms of the programming environment, automated marking assignments deal with effective aspects, how to arrange important errors, and explanation of how to correct answers and avoid errors (e.g. SPROUT (Pardoe & Vickers, 1994; Rimmer et al., 1995); Ceilidh (Foxley et al., 1999) and CourseMaster (Foxley et al., 2001; Higgins et al., 2002); BOSS (Joy & Luck, 1998; Joy et al., 2000)). We will explore this issue below.

4.4.1.1 SPROUT

SPROUT (Simple Programs Routinely Observed Under Test) is an automatic assessment tool developed at Liverpool John Moores University to help novice students access their Pascal code. It can help the students practice ten programming exercises to receive feedback so that the teachers can have adequate time to help with other difficulties students experience (Pardoe & Vickers, 1994). This system structures the domain for giving feedback that consists of three assessments which are: source code profiling, identifier analysis, and code formatting in order to access how students script is designed in Pascal code from pseudo-code (structured English) for appropriate implementation and good style – generally acceptable in Pascal standard and give automatic feedback to a student as a text file –a summary of assessment- on the student's diskette. This system can report redundant errors to the student and not provide important errors to the student so they can practice exercises until achieving the complete result.

Common errors in this system comprise of: Errors made during file handling; Unnecessary assignments and initialisation statements; Errors in integer and real arithmetic calculations; Undesirable read and readln statements; Incorrect, undesirable, missing or extra writes and writelns; Incorrect or undesirable relational expressions used in WHILE, REPEAT, IF and FOR statements; Failure to follow the provided pseudo-code; Insertion of extra unnecessary BEGIN/END pairs; Incorrect declarations, typographic errors, undesirable order; and other common errors (Rimmer et al., 1995). The first evaluation resulted in effective employment of the system.

After the pilot study was evaluated, a year later Rimmer, Pardoe, & Vickers (1995) evaluated SPROUT with a comparison between automatic marking and manual marking. The result showed that SPROUT could not distinguish stylistic features of student code as good as manual marking can. Despite the usefulness of instant marking, from the perspective of some students using SPROUT it was too strict in some aspects (e.g. in the code profile that Rimmer, Pardoe, & Vickers found that there were several problems which are: The statement included in the students solution as required has been incorrectly implemented; The student has omitted a required statement; The student has included unnecessary statements; The student has implemented the required statements but in the wrong order (p.290)) as well as the unclearness of the meaning of good programming style according to Rimmer, Pardoe, & Vickers (1995).

Even though this system has a limited specification, it is not general purpose for all Pascal assignments, as well as not helping novice teachers to learn to give quality feedback. The aspect of assessment in SPROUT gives us an idea to build the TA system to help novice teachers or the TAs to learn to give quality feedback in a real situation – of marking programming assignments in terms of design problems, implementation problems, and style problems.

4.4.1.2 Ceilidh and CourseMaster

At the University of Nottingham, CourseMaster (Foxley et al., 2001; Higgins et al., 2002) is an assessment system based on software metrics developed from Ceilidh (Foxley et al., 1999) to give students informational feedback in the form

of a "feedback tree" as the structure of the feedback giving domain. Such feedback can be extended to help students to learn to identify type and problem, including dynamic correctness, dynamic efficiency, typographical analysis, complexity analysis and structural weakness, from their programming code and can only give students feedback quickly and according to the objective. In Ceilidh feedback to the student is limited to a mark composed of the results obtained by the marking tools that participated in the marking process. No direct justification to explain the loss of marks to the students is provided. In addition, no explanation is given for the student to improve their mark. One benefit of using a tree structure for representing the results in CourseMaster is that feedback with more details and precision can be created and presented to the students, according to Tsintsifas (2002).

The result of employing the CourseMaster, (Foxley et al., 2001; Higgins et al., 2002) showed that a considerable number of students preferred a system that tells them which areas they are weak in even it was difficult to give feedback in other areas (e.g. complexity and efficiency). A number of feedback areas from this system are controlled by teachers or developers. In spite of that, this system gives a tree structure of feedback and comment to students on how to improve and then provides links for further reading. It does not however provide any training for novice teachers or TAs to learn to give quality feedback.

4.4.1.3 BOSS

At the University of Warwick, BOSS (Joy & Luck, 1998) has been employed for many years in the form of semi automatic system for a number of courses concerned with programming assignments in computer science for fundamental coursework. Boss is a structured domain for giving feedback e.g. technique for design, and methodologies. This system consists of two versions: one for students to practice exercises for obtaining short feedback, another for the marker to provide further feedback and justify grades such as explaining student misconception of the result of testing from BOSS. Two year after the first version of BOSS, a new version of BOSS –called BOSS2 (Joy et al., 2000) was developed in Java language. Although evaluation from employing this system

from the students was generally favourable, it does not provide any component to train the novice marker for the provision of quality feedback.

4.4.1.4 AssessmentMaster

At the University of Joensuu, Finland, a system called AssessmentMaster (Suhonen et al., 2001) written in Java applet and emphasising graphics to motivate students to learn programming was used to assess students work. The purpose of this system differs from CourseMaster and BOSS in that CourseMaster offers programming instruction assistance while BOSS focuses on managing the course (e.g. help the marker access submitted students' assignments via the system as well as help testing assignment for giving feedback to students). Nevertheless, this model is only ideal to establish metadata by using XML language and insert feedback by guiding students to read more detail further to the current exercise being assessed. Suhonen et al (2001) argued that completely automatic assessment cannot suitably assess; on one hand, their model of semi-automatic assessment could help the teachers to edit and add greater quality to giving feedback automatically; however, they were disappointed with their results regarding the quality of feedback provided to students by the current method. Due to the fact that giving feedback is a crucial educational issue, they found that it to be too time consuming to give feedback when marking a large number of exercises and the quality of feedback could decrease. In the approach of "Virtual Certificate" project, in some cases it lacked obvious feedback concerning their errors so students were unable to develop their knowledge. Further, they developed their system to give more detailed feedback to a high number of students. They also suggested that it is useful to give visual feedback rather than text feedback in which this might be a new feedback dimension. This system may well close the gap of automatic marking systems that cannot either mark or solve some programming problems and provide a higher quantity of more useful feedback from authentic teachers.

This system structured the domain for giving feedback by considering the basics and from the aspect of introductory programming. This system focuses on tutors who might be either novice teachers or TAs. Nevertheless, how do we know they provide effective or good feedback to students to help them develop or

improve their learning? Novice tutors may not have enough experience to give quality feedback. Feedback should be clear and adequate because it will be time consuming for tutors to devote a great deal of time to give feedback to a large number of students. Therefore, it would be better to develop computer-support to train them to provide quality feedback to the learners.

4.4.1.5 Summary: How do computer systems give feedback on programming assignments?

According to automated marking assignments as mentioned earlier, they have a structure domain for giving feedback; however, they do not summarise error or provide feedback on important errors, and if a number of the same feedback messages are found it does not arrange error feedback i.e. support the tutors to organise feedback. A comparison of features of the systems can be seen in Table 4.1. Although such systems could help teachers provide feedback to the students, our research specifies how to train teachers to give quality feedback on students' assignments. In addition, automated marking systems for computer science assignments are still limited in the range of feedback they can give even though some systems (e.g. Course Master (Foxley et al., 2001; Higgins et al., 2002)) include detailed feedback, automated systems are, by their very nature, unable to provide individual analysis and advice to students in the same way as a human marker or tutor can. In addition, there is no common criteria in marking programming assignments according to Ala-Mutka & Järvinen (2004). They developed a semi-automatic marking system model for giving feedback as general automatic marking systems are limited in giving descriptive feedback. As a consequence, in this manner, developing semi-automated support helps TAs to do such tasks that would be helpful for both markers and learners. In addition, most semi-automatic marking systems are developed to correct student error in various programming languages according to the basic measurement. However, different teachers emphasize different features, according to their personal experience. (Ala-Mutka & Järvinen, 2004). Therefore, this could be useful to employ in Prolog programming language as a domain for training TAs to give quality feedback because it is Artificial Intelligent Programming Language and this is also a difficult course so it may be helpful in providing feedback to the learners.

Table 4.1 Investigation of providing feedback from semi- automatic /automatic marking systems

System	Effective aspects	How to give feedback	Explain how to		Evaluation
			correct error	avoid error	
SPROUT (automatic marking system) (Rimmer et al., 1995)	- provide instant marking to help students practice exercise	- Feedback on common errors to a text file to each student - require the tutors to justify grade and explain errors	N/A	NO	– problem from code profile
Ceilidh (automatic marking system)(Foxley et al., 2001)	- provide instant feedback to justify grade	- instant feedback explain why student lost marks that depend on students' grade, give detailed weakness(if any) of the solution but not explain why it is wrong or how to improve	NO	NO	- Limited feedback - no explanation
BOSS (semi-automatic marking system)(Joy & Luck, 1998; Joy et al., 2000)	- assist the marker to mark students script	- detailed feedback given by marker - short feedback by the system to students to practice	N/A	N/A	- Most students prefer the system
CourseMaster (automatic marking system)(Foxley et al., 2001)	- The amount of feedback is controlled by teachers or developer - helpful feedback	- Given feedback by the system and comment how to improve student solution via the link on further reading	N/A (link to further reading is the Topic Contingent feedback)	N/A	- Most students prefer the system to tell them what areas they are weak in.
Assessment Master (semi-automatic marking system)(Suhonen et al., 2001)	- Good model of self-evaluation	- Feedback by marker and the system	N/A	N/A	Problem from giving quality feedback

4.4.2 How to mark error

Due to the different problems of each learner, students may experience problems on receiving feedback from their tutors. Further, finding bugs/errors is difficult and time consuming for the learner. Thus, giving feedback on a programming assignment is of interest to us according to the programming marking tool as mentioned above. TAs may not want to tell the learners that their assignments are correct or incorrect directly; however, they may tell them it is acceptable or unacceptable or how to improve their student programming skill. Because they do not know everything is correct, the observation of other fields for providing feedback could inspire some ideas of such tasks in the previous section on how to

mark an error which is close to the correct answer (e.g. essay feedback, mathematics feedback, science feedback).

Furthermore, even though assessment in the classroom is interesting for learners (Brown & Knight, 1994), we argue that it would be difficult for marking programming assignment because the TAs could spend more time with each learner. However, if teachers design moderate sized assignments, they could considerate this by giving students sub-assignments; thus, they could mark assignments in a reasonably short time in the classroom.

Considering marking systems as mentioned earlier, when can we learn from other semi-automatic systems? It is not necessary to observe just programming feedback. However, providing feedback via automatic systems is still not as good as feedback given by human tutors. In order to help TAs learn to provide good feedback to students, it is vital to have a framework for analysis of students programming weakness and also an approach of giving good feedback on programming assignments as the characteristics of good feedback for marking an assignment (see Chapter 3).

We have chosen a framework that stresses three different kinds of skill associated with the design issue (see Section 4.5.1), the implementation issue (see Section 4.5.2), and the style issue (see Section 4.5.3). Before presenting a framework for classification on programming skill, it is important to know how computer systems help the teacher to provide feedback to the student. Do they structure the domain for feedback? We have investigated the marking systems for programming assignments as can be seen from the following.

4.5 A Framework for classification of types of weaknesses

In general, compiler gives warning to the users; nevertheless, they may not know the exact problem to solve. In particular, a novice programmer may not understand all of a warning message. Learning programming language in University, TAs may tell students, for example, "Did you really mean to implement this...." as an implement issue. In other words, it will be helpful to have a system to help TAs learn together with providing feedback messages to

students e.g. "I suspected that you have a design and implementation problem, as I cannot read it very well". Besides, the system may inform TAs "Is it useful information for you to choose" i.e. TAs decide that it should be 'error' or 'issue' (system may give 2 choices for TAs then report to students)". We are therefore required to classify students' weakness so that they can be used for particular cases -situated learning- for TAs to learn to give feedback to students.

Concerning students' weaknesses/errors in design, implementation, and style issues, admittedly, effective software development depends on software process which is "a set of activities and associated results which produce a software product" and software development is one of a set of activities of software processes (Sommerville, 2001). The waterfall approach is one of a number of different models of software development in which its approach consists of requirements specification, software design, implementation, testing and maintenance, i.e. activity of software development includes requirements specification, software design, implementation, testing and maintenance. In such a case, in order to have a qualitative teaching-learning programming language for the novice learner, in our context, it is important to focus on design, implementation and maintenance. In terms of maintenance, generally, it is concerned with improving software modification to give higher quality software. It can result in easy to modify software when it is coded in a readable format or layout in which it is a part of programming style. For that reason, in this thesis, we use style issues for readable format or layout issues to enclose our limitation rather than abstract style or 'how to' track program. In other words, in this thesis, we pay special attention to teach novice teachers or TAs to give feedback to students' assignments. According to TAs or novice teachers understanding, when TAs say something on a students' work, they may say, for example, "This framework should be" a) "your solution to solve this problem is wrong or students are thinking along the right lines to write the program (implement), but it's gone wrong in design (design)", b) "Is it the wrong style?". For these reasons, we have three different issues for classifying types of students' weaknesses in our context. In addition, some systems can provide such answers, for example SPROUT (Pardoe & Vickers, 1994; Rimmer et al., 1995), Boss (Joy & Luck, 1998; Joy et al., 2000), Celidh (Foxley et al., 1999), and CourseMaster (Foxley et al., 2001;

Higgins et al., 2002). Even though our system can give "design weakness messages" grouped by design issue as to help TAs make a decision before providing feedback on students' assignment, sometimes TAs might think it should be grouped by 'implementation'. However, our system can help them create additional feedback messages with three error/weakness types classified by the system (design, implement and style issue).

Thus, our focus is on giving feedback to novice students' solutions in the stages of design, implementation and style problem of students' weaknesses on programming assignments rather than syntax or semantic problem. Most students can detect their syntax error from a compiler; however a semantic problem is, sometimes, quite broad to explain or give a definition to. According to Watt (1990), syntax and semantics appears in every language; any programming language syntax has its own program form, i.e., how to put expressions, commands, declarations, etc. together to structure the programs; any programming language semantics has its own programs meaning, i.e., how their operation is on the execution of computers. Besides, the creation of phrases in a language relates to syntax because phrases consist of a vocabulary of symbols (expressions, command, declaration, and finish programs) and rules to put them together (1991). Consequently, giving a meaning of syntax depends on the structure of each programming language which is not included in our context.

Turning to semantics is generally mapping from one system to another in which people agree on what they understand. Semantics is the relationship between the expressions of language and their meaning (Schwarez, 1969) in which it is accepted by social communication, i.e. communicants do agree with their meaning of communication; however, in formal computing, we can specify a semantics for a programming language. Moreover, each programming language has its own syntax and semantics. In addition, Watt (1991) also pointed out the meaning of semantics is like the meaning of phrases in a language; nevertheless, programming language semantics differs from the meaning of phrases in a language because the meaning of each phrase in programming language will be computed to behave as per the execution when programs run. However, "semantics is much more difficult to specify than syntax, and no method of

specifying semantics formally has achieved wide popularity outside the academic community" (p.7).

In fact, every programming language has its own semantics. In this thesis, we do not use the semantic issue as a criteria for the classification of students' weaknesses on programming because there are no methods to define formal semantics (Watt, 1991) and its meaning is not clear enough to give a definition. In addition, logic semantics is better to understand, if we create our own logic in which other people already have. Nevertheless, we do not map logic to formal issues because there are too many ways to map. As a result, it is quite difficult to find the context of semantics in programming language. In particular, each programming language has its own semantics; therefore, we avoid the use of semantic issues because it is not totally well-defined for design, implementation and style in which they reflect the sequence of software development in which these can be clear for the definition. For example, teachers may have a process to teach novice students e.g. "students should design then implement then use a certain style". Accordingly, this thesis is a different domain from software engineering issues in which we emphasise helping the teacher to give feedback on students' weaknesses. Therefore, we can divide the property of the types of students' weaknesses based on our understanding in a way similar to software development as described above (Sommerville, 2001), and programming language; and classify the types of students' weaknesses into design, implementation, and programming style as shown in Figure 4.1

4.5.1 Design Issue

When learning to write good programs, students should learn how to write a well-designed program in order to improve software design skills.

> *"**Design:** one of the main phases of any software development, which comes after the SPECIFICATION but before the writing of any CODE. The design stage includes tasks such as breaking down the problem into manageable sub-problems, choosing suitable DATA STRUCTURES and ALGORITHMS. Program design is an intensive studies subject, and there are many competing design methodologies that claim to simplify or formalize the task ,p.130)" (Pountain, 2001)*

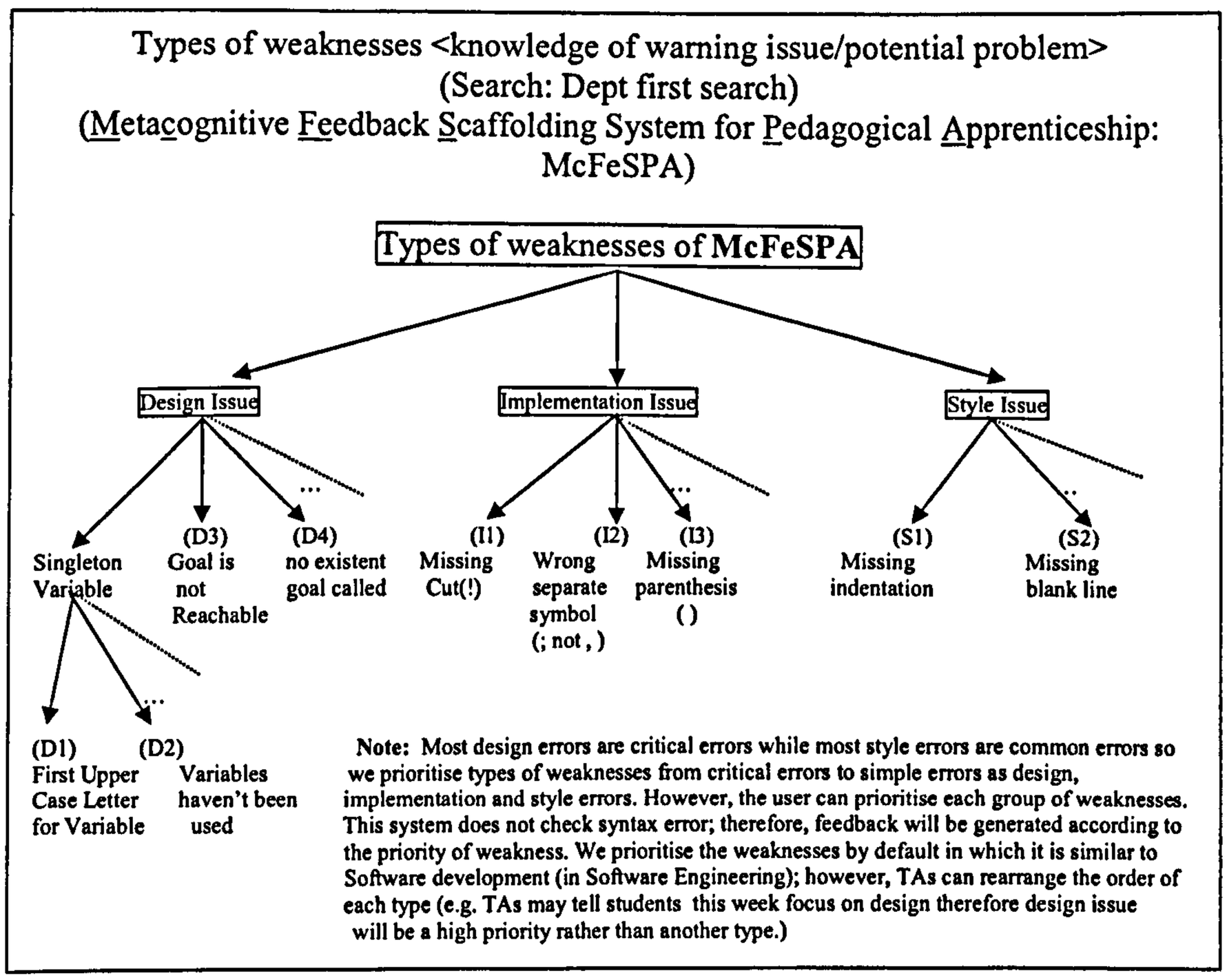


Figure 4.1 Types of weaknesses

Brown and Chandrasekaran (1989) have classified three classes of design problem, with a number of structures:-

- open-ended creative design in which the goals are not clearly specified and there are no ready-made ways to decompose the problem into smaller elements;
- design for which the problem can be decomposed in standard ways but major components of the design must be created from scratch;
- design is relatively routine in which this class is quite complex.

Debugging of design errors was done by Bental (1994) who considered the error from students' program into three problems space:

- general algorithm design, for example hidden bugs e.g. predicate8 was not

[8] Predicate is the short name of a predicate symbol which is the name of a procedure ,like functors, has two attributes: a name and an arity (Kluz'niak et al., 1985).

called (p.62);

- algorithms design in the domain of AI game-playing; and
- Prolog program design which consists of general software design, design in an unfamiliar problem domain, and design in an unfamiliar programming language e.g. misconception about the prolog programming (p.55-56).

For such reasons, in our domain we are interested in design in the context of an unfamiliar programming language in which it is a part of prolog program design. According to Bental,

> *"Novices learning a new programming language make many mistakes because they have a flawed understanding of the behaviour and structures of primitives and the Prolog execution model. They are also ignorant of the standard programming techniques in the programming techniques in the programming languages. (p.54)"*

She also studied students' design errors and created these criteria without prioritising: use of standard and appropriate programming methods; correctness; robustness; clarity and readability; appropriate use of Prolog's declarative features (p. 55). She suggested a number of protocols of design decisions that relate to implications for detecting and critiquing design errors. In addition, she found many of the requirements of recognising and correcting design decisions in which "... parts of the code cannot be considered in isolation but must also be considered in the context in which they are being called; ...simulation of the code is necessary to understand, the students' code, to detect flaws in that code and to check the correctness of proposed improvement to that code ..." (p. 82-83). This is a design problem that we are interested to continue to explore to help novice teachers give more feedback to students on such a problem. The following paragraphs will show examples of design problems. When Problem domain is finding X from the equation of $\mathbf{ax^2+bx+c = 0}$

$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

if $d = b^2 - 4ac$

Then if $d < 0$ then we can not find the solution of X.
if $d = 0$ then X = -b/2a
if $d > 0$ then X = such solution

As can be seen from Figure 4.2, this is an example of hidden bugs i.e. predicate is not reachable. Due to the fact that the user may forget to call predicate 'final/0', we define this error as **design weakness type 3**.

According to Figure 4.3, this is a prolog code in which no predicate is called. The program couldn't find predicate 'final/0' during the execution state. The user may forget to put predicate 'final/0' so we point out this problem as **design weakness type 4**.

Bental (1994) also modeled students' design of code structure and code behavior at the high design level. From that model, we are interested in behavioral design on the task of "the identification of variables in the program with objects in the program domain" (p.85). The following paragraphs show a prolog code that will obtain a warning message as "singleton variable". In addition, the user may forget to add design **x1** variable in line 21 as an upper-case letter, according to Figure 4.4. Therefore, we assign this as **design weakness type 1**.

As shown in Figure 4.5, this is the prolog code that will obtain a warning message as "singleton variable". In addition, some variables haven't been used e.g. variable A at line 6; however, they should be underscore (_) instead. As a result, we define this problem as **design weakness type 2**.

4.5.2 Implementation Issue

In learning to write good programs, students should learn how to achieve good implementation in order to improve implementation skills. Pountain (2001) defined the meaning of implementation as

> *"**implementation**: ...The term carries special significance in programming language design, because many (if not most) language specifications are incomplete, leaving certain details up to the implementer. Such details are said to be IMPLEMENTATION-DEFINED or IMPLEMENTATION-SPECIFIC (p.240). **implementation-defined**: Use of features in a programming language that are not laid down in its standard, but are to be decided by each implementer (p.240). "**implementation-specific**: Use of some feature of a programming language that is present only in a particular implementation (p.240)."*

Further, Illingworth and Pyle (1996) also defined the meaning of implementation as writing or coding programming language after the design of a

system in which it is written in different language on different hardware. In this thesis, implementation issues are based on the methods of Shapiro (1983) which can be applied to Prolog programs to diagnose three kinds of errors: termination with incorrect output, termination with missing output and nontermination. In this manner, we have chosen the error types of the termination with incorrect output to be students' weakness on implementation. Consequently, in terms of implementation in this thesis, it relates to students' weaknesses on writing prolog program and termination with incorrect output.

In the case of wrong implementation, as seen in Figure 4.6, it can be missing the cut (!) as shown on line 8 of the following example so we indicate this error as **implementation weakness type 1.**

It is crucial to spot the problem in the final closed loop. In order to control backtrack, students should make sure in other cases that they did add cut (!) in the appropriate case in their code. In this manner, there is an overlap condition. In general, it is hard to identify test. Due to the fact that generally using cut is quite difficult, we look at the simple cut. For a marker, we are looking at case structure for implicit test. In addition, we do not look at it in-depth and specifically in case structure due to the fact that our thesis does not focus on only using cut (!) symbol. For this reason, we do not look at all scenarios for using cut.

In the case of symbol semi-colon (;) on line 7 as shown in Figure 4.7, it should be changed to symbol comma (,) instead. Namely, if a user uses semi-colon, in the case of the result of D>0, then the program can find the solution but the real result will be "No solution". Consequently, we define this symptom as **implement weakness type 2.**

In the case of missing parenthesis or wrong implementation, from Figure 4.8, it can result in the wrong solution in that the program will calculate the solution according to the priority of operator (do '*' and '/' before '+' '-')", also there is no warning or error by compiler. Hence, we call this problem **implementation weakness type 3.**

4.5.3 Style Issue

According to software development issues as discussed above, good programming style can help programmers develop existing software; therefore, learning to write good programming, students should not only write the correct program, but their program should also have a good writing style and be easily readable by humans in order to be used to practice programming skills. Besides, the majority of problems relating to computer programming issues are readability, portability, learnability, maintainability (Norvell, 2001). In this manner, it is significant to give stylistic feedback to students. Bratko (2001) defined programming style as

> *"Programming style(p.176)...to produce programs that are readable and easy to understand, easy to debug and to modify.... some rules of good style(p.176-177)...the layout of program is important, spacing, blank lines and indentation should be consistently used for the sake of readability. Clauses[9] about the same procedure should be clustered together; there should be blank lines between clauses (unless, perhaps, there are numerous facts about the same relation); each goal can be placed on a separate line.(p.177)."* (Bratko, 2001)

Concerning PRAM (Mansouri et al., 1998), a system for marking Prolog based on Ceilidh (Foxley et al., 1999) it has various style metrics to measure the style of prolog programming, for example, comments, user-defined identifier, layout (e.g. blank lines, indentation), cause. From these styles, our thesis focuses on the measure of layout, i.e. blank lines and indentation.

It can be seen from Figure 4.9 that this is an example of inappropriate style of program writing, in the case of missing indentation of the body of the goal between lines 2, 3, and 4. As a consequence, we define this problem as **style weakness type 1.**

Figure 4.10, is an example of unsuitable programming writing style. In the case of missing blank lines between the different clauses, i.e., the end of a clause (run/3) in line 4 and the beginning of another clause (solve/3) in line 5 it should have another blank line to separate these, thus we call this weakness **style weakness type 2.**

---

[9] Clauses, or Horn clauses are rules, facts and queries. Rules are statements of the form: $A \leftarrow B_1, B_2, \ldots, B_n$ where $n \geq 0$. **A** is the **head** of the rule, B_i**'s** are its body. Both **A** and B_i**'s** are **goals** ; a fact is a special case of a rule when $n = 0$. Facts are also called unit clauses and a special name for clauses with one goal in the body when $n = 1$ (Sterling & Shapiro, 1986).

As a consequence, it seems reasonable to conclude that we can analyse students' weaknesses in general weakness of prolog assignment into 3 kinds:

1)Design Issue

- Singleton Variable
  >> D1) First Upper case Letter for Variable
  >> D2) Variable has not been used
- D3) Goal is not reachable
- D4) No existent goal called

2) Implement Issue

- I1) Missing Cut (!) in the case of D < 0
- I2) Wrong Distinction symbol which actually should be ',' but user thinks it is ';'
- I3) Wrong inference from the equation $\frac{-B+S}{2A}$ due to missing parenthesis so it should be (-B+S)/2A. If there are no parentheses, the program will calculate S/2A before plus -B.

3)Style Issue

- S1) Missing indentation of the body of the goal
- S2) Missing Blank line

For such classification, we can point out the explanation students' weaknesses pattern (Annotate explanation of weaknesses) in the following.

In the case of declaration predicate 'final/0' and the program never calling it at all, also there is no warning or error by the compiler.

```
run(A, B, C) :-                          /*1 */
  D is B*B - 4*A*C,                      /*2 */
  solve(A, B, D).                        /*3 */
                                         /* 4*/
solve(_, _, D) :-                        /*5 */
  D < 0,                                 /*6 */
  write('No solution'),                  /*7*/
  !.                                     /*8 */
solve(A, B, D) :-                        /*9 */
  D = 0,                                 /*10 */
  X is -B/ (2*A),                        /*11*/
  write(' x = '),                        /*12*/
  write(X),                              /*13*/
  !.                                     /*14*/
solve(A, B, D) :-                        /*15*/
  S is sqrt(D),                          /*16*/
  X1 is (-B + S) / (2*A),                /*17*/
  X2 is (-B  - S) / (2*A),               /*18*/
  write('x1 = '),                        /*19*/
  write(X1),                             /*20*/
  nl,                                    /*21*/
  write(' and x2 = '),                   /*22*/
  write(X2),                             /*23*/
  nl.                                    /*24*/
                                         /*25*/
final:-                                  /*26*/
   write('good bye').                    /*27*/
```

Figure 4.2 Design Weakness Type 3

In the case of no declaration predicate 'final/0' but the user calls it, also there is no warning or error by the compiler.

```
run(A, B, C) :-                          /*1 */
   D is B*B - 4*A*C,                     /*2 */
   solve(A, B, D),                       /*3 */
   final                                 /*4 */
                                         /*5*/
solve(_, _, D) :-                        /*6*/
   D < 0,                                /*7*/
   write('No solution'),                 /*8*/
   !.                                    /*9 */
solve(A, B, D) :-                        /*10 */
   D = 0,                                /*11 */
   X is -B/ (2*A),                       /*12*/
   write(' x = '),                       /*13*/
   write(X),                             /*14*/
   !.                                    /*15*/
solve(A, B, D) :-                        /*16*/
   S is sqrt(D),                         /*17*/
   X1 is (-B + S) / (2*A),               /*18*/
   X2 is (-B  - S) / (2*A),              /*19*/
   write('x1 = '),                       /*20*/
   write(X1),                            /*21*/
   nl,                                   /*22*/
   write(' and x2 = '),                  /*23*/
   write(X2),                            /*24*/
   nl.                                   /*25*/
```

Figure 4.3 Design Weakness Type 4

```
run(A, B, C) :-                          /*1 */
   D is B*B - 4*A*C,                     /*2 */
   solve(A, B, D),                       /*3 */
   final.                                /*4 */
                                         /*5*/
solve(_, _, D) :-                        /*6*/
   D < 0,                                /*7*/
   write('No solution'),                 /*8*/
   !.                                    /*9 */
solve(A, B, D) :-                        /*10 */
   D = 0,                                /*11 */
   X is -B/ (2*A),                       /*12*/
   write(' x = '),                       /*13*/
   write(X),                             /*14*/
   !.                                    /*15*/
solve(A, B, D) :-                        /*16*/
   S is sqrt(D),                         /*17*/
   X1 is (-B + S) / (2*A),               /*18*/
   X2 is (-B  - S) / (2*A),              /*19*/
   write('x1 = '),                       /*20*/
   write(x1),                            /*21*/
   nl,                                   /*22*/
   write(' and x2 = '),                  /*23*/
   write(X2),                            /*24*/
   nl.                                   /*25*/
                                         /*26*/
final:-                                  /*27*/
   write('good bye').                    /*28*/
```

Figure 4.4 Design Weakness Type 1

```
run(A, B, C) :-                    /*1 */
  D is B*B - 4*A*C,                /*2 */
  solve(A, B, D),                  /*3 */
  final.                           /*4 */
                                   /*5*/
solve(A, _, D) :-                  /*6*/
  D < 0,                           /*7*/
  write('No solution'),            /*8*/
  !.                               /*9 */
solve(A, B, D) :-                  /*10 */
  D = 0,                           /*11 */
  X is -B/ (2*A),                  /*12*/
  write(' x = '),                  /*13*/
  write(X),                        /*14*/
  !.                               /*15*/
solve(A, B, D) :-                  /*16*/
  S is sqrt(D),                    /*17*/
  X1 is (-B + S) / (2*A),          /*18*/
  X2 is (-B  - S) / (2*A),         /*19*/
  write('x1 = '),                  /*20*/
  write(X1),                       /*21*/
  nl,                              /*22*/
  write(' and x2 = '),             /*23*/
  write(X2),                       /*24*/
  nl.                              /*25*/
                                   /*26*/
final:-                            /*27*/
  write('good bye').               /*28*/
```

D_2

Figure 4.5 Design Weakness Type 2

```
run(A, B, C) :-                    /*1 */
  D is B*B - 4*A*C,                /*2 */
  solve(A, B, D),                  / *3 */
  final.                           /*4 */
                                   /*5*/
solve(_, _, D) :-                  /*6*/
  D < 0,                           /*7*/
  write('No solution').            /*8*/
solve(A, B, D) :-                  /*9*/
  D = 0,                           /*10 */
  X is -B/ (2*A),                  /*11*/
  write(' x = '),                  /*12*/
  write(X),                        /*13*/
  !.                               /*14*/
solve(A, B, D) :-                  /*15*/
  S is sqrt(D),                    /*16*/
  X1 is (-B + S) / (2*A),          /*17*/
  X2 is (-B  - S) / (2*A),         /*18*/
  write('x1 = '),                  /*19*/
  write(X1),                       /*20*/
  nl,                              /*21*/
  write(' and x2 = '),             /*22*/
  write(X2),                       /*23*/
  nl.                              /*24*/
                                   /*25*/
final:-                            /*26*/
  write('good bye').               /*27*/
```

I1

In the case of the value of D < 0, this program will ask the user to execute the next predicate in which it will execute in the wrong case; therefore the user should add the symbol cut (!) at line 8

Figure 4.6 Implement Weakness Type 1

```
run(A, B, C) :-                          /*1 */
   D is B*B - 4*A*C,                     /*2 */
   solve(A, B, D),                       /*3 */
   final.                                /*4 */
                                         /*5*/
solve(_, _, D) :-                        /*6*/
   D < 0,                                /*7*/
   write('No solution'),                 /*8*/
   !.                                    /*9 */
solve(A, B, D) :-                        /*10 */
   D = 0,                                /*11 */
   X is -B/ (2*A),                       /*12*/
   write(' x = '),                       /*13*/
   write(X),                             /*14*/
   !.                                    /*15*/
solve(A, B, D) :-                        /*16*/
   S is sqrt(D),                         /*17*/
   X1 is (-B + S) / (2*A),               /*18*/
   X2 is (-B  - S) / (2*A),              /*19*/
   write('x1 = '),                       /*20*/
   write(x1),                            /*21*/
   nl,                                   /*22*/
   write(' and x2 = '),                  /*23*/
   write(X2),                            /*24*/
   nl.                                   /*25*/
                                         /*26*/
final:-                                  /*27*/
   write('good bye').                    /*28*/
```

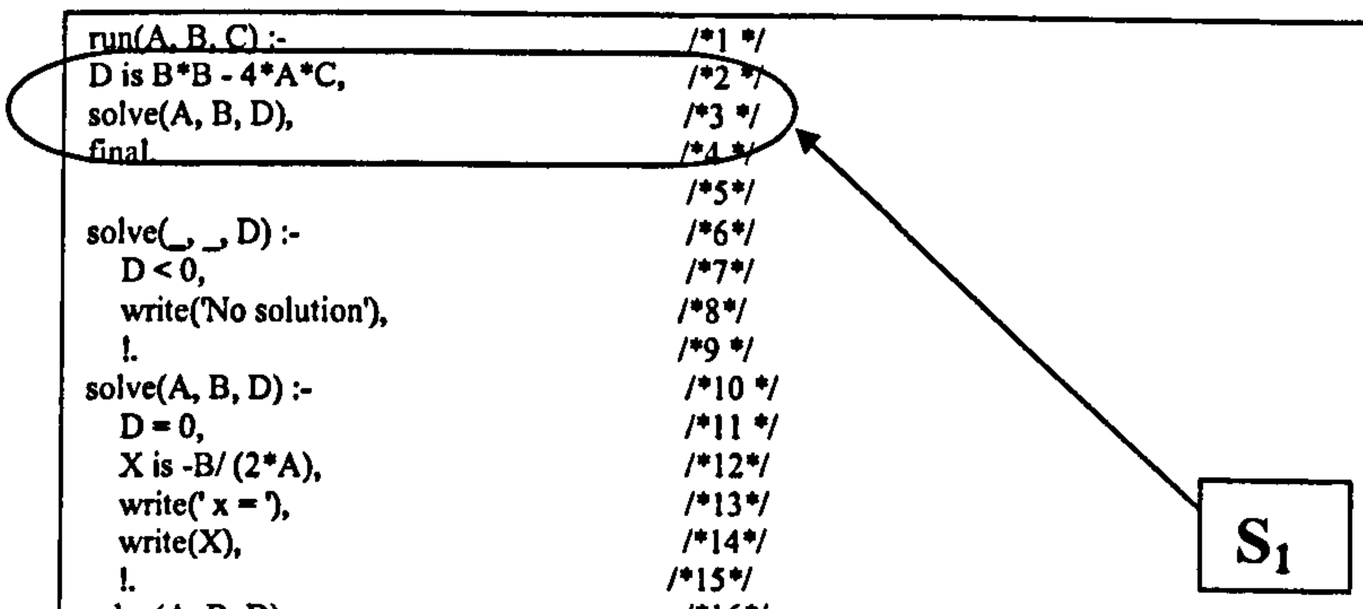
Figure 4.7 Implement Weakness Type 2

```
run(A, B, C) :-                          /*1 */
   D is B*B - 4*A*C,                     /*2 */
   solve(A, B, D),                       /*3 */
   final.                                /*4 */
                                         /*5*/
solve(_, _, D) :-                        /*6*/
   D < 0,                                /*7*/
   write('No solution'),                 /*8*/
   !.                                    /*9 */
solve(A, B, D) :-                        /*10 */
   D = 0,                                /*11 */
   X is -B/ (2*A),                       /*12*/
   write(' x = '),                       /*13*/
   write(X),                             /*14*/
   !.                                    /*15*/
solve(A, B, D) :-                        /*16*/
   S is sqrt(D),                         /*17*/
   X1 is -B + S / 2*A,                   /*18*/
   X2 is -B  - S / 2*A,                  /*19*/
   write('x1 = '),                       /*20*/
   write(x1),                            /*21*/
   nl,                                   /*22*/
   write(' and x2 = '),                  /*23*/
   write(X2),                            /*24*/
   nl.                                   /*25*/
                                         /*26*/
final:-                                  /*27*/
   write('good bye').                    /*28*/
```

At line 18, the user should insert parenthesis be X1 = (-B + S) / 2*A, and line 19 should be X2 = (-B - S) / 2*A. Because if the user does not do that according to the priority of the operator '/' is higher than '-', program will calculate S/2*A before plus '-B' at line 18. Similarly, program will calculate -S/2*A before minus '-B'.***

Figure 4.8 Implement Weakness Type 3

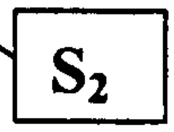
```
run(A, B, C) :-                    /*1 */
D is B*B - 4*A*C,                  /*2 */
solve(A, B, D),                    /*3 */
final.                             /*4 */
                                   /*5*/
solve(_, _, D) :-                  /*6*/
   D < 0,                          /*7*/
   write('No solution'),           /*8*/
   !.                              /*9 */
solve(A, B, D) :-                  /*10 */
   D = 0,                          /*11 */
   X is -B/ (2*A),                 /*12*/
   write(' x = '),                 /*13*/
   write(X),                       /*14*/
   !.                              /*15*/
solve(A, B, D) :-                  /*16*/
   S is sqrt(D),                   /*17*/
   X1 is (-B + S) / (2*A),         /*18*/
   X2 is (-B  - S) / (2*A),        /*19*/
   write('x1 = '),                 /*20*/
   write(x1),                      /*21*/
   nl,                             /*22*/
   write(' and x2 = '),            /*23*/
   write(X2),                      /*24*/
   nl.                             /*25*/
                                   /*26*/
final:-                            /*27*/
   write('good bye').              /*28*/
```



User should add indentation in front of each line of the body of goal between line 2,3 and 4 about 3 or 4 columns.

Figure 4.9 Style Weakness Type 1

```
run(A, B, C) :-                    /*1 */
   D is B*B - 4*A*C,               /*2 */
   solve(A, B, D),                 /*3 */
   final.                          /*4 */
solve(_, _, D) :-                  /*5*/
   D < 0,                          /*6*/
   write('No solution'),           /*7*/
   !.                              /*8 */
solve(A, B, D) :-                  /*9 */
   D = 0,                          /*10 */
   X is -B/ (2*A),                 /*11*/
   write(' x = '),                 /*12*/
   write(X),                       /*13*/
   !.                              /*14*/
solve(A, B, D) :-                  /*15*/
   S is sqrt(D),                   /*16*/
   X1 is (-B + S) / (2*A),         /*17*/
   X2 is (-B  - S) / (2*A),        /*18*/
   write('x1 = '),                 /*19*/
   write(x1),                      /*20*/
   nl,                             /*21*/
   write(' and x2 = '),            /*22*/
   write(X2),                      /*23*/
   nl.                             /*24*/
                                   /*25*/
final:-                            /*26*/
   write('good bye').              /*27*/
```



User should add a blank line after line 4 between goal/ functor/predicate 'run' and goal/functor/predicate 'solve'

Figure 4.10 Style Weakness Type 2

4.6 Pattern of Error/Weakness Message

In order to transfer all attributes for generating a weakness message i.e. contents of feedback messages to students. We use prolog to analyse students' solution then the output from this process will be the input for creating the template for feedback messages from our system to the user. In this case, we use visual basic because it can communicate with prolog. The output from the process of analysing students' solution is considered from these attributes (in brackets) to create the structure of the feedback message explanation, i.e. message (WeakType, StartLine, EndLine, VarName, GoalName, Arity). Each case will appear either as attribute 'VarName' or 'GoalName'. There are two levels of explanation we suggest. Level 1 is provided by our system directly i.e. default by system. Another level can be either the example or choice for TAs to choose in order to annotate to feedback message.

4.6.1 The Pattern of feedback messages for Design Issue

Level 1 : Type D1

Design Issue line [Start line# -- End line #]: warning, [variable name], upper-case letter in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Design Issue between line no. Start line# and End line # : variable name should start with upper-case letter in predicate goal name/ #arg

Figure 4.11 The Pattern of feedback messages for Design Issue Type 1

Level 1 : Type D2

Design Issue line [Start line# -- End line #]: warning, not using [variable name], in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Design Issue between line no. Start line# and End line # : variable name hasn't been used in predicate goal name/ #arg

Figure 4.12 The Pattern of feedback messages for Design Issue Type 2

Level 1 : Type D3

Design Issue line [Start line# -- End line #]: warning, [goal name/ #arg] not reachable

Level 2 : e.g.

Warning : This looks like Design Issue between line no. Start line# and End line # : predicate goal name/ #arg hasn't been reachable

Figure 4.13 The Pattern of feedback messages for Design Issue Type 3

Level 1 : Type D4

Design Issue line [Start line# -- End line #]: warning, does not exist [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Design Issue between line no. Start line# and End line # : the procedure of predicate goal name/ #arg does not exist

Figure 4.14 The Pattern of feedback messages for Design Issue Type 4

4.6.2 The Pattern of feedback messages for Implementation Issue

Level 1 : Type I1

Implementation Issue line [Start line# -- End line #]: warning, missing cut (!) in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Implementation Issue between line no. Start line# and End line # :no cut (!) in predicate goal name/ #arg

Figure 4.15 The Pattern of feedback messages for Implementation Issue Type 1

Level 1 : Type I2

Implementation Issue line [Start line# -- End line #]: warning, change from ';' to ',' in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Implementation Issue between line no. Start line# and End line # :suspect that you should change ';' to ',' in predicate goal name/ #arg

Figure 4.16 The Pattern of feedback messages for Implementation Issue Type 2

Level 1 : Type I3

Implementation Issue line [Start line# -- End line #]: warning, missing parenthesis () in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Implementation Issue between line no. Start line# and End line # :suspect that you forgot parenthesis( ) for operating the priority of operator in predicate goal name/ #arg

Figure 4.17 The Pattern of feedback messages for Implementation Issue Type 3

4.6.3 The Pattern of feedback messages for Style Issue

Level 1 : Type S1

Style Issue line [Start line# -- End line #]: warning, no indentation in [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Style Issue between line no. Start line# and End line # : no indentation in the body of predicate goal name/ #arg

Figure 4.18 The Pattern of feedback messages for Style Issue Type 1

Level 1 : Type S2

Style Issue line [Start line# -- End line #]: warning, no blank line before [goal name/ #arg]

Level 2 : e.g.

Warning : This looks like Style Issue between line no. Start line# and End line # : no blank line before predicate goal name/ #arg

Figure 4.19 The Pattern of feedback message for Style Issue Type 2

With regard to the pattern of feedback messages for Design/Implement/Style Issue as above, the underlined phrases which appear in the pattern of feedback messages in level 2 is the choice for TAs to choose from the system.

4.7 Discussion

Regarding system design in relation to types of weakness messages, our system can give student feedback for each kind of weakness message. If our system finds each type issue (D1, D2, D3, D4, I1, I2, I3, S1, S2, S3) more than once, our

system will have choices for TAs to make e.g. giving feedback only once for one kind of error/weakness type and not providing all the same kind of type issue. To put it simply, good feedback messages on error type should not be provided to follow every error (see Chapter 2). In this manner, our system will have a process of help to give feedback which is scaffolded by the system. So far, classification of students' weakness/error from their solutions is a part of our case study. Yet there are various parts of this case study that need to be taken into account according to Chapter 5 through Chapter 7.

Due to the fact that using either indirect comment or polite comment (Graesser et al., 1995) appears to be useful for improving students' learning on their weaknesses, we believe that teaching TAs to annotate such comments into feedback messages for students on level 2 can help students improve their learning.

Therefore, good assistant systems should help TAs to make good decisions in which it depends on the situation. Good systems should have some choices for TAs to help them to recognise useful detail for feedback to students. Sometimes, giving the answers may not correspond to TAs' thinking, resulting in thought conflict for TA's. This may be an appropriate answer for system to give TAs e.g. "This looks like a style problem…..". As TAs are adults, if they do agree with the systems suggestion it will be ok. Therefore, our system can consider what the TAs can work with as well as using andragogical approaches (see Chapter 5). Furthermore, they can develop their skill for giving quality feedback by the approach of cognitive apprenticeship (see Chapter 5) in a real world situation of marking assignments in which real time learning can help TAs to learn well. These approaches can help TAs to agree with our systems; however, the worst case scenario is TAs may give comments by hand because our system does not provide all kinds of weaknesses on prolog programming issues. Due to the fact that our system also focuses on teaching TAs to give feedback, classification for all such kinds of weaknesses will be out of the limit of this thesis. This work does not give all details of weakness; nonetheless, we aim to show some kinds of weaknesses in order to set the examples for TAs to learn to give quality feedback to students in which this is the main task of this system. Even though our categorisation provides only a few examples, in fact, there are many more of them

in which this is only a part of the thesis in order to adequately test the aim of the research, yet other parts relate to helping novice teachers to learn to give feedback (see Chapter 5) e.g. if students did something wrong in any problem issue in which the same classification type is repeated many times, the system will not repeat the same information in the messages. In addition, it will tell TAs to decide feedback for students e.g. "you might care to know this is not a suitable style and it's in over 100 places ...what would you like to do" and the system gives a choice for TAs or "you are misunderstanding using ... according to implementation issue" or "TA, here is an example of a design issue ...singleton variable...". Currently our system has a template for TAs for giving feedback. We should also prepare lists of comments for the TAs to copy in the feedback-report. Providing module for summarising many errors when students make an error in any way could help the TA. Furthermore, our system should allow TAs to tailor messages –manage feedback message- in which they associate the error/weakness according to their decision. However, the system should not leave TAs to spend much time to edit feedback messages on a number of errors. This research followed the basic idea of "make error message or statement to help TAs first". In addition, our system also provides the choice for TAs to recognise students who have made an error according to our classification (design, implement, style)

In this thesis, we do not include the problem of understanding issues in classification, mainly, because most of those problems have to be decided before managing the assignment context, i.e., we try to avoid assignment context as much as possible.

4.8. Summary

To sum up, providing a framework with a domain topic to analyse student performance would have two benefits:

1) Help the TA to structure the domain of feedback

2) Help the student to structure the nature of programming

The advantages might be assumed to hold for all TAs but not every lecturer. According to our analysis of students' weaknesses, we obtain three types of error

messages which are: design error, implement error, and style error. For each error type we present the pattern of providing error messages in order to be an example of system design of how to help people learn to provide quality feedback in the rest chapter, therefore the next chapter we will take feedback design (see Chapter 3) off and use the domain of our system in this chapter to be part of design of the scaffolding system to help the TAs learn to provide quality feedback.

Chapter 5

Scaffolding System Design

5.1 Introduction

In the previous chapter we described the feedback design and this design has now been adopted. In this chapter we investigate several scaffolding systems (e.g. PACT Geometry tutor (Aleven & Koedinger, 2000; Aleven & Koedinger, 2001); Ecolab (Luckin & du Boulay, 1999); SCI-WISE (White et al., 1999), etc.) and adopt a number of approaches to design a scaffolding support system to help the TA to give good feedback – called McFeSPA (Metacognitive Feedback Scaffolding system for Pedagogical Apprenticeship). The system design is founded on several interesting ideas based on topics such as scaffolding, andragogy, feedback patterns, giving quality feedback, and cognitive apprenticeship. These notions are adapted to establish McFeSPA's Architecture. In particular, this architecture provides for metacognitive feedback because the system will be designed to provide prompts, hints, and help messages in order to encourage the TA think about how to give good feedback. This chapter also presents: an investigation of help design in scaffolding systems; scaffolding design in McFeSPA, design of McFeSPA's behaviour, McFeSPA's architecture, principles of McFeSPA, context of hints, andragogical model for training TAs to improve giving feedback, and contextual design. Some basics ideas are shown in the following.

5.2 Help Design in Scaffolding systems

In order to support the TA to provide good feedback to students, we reviewed various scaffolding approaches and analysed well-known scaffolding systems to capture and adapt the idea in each scaffolding system to be applied to McFeSPA.

5.2.1 Scaffolding Approach

Scaffolding is described by Wood, Bruner, and Ross (1976) as "...controlling those elements of the task that are initially beyond the learners capability thus permitting him to concentrate upon and complete only those elements that are within his range of competence" (p 9). Teachers are responsible for leading the learners toward understanding of the task and helping them develop their own conception of the task. This can be done by creating a balance between supporting and challenging the user. Support can be provided through scaffolding and challenge is provided through the learner's interest in completing the task. Learners are given opportunities to act like they know how to complete a task before they actually do. Scaffolding and challenge need to be presented holistically and in a context that signals value and usefulness. In the Zone of Proximal Development (ZPD), scaffolding is assistance which is a major component of the teaching activity (Bruner, 1984). Scaffolding is an instructional tool that reduces ambiguity, thereby increasing growth opportunities (Doyle, 1986). Scaffolding developed to assist learners internalize information best occurs in learning situations where the learners have opportunities to communicate their thoughts, according to Roehler, and Cantlon (1996). From their studies, they proposed the following types of scaffolding that can be used in the classroom:

- Offering explanations
- Inviting student participation
- Verifying and clarifying student understandings
- Modeling of desired behavior
- Inviting students to contribute clues.

These are temporary supports which are gradually removed by the teachers. It is expected that these forms of scaffolding will be helpful to employ in the design of a scaffolding system.

In terms of Scaffolding in the classroom, Roehler and Cantlon (1996) observed that in the first stage the teacher or knowledgeable tutor controls and guides the learners' activities. Then both teacher and learners alternate to lead the conversation. Thereafter, the teachers continue to guide the learners emerging understandings, providing assistance depending upon the learners' requirement.

Finally, the teachers remove all assistance in order to allow the learners to take a full range of responsibility. In order to apply this, a scaffolding system is like a teacher and the scaffolding system's user is like a learner –referred to as a TA in this thesis.

Recently scaffolding designed for a learning environment has mainly acted as a support for users to assist them as they work toward a learning goal. Types of scaffolding as discussed above might be useful for classroom learning, and some types could be useful to apply for scaffolding the TAs learning process to give quality feedback in the situation of marking an assignment. Not only can successful scaffolding assist the learner by offering appropriate help, but also it is possible to withdraw/fade the support when the learner is ready (Luckin & Hammerton, 2002). In addition, Jackson, Krajcik, & Soloway (1998) categorised scaffolding in three ways: supportive scaffolding; reflective scaffolding; and intrinsic scaffolding which is implemented in TheoryBuilder together with fading. They revealed that the question of how to design scaffolding is still open even though some combination of adaptive and adaptable fading appears to be the most appropriate direction for future research. Randoll & Kali (2002) also categorized scaffolds with regards to how and where scaffolds are used. Scaffolds can be categorised into "types" and "interfaces". Types of scaffolds answer the question "what does the scaffold help to do?" –which are functional, process, content, metacognitive, interpersonal, and scaffold. Interfaces answer the question "how is the scaffold presented to the student?" –which are stable, adaptive, and adaptable.

In order to apply and integrate types of scaffolding in our context, the work was established on a combination of Jackson et al.'s (1998) approaches and Randoll & Kali (2002) approaches which correspond with types of scaffolding in electronic performance support systems (Cagiltay, 2006). The types of scaffolding, scaffolding-interface, and scaffold fading are presented in Section 5.3.3.

5.2.2 Analysis of scaffolding systems

The idea of scaffolding is commonly applied to several artifacts such as a learning environment. Software scaffolding has been successfully employed within educational technology to help the learner whether they require assistance or not. Recently, scaffolding has begun to be incorporated as a learning support tool in educational software (see the subsections of this section). There is evidence that educational software with extensive scaffolding is more educationally effective than software without support (e.g. Ecolab (Luckin & du Boulay, 1999), TheoryBuilder (Jackson et al., 1998)). Several systems were designed to support students to learn to use the system to achieve knowledge in the specific domain. These ITSs emphasise help-seeking and help the learners learn but none of these systems offer assistance in giving feedback. Each system employed different approaches and theories e.g. theory of cognition and learning, theory of contingent tutoring, Vygotsky's instructional theory, etc. In this section we analyse scaffolding systems with a view to adapting the more relevant approaches in order to design McFeSPA.

5.2.2.1 Cognitive tutors

The first cognitive tutor (Anderson et al., 1995) was designed in the domain of algebra based on ACT-R (Atomic Components of Though-Rational), theory of cognition and learning, and productive rule (learn from example), and the level of help was controlled by the learners (help-on demand). The level of help messages consists of

- problem solving goal (a rule contributes)
- important features of the problem solving context
- bottom-out hint (specific action to be taken) and the help messages that are mostly error feedback messages.

This system allows the learners control over help requests and self-explanation of correct response. The limitation of the ACT-R theory is "help on demand", decisions made by learners leading to difficulties in providing help (e.g. distinguish errors, missing knowledge, misconceptions) and students requesting too much help. Even though this system helps students learn algebra – rather than

help adult learners to give help, its approach is interesting to consider in terms of the level of help and self-explanation through the use of a glossary.

5.2.2.2 Contingent tutoring systems

EXPLAIN (EXperiments in PLanning And INstruction) (Wood et al., 1992), and QUADRATIC tutor (Wood & Wood, 1999) were developed using the theory of contingent tutoring (which is rather consistent with the ACT-R theory). EXPLAIN's domain is the tower of Nottingham while the domain of the QUADRATIC tutor is algebra and solving quadratic equations. These systems provide contingent help according to the learner's need (interaction between tutor and learner). The tutor adapts help until students progress to the next step and reduce help when the students succeed. There are five levels of contingent help

(1) problem (feedback),
(2) quick vague cue,
(3) less vague cue,
(4) almost the answer
(5) the answer.

The limitation of the ACT-R theory and contingent tutoring is "help on demand". In contingent tutoring the tutor decides the level of help while in cognitive tutors the learner makes the decisions. According to Wood's (2001) findings from QUADRATIC the relation between domain knowledge, help seeking and time on task should be taken into account when developing tutoring systems to help the learner to learn. Wood tries to study self-regulation, help seeking and learning use for computer-based tutoring systems in order to develop the principle of designing tools to support learning. He stated that computer-based environments were designed to

- determine a useful method for use with computer-based-models of tutoring
- give the means of exploring the limitation of contingency theory.

He argued that currently tutors cannot offer any guidance that is contingent upon them. As a consequence, he affirmed that there are limitations that apply to contingency theories for adaptive learning environments. Even though

contingency theories have limits for adaptive learning environments in helping students to learn specific domain knowledge, this theory is worth considering in terms of the level of help required to in helping adults to learn. It could be useful to adopt a contingent help strategy that offers guidance depending on the TAs' performance coupled with a record of previous help from the system to the TAs as adaptive help.

5.2.2.3 PACT Geometry tutor

The PACT Geometry tutor (Aleven & Koedinger, 2001) was designed for the domain of geometry. Help in PACT Geometry Tutor consists of intelligent help and unintelligent help. Intelligent help provides a hint button that the learner can use to request help when he/she is confronted with two or more errors for each step. This help is in the form of on-demand hints such as adaptable support that the user can use to ask for help when he/she requires it. Unintelligent help is a low cost source of help that the user can request when he/she is confronted with either one error or none in each step. If the learner requests a hint before the appropriate level of required error is reached, a reminder message appears that suggests that the user solves the problem by using the glossary – this can be referred to as a metacognitive scaffold.

With respect to a rational help-seeking strategy, tutors should help the learners learn to develop effective strategies for metacognitive help-seeking (e.g. Wood & Wood (1999); Recker & Pirolli (1992); Conati & VanLehn (1999)). According to the PACT Geometry Tutor, the learners rarely use the Glossary, and this leads to them making a number of errors. In order to help the learner develop their metacognitive skill, Aleven & Koedinger (2000) suggested that the learner should ask for a Glossary (lower-cost source of help) before requesting hint (high cost one). If not, the learner does not master the desired metacognitive strategy.

Later, in studying the relation between help seeking, prior knowledge and learning outcome, Aleven & Koedinger (2001) found that the theory could be influenced between system control and student control in order to develop a tutoring system with an improved dynamic balance. The result is that the influence of help-seeking on learning did not change and the student relied on their prior knowledge. This result contrasted with Wood & Wood's (1999) result.

Aleven & Koedinger explored myriad reasons of why it is difficult to design a system to help in the appropriate time in order to assess whether students' help requests are appropriate. For example, it is not easy to determine when a particular step in a particular problem at a particular time has gone too far beyond a particular student's ability.

The study of the relationship between help seeking and learning in order to find what influences students' help-seeking behavior in using PACT Geometry is similar to QUADRATIC even though PACT Geometry's domain knowledge is small scale compared to QUADRATIC's. The PACT Geometry Tutor is a tutor that provides a context-sensitive hint between 5-8 levels of hints for each step. Each level of hints depends on the learner's skill. As more hints are given, so the advice will become more specific. When the learner requests a hint at first, the tutor starts at the first (most general) level. It displays the next level when the student makes a help-request. The PACT Geometry Tutor uses a cognitive model that represents the targeted skills. Model-tracing is used to analyse the student's activities (Anderson et al., 1995). The tutor also maintains a student model, estimating the probability that the student knows each crucial skill in the model. The model is updated, through a process called knowledge-tracing, once for each step. The learner can view the estimates of skill mastery from the tutor in the skill meter window in the form of a bar chart.

Aleven & Koedinger (2001) use their help-seeking behavior model which contains more steps than Wood & Wood's (1999) model. Aleven & Koedinger's (2001) findings are consistent with those reported by Wood & Wood (1999), even if not entirely the same, as they concluded that ITSs often provide feedback on-demand. In addition they also proposed several open-ended questions, i.e. how good are students at making optimal use of such help facilities? how does students' help-seeking behavior relate to the learning outcome? and how does students' prior knowledge influence the relationship between help-seeking and learning? Their current studies address these questions, and their results could be used to design tutoring systems. In addition, they have an open question: how can their results help students help themselves to find a better balance between system control and student control? It is difficult to specify the requirement of help-seeking behavior (e.g. by quantitative prediction) so they still have to ask the

question: what is a crucial theory required in the study of help-seeking behavior? There have also been other open questions "What is the relation between tendency to help-seeking and learning outcome that Wood & Wood found (in the low prior knowledge group) and their findings from the high prior knowledge group?" Psychometric theories (Hambleton & Swaminathan, 1985) (e.g. item response theory) can provide a metric on how "advanced" a student is likely to be and how difficult a particular step (item) is going to be, but this does not deal with time. Thus, it is very difficult to attempt to make progress on help-seeking which is an important element to all kinds of learning and in all domains.

Although the PACT Geometry Tutor tries to help students learn and seek help with the system, the system does not help adult learners to give feedback. However, their approaches to intelligent and unintelligent help are worth considering in terms of helping adults learn to give improved feedback.

5.2.2.4 Ecolab

Ecolab (Luckin & du Boulay, 1999) is a software design framework which was implemented in a number of versions. The evaluation of the design framework is effective in helping a single user to learn about the relationships which exist within a food web. Three variations on the Ecolab theme were implemented: the Vygotskian Instruction System (VIS), Woodsian Inspired system (WIS), No Instruction-intervention system (NIS). Ecolab was designed to evaluate Vygotsky's instructional theory in software design. This framework is broader than the contingent tutoring framework, with a broader range of task selection strategies, and a broader range of strategies for dividing the responsibility for controlling the level of hints and issues connected with student modeling. The scaffolding approaches contain graded help (5 levels of help) which provide specific help for a particular situation and activity differentiation (3 levels of hints) which provides specific hints to support the learner's activity. This research does not discuss the role of proactive help seeking on the part of the learner. The results of the three systems (VIS, WIS, NIS) are quite similar. Due to the small number of subjects in the evaluation, further future research in the WIS system is required with a larger evaluation to determine what and how to process. Even though the design framework was effective in helping a single user in the domain

of food web ecology, the system does not help the learner to give help. However, graded help approaches appear useful if they are adapted in the design of a scaffolding system to help TAs learn to give feedback to students.

5.2.2.5 SE Coach

SE Coach (Conati & VanLehn, 1999, 2000) is a software design framework which was implemented and evaluated in the domain of learning from examples through self-explanation within Andes, a tutoring system that helps students learn Newtonian physics through both example studying and problem solving. The framework includes solutions to three main problems in ITSs:

1) Designing an interface that effectively monitors and supports self-explanation i.e. the menu-based interface that monitors student's attention and provides structured prompting and scaffolding for self-explanation. The SE-Coach's menu-based tools allow students to justify a solution step by describing the domain rule from which the step derives, in terms of

- the preconditions that must be verified to apply the rule;
- the results that the rule application generates.

This description in terms of preconditions and consequences reflects the SE-Coach's rule-based domain representation, and allows the system to provide feedback for correctness based on this representation. The SE-Coach provides incremental support to self-explanation through three different levels of scaffolding. The first level of scaffolding in the SE-Coach's interface is provided by a masking mechanism that presents different parts of the example covered by grey boxes, each corresponding to a "unit" of information. The second level is provided by the SE-Coach's interface through specific prompts to self-explain. The third level consists of menu-based tools that provide constructive but controllable ways to generate the desired self-explanations.

2) Designing a student model that allows the assessment of example understanding from reading and self-explanation actions. The model of correct self-explanation, which is the core structure of the SE-Coach's expertise, encodes the knowledge to provide feedback on student's self-explanations. This is used in the student model to assess how the students' self-explanations reflect example understanding and it guides the SE-Coach's tutorial interventions. The SE-Coach

student model relies on the Bayesian network framework for probabilistic reasoning.

3) Eliciting further self-explanation that improves student's example understanding.

An Evaluation study of the SE-Coach regarding the effectiveness of support for self-explanation during example studying indicates that structured scaffolding of self-explanation can be more beneficial at early learning stages. Thus as students become more proficient in the subject matter even simpler forms of prompting can successfully trigger self-explanation. However the empirical evaluations are fundamental for the development of instruction systems of real effectiveness that were used in the system.

Nevertheless, even though the design framework was effective in helping a single user in the domain of learning from example through self-explanation, the system does not help the learner to give help. However, providing a different level of scaffolding appears useful if adapted in the design of a scaffolding system to help TAs learn to give feedback to students.

5.2.2.6 SCI-WISE

SCI-WISE is a tool for fostering students' metacognitive development. This system combines cognitive and social aspects of cognition within a social framework that takes the form of a community of advisors who work together to guide and support reflective inquiry. SCI-WISE allows students to create and represent types of expertise as they work to improve the advisors. According to the investigation of the types of social interaction and activities which involve the finding of support the view of social process as well as cognitive process, White, Shimoda and Frederiksen (1999) considered three views of metacognition. These are

- "knowledge about knowledge", including knowledge of the form and content of cognitive and social expertise and when and why such expertise is useful;
- "regulatory skills", including skills needed to employ sociocognitive expertise e.g. planning and monitoring skill; and
- "development expertise", including the ability to reflect on

sociocognitive knowledge and its use to determine how to modify and improve both of these.

They have proposed the categories of beliefs for the hypothesizer. An interesting approach to apply in McFeSPA is to include beliefs about

- task context and status;
- prior interactions with the agent;
- users' history and characteristics;
- users' goals and desires;
- agent's own goals and priorities.

Help in SCI-WISE contains helping students learn about the advisor's expertise; helping students understand the nature of the task being undertaken; helping students get the task done; helping students develop widely applicable cognitive and social skills; helping students learn how to assess, reflect on, and improve their inquiry processes. They found that metacognitive Reflective-Assessment Process can help low-achieving students improve their metacognitive skills including understanding the purpose and steps of the Inquiry Cycle, i.e. "learning how to learn". However, the existing system is limited to supporting work on tasks related to scientific inquiry and SCI-WISE would be annoying and confusing to the learners in that there are too many agents who are indistinguishable from one another.

They suggested that there should be a pedagogical advisor who can provide the theories of learning and coaching – i.e. pedagogue could present users with pedagogical principles like "give less and less advice each time so that users learn how to do the task without help" or "only give advice when user say they want it otherwise they may get annoyed at being told what to do all the time." These ideas are associated with scaffolding and fading (by scaffolding first then gradually fading). In order to support SCI-WISE for reifying and testing its theories, they also suggested that students and their teachers could collaborate with educational researchers to address some of the difficult issues related to the nature of lifelong learning skills and the design of effective learning environments.

Nevertheless, even though the design framework was effective in helping a single user in the domain of a Physics project, the system does not help adult learners to give help. However, this system leads us to consider how to help the TA think about the knowledge of giving quality feedback then monitoring their expertise of how their learning of giving quality feedback improved. The aim is to help the TA learn how to reflect on and improve their feedback giving skill process.

5.2.2.7 TheoryBuilder

TheoryBuilder (Jackson et al., 1998) is designed to use an approach called Guided Learner Adaptable Scaffolding (GLAS) in the domain of building a reasonable scientific model by creating relationships between selected objects. The system has two goals:

- providing support by program;
- design and implement scaffolding in interactive learning environments (ILEs) by supporting fading by the learner's understanding and improved abilities.

The learners can control the fade scaffolding by with guidance from the system. There are three kinds of scaffolding in TheoryBuilder: supportive scaffold (guiding through subtasks, coaching and modeling throughout the software, proving context-sensitive help and examples); reflective scaffold (eliciting articulation with forms and prompt for planning, explanation, evaluating); intrinsic scaffold (multiple linked representation (from simple to advance), hiding complexity but providing advanced options. GLAS was applied to TheoryBuilder which was implemented with a tool that the researchers developed to support the complex and educationally valuable task of scientific modeling. It supports fading for each of its scaffolds under the learner's controls.

Although the analysis of the data indicated the success of the GLAS approach in developing an adaptable tool to support the diverse and changing needs of learners, the interface presents a lot of options that will initially confuse many users who are new to the task. The "Stop-reminding me" button will be turned-on every time the user starts the project (system) because the system does

not remember the user's preferences. The researchers suggested that allowing users to choose the options they want to use will help them tailor the interface in ways that make more sense to them.

Despite the system helping the learners learn to create a scientific model, it does not help to give feedback. However, its use of different types of scaffolding is worth adopting for the design of McFeSPA.

5.2.3 Discussion of help design in scaffolding systems

A number of research projects are concerned with 'on-demand help' and 'system-initiated feedback' (Aleven et al., 2003). This is crucial to the issue of how learners' interaction with ILEs is associated with theories of learning or tutoring, according to Aleven and colleagues (2003). Furthermore, Renkl (2002) suggested that giving help in ILEs should emphasize communicating domain principles. A great deal of research has underlined the use of a multilevel (or progressive) help framework; however, on-demand help-systems differ as to whether the learners or the system control the level of help. Beyond this the help information provided by ILEs is inadequate to help the learner to decide whether to request the help. Aleven et al. (2003) examined help-seeking frameworks that do not include certain concepts in psychology for explaining and developing the learners activities associated with the help given. They reported that, in a number of studies various help procedures could allow waiting time before giving help, asking questions of varying difficulty and complexity, selecting respondents and reacting to correct and incorrect responses; however help relies on various factors such as achievement level, problem, grade level. Many of the studies found that students could use help effectively; however they rarely chose to use help (Aleven et al., 2003). There was clear evidence that help seeking behaviors are not effective in Geometry cognitive tutors, according to Aleven and Koedinger (2000). They have analysed that most students need the last (final) hints i.e. the students do not read much of the explanation given by the hint. Aleven et al. (2003) suggested that giving help on demand results in better learning in specific principles based on examples. Schworm and Renkl (2002) found that prompting for self-explanations supports better learning than provision of instructional explanation. Nevertheless, Aleven et al. (2003) revealed that providing on-

demand help is always ineffective resulting in a decrease of the self-explanation activity and a decrease of learning outcomes. They reported that the evidence is inadequate to verify how the detail of help systems and task selection influence help seeking. For this reason, they proposed that the factors for use by on-demand help lead to learning results, as shown by Figure 5.1. The author developed this figure as ontology based on the important and interesting keyword relating in the factors of the use of on-demand help and then linked together "by hand" based on Aleven et al. (2003). Very few studies investigated the best types of help for individual learners. A question then needs to be asked concerning: 'What kind of help is most effective?' In this sense, their studies are associated with the number of dimensions of the help systems and the relevance to the learning outcome. In the studies of Luckin and du Boulay (1999), this involved a number of help levels; point of students controls of the levels and parameters of the nature and of the content of the learners tasks. The result could not conclude how differences in learning outcomes are related to the differences in the help system. Even though the systems we reviewed emphasise help-seeking for high school students- rather than for adults, it has proved useful to study the design of help system influence on learning, help-seeking behaviors (Aleven et al., 2003).

According to Aleven et al. (2003), giving help may mean

- finding appropriate ways to diagnose students' misconception in the learning process;
- identifying specific instructional intervention (guide from misconception to correct model);
- defining possible or frequently occurring incorrect models or misconceptions in the learning process; and
- defining the knowledge base (on-demand).

They also concluded that

- different instructional goals result in different types of ILEs whose help systems provide different types of information;
- learners often use help systems ineffectively or ignore them altogether. However, when they do use help, learning processes and outcomes may be substantially improved;

- a variety of learner characteristics influence help seeking, individually or in combination;
- different types of help may cause different types of help-seeking activities and result in different learning outcomes;
- design -and learner- related factors interact in their effect on help seeking and learning;
    - depending on the learning context, the same type of help may trigger different help-seeking behavior, which in turn is related to different effects on learning outcomes.

When students make multiple errors on a step, they suggested that ILEs could volunteer help depending on the balance between system-initiated help and feedback, and students-initiated help. Different types of learners need different type of help.

Thus, design of McFeSPA should consider several parameters (variables) varying in levels of help messages, according to the learner action from using the system. As we know so far, ITSs support help-seeking (providing on-demand help) but do not help adult learners to give feedback. In addition, there is little research (empirical studies) of effective help seeking and learning with ILEs and also no theoretical framework describing the useful design and implementation in ILEs from a social context, or computer science context, according to Aleven and colleague (2003). Therefore, it is useful to carry out empirical work on the design of a help system in helping TAs to give feedback by adapting the relevant approaches to be applied in McFeSPA.

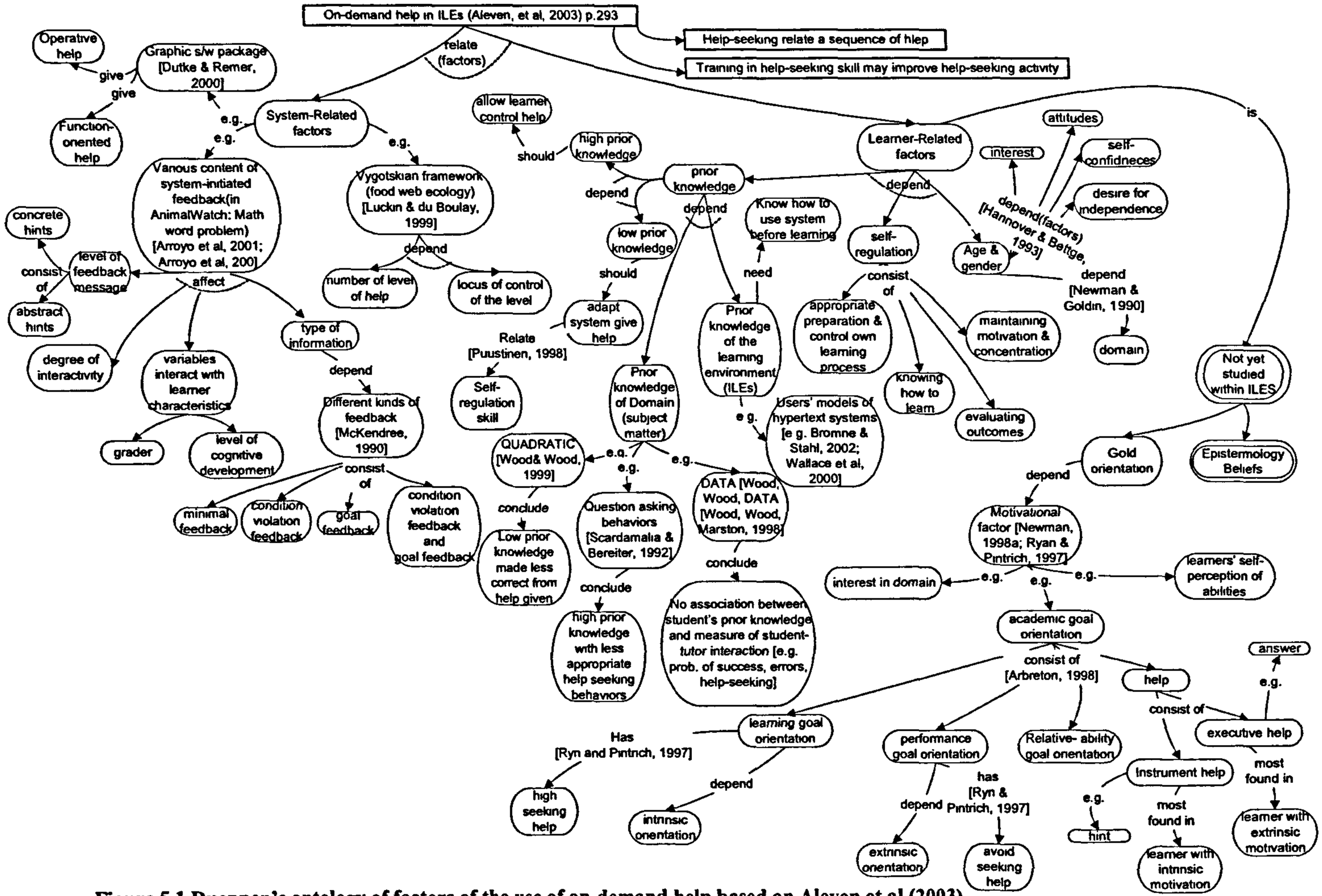


Figure 5.1 Duenpen's ontology of factors of the use of on-demand help based on Aleven et al.(2003)

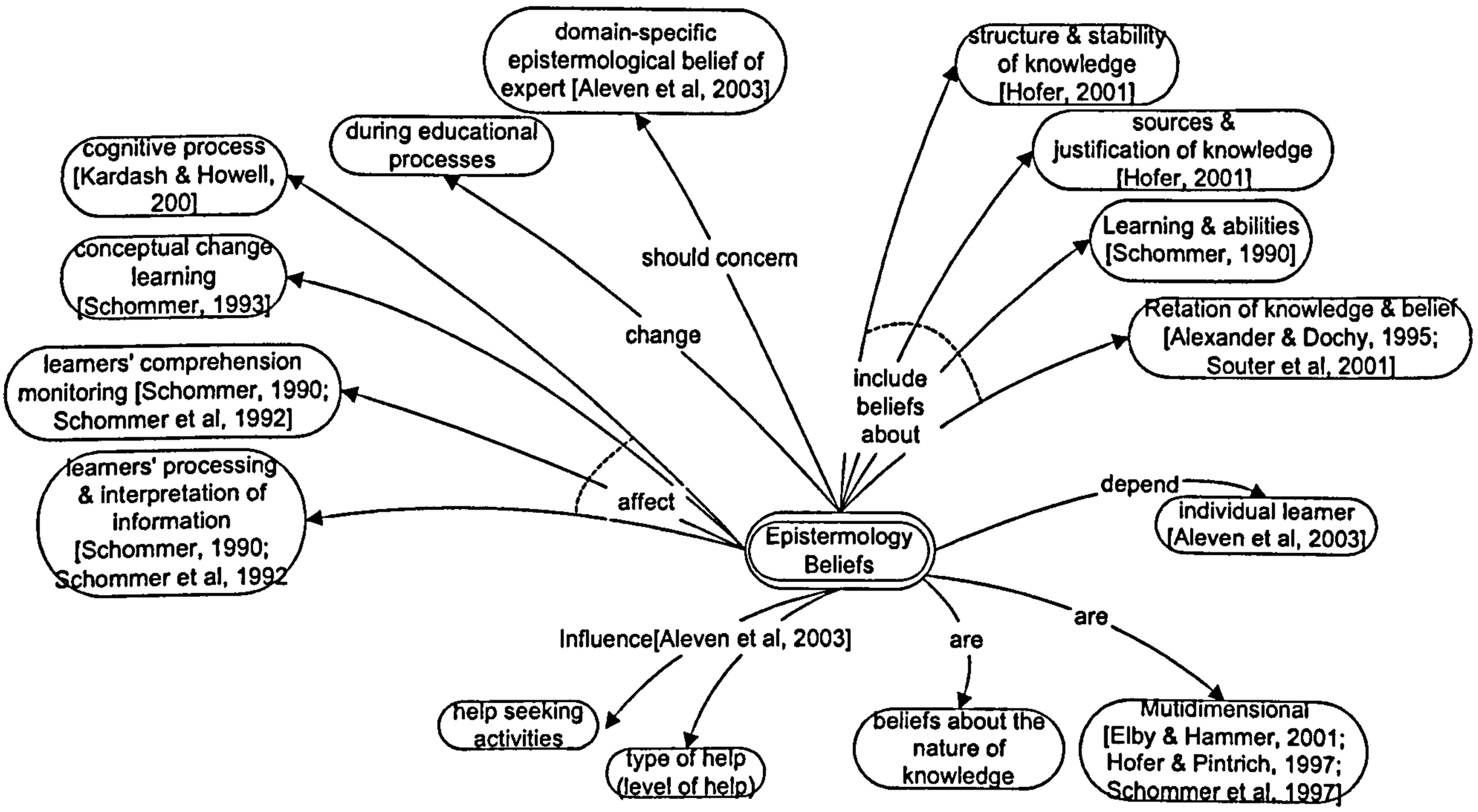


Figure 5.1 Duenpen's ontology of factors of the use of on-demand help based on Aleven et al.(2003) (contd.)

5.3 Scaffolding Design in McFeSPA

To design scaffolding in McFeSPA, we present the context of the thesis, according to Figure 5.2. The context involves employing a scaffolding framework to help the teaching assistants (TAs) improve the quality of their feedback giving skills in the actual environment of marking a programming assignment. The context relates between a semi-automated marking system and the TA directly, and with an indirect relationship between the student and the system. The system design is based on several approaches e.g. design of framework by ITS component (Applications of AI in Education: Beck, Stern, Haugsjaa, 2002); system interface by scenario-based design (Carroll, 1995) and HCI design (Benyon, and Imaz, 1999); scaffolding approaches; andragogical approaches; feedback patterns; giving quality feedback; and cognitive apprenticeship adopted to establish the McFeSPA's Architecture.

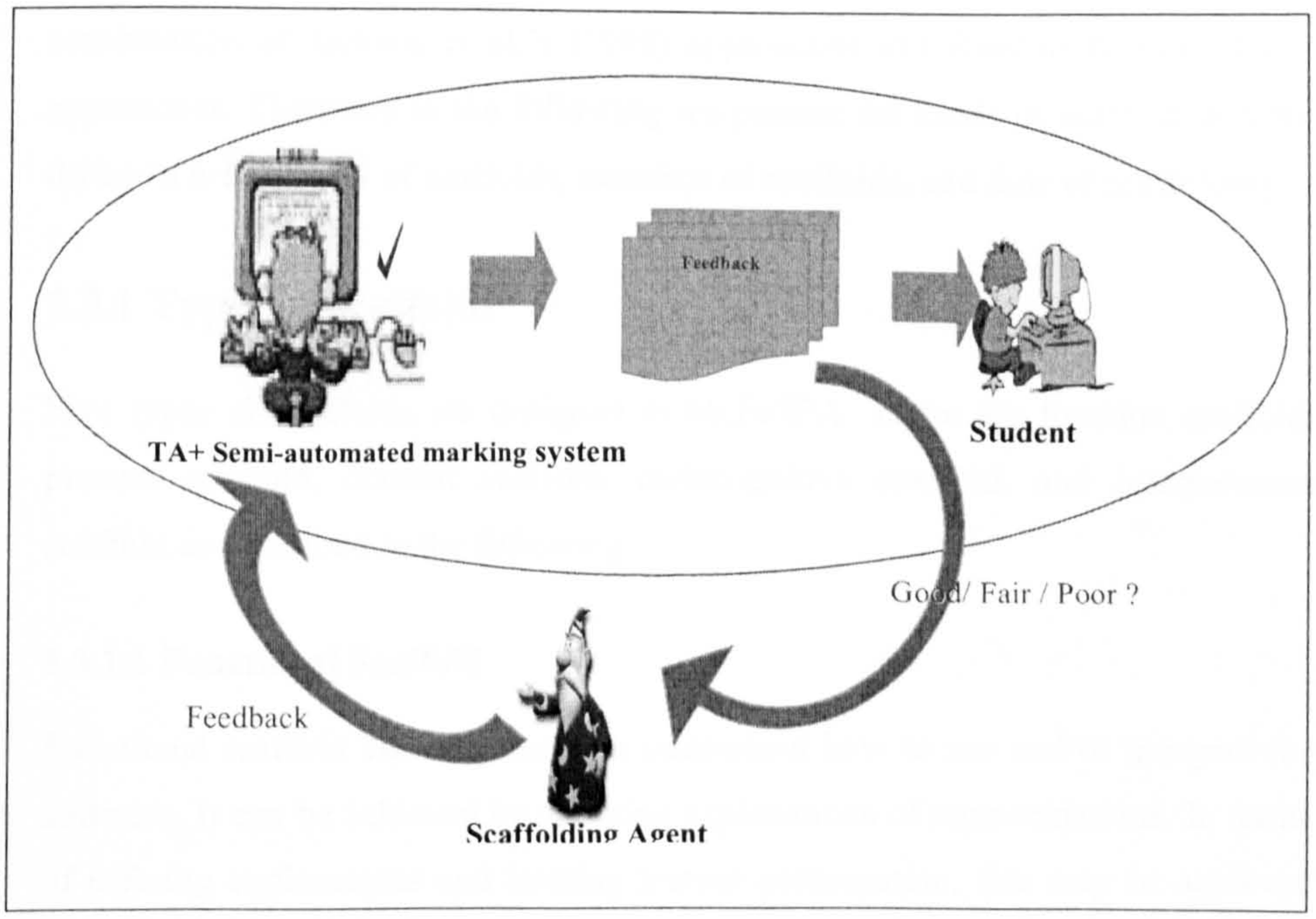


Figure 5.2 Context of the thesis

A number of researchers are concerned with 'on-demand help' and 'system-initiated feedback' (Aleven et al., 2003). A great deal of research emphasises multi-level help frameworks which are different to on-demand help i.e. whether the learners or the system control the level of help. Aleven et al. (2003) have suggested that giving help on demand results in students' learning better. There are a number of measurements of help/scaffold systems studied that have different kinds of effective help. However, the results could not conclude how the differences in the learning outcomes relate to the differences in the help system (Aleven et al., 2003). Most researchers report that learners achieve a better learning outcome in a scaffolding environment (e.g. Ecolab (Luckin & du Boulay, 1999), EXPLAIN (Wood et al., 1992), SCI-WISE (White et al., 1999), PACT Geometry tutor (Aleven & Koedinger, 2000) , TheoryBuilder). However, fading in scaffolding can still be problematic as reported by Jackson and colleagues (1998). There is little research, if any, in empirical studies of the implementation of different help/scaffold in the ILEs in a computer science context. Thus, design of a help/scaffold system for marking programming assignments and learning to give feedback to students could help the feedback giver improve.

According to scaffolding approaches in Section 5.2.1, we discussed a combination of Jackson et al.'s (1998) approaches and Randoll & Kali (2002) approaches. Therefore in the following we present the kinds of scaffold in three different areas, types of scaffolds, interface of scaffolds, and fade of scaffolding:

5.3.1 Types of Scaffolds

Five types of scaffolds are designed in McFeSPA. These are function scaffold, process scaffold, content scaffold, metacognitive scaffold, and interpersonal scaffold as described in the following.

5.3.1.1 Functional Scaffold

Functional scaffold can help learners understand how to use and/or interpret the software. It can be achieved by applying explanations of representations. In terms of offering explanations and inviting learner participation, this may be achieved by way of explanation and clarification to the TAs regarding how to use the tool. Functional scaffold in McFeSPA can be applied as an explanation of each object represented to each interface.

5.3.1.2 Process Scaffold

Process scaffold can help learners understand his/her path within the software in which it is associated with supportive scaffold (Jackson et al., 1998). Supportive scaffolding is support to help the learner do the task, rather like the scaffolding process (Randoll & Kali, 2002). Process scaffold can be done by sequencing and history of the TA's previous actions applied in McFeSPA. Process scaffold can provide various functionalities, including the following activities:

- Reading student's solution
- Analysing student's solution consisting of analysis of all error types, analyzing for particular error type i.e. design/implementation/style
- Generating error messages via default messages in the case of repeated types of error messages
- Generating feedback reports consisting of selecting a feedback template (offering six choices of template in which there will be only one

appropriate feedback template per feedback pattern) and generating final feedback report (organising feedback after the correct feedback template has been chosen then allowing the TA to add any part of student's script according to the TA's need to temporary report or the TA can provide the correct answer according to the incorrect part of student's script.

- Sending electronic feedback report to the student.

5.3.1.3 Content Scaffold

Content scaffold can help the learner figure out an answer. Hints are the contents that could guide the TAs to process the correct path. An example of content scaffold to appear in the help message for each level of help can be seen in Figure 5.4.

5.3.1.4 Metacognitive Scaffold

Metacognitive scaffold could help TA's rethink decisions, and it is therefore associated reflective scaffolding (Jackson et al., 1998). Reflective scaffolding is support for thinking about the task. This type of scaffold can help the learner to be aware of his/her own learning through reflection, monitoring, etc. For example, assessment of understanding "Do I know more/understand better now?", progress through the learning process. In addition, metacognitive process can develop lifelong learning skills (White et al., 1999). In a tutoring system, this type of scaffold could assess the learner's metacognitive process in terms of skill meter's measurement. The design of metacognitive scaffold in McFeSPA is adopted from PACT Geometry Tutor in which the hints are designed to encourage a general metacognitive strategy e.g. "when you do not know something, use an available resource, such as the Glossary, to look it up. Look at what kind of problem you are dealing with, and then look at the Glossary rules that are relevant to that kind of problem (p. 294)". Metacognitive approach in SCI-WISE inspires us to consider how to help the TAs think about improving their knowledge of providing good feedback. The aim is to help the TAs learn how to reflect on and improve their feedback giving skill process. This could be done by prompting the user to add/delete/update an error message or give further positive feedback via the window provided in order to reflect on the understanding of the user for

classification of error types (design/implement/style) and positive feedback (e.g. quality feedback such as feedback sandwiches). In addition, The TA can view the estimates of skill mastery from the system in the skill meter window in the form of a bar chart so the measurement of the skill meter for giving feedback can be employed in McFeSPA to help the TAs to reflect on their actions as Metacognitive scaffold.

5.3.1.5 Interpersonal Scaffold

Interpersonal scaffold can help facilitate social interaction. In particular, turn taking i.e. TAs can take turns exchanging messages with the system, in addition, interpersonal scaffold as intrinsic scaffolding for support that changes the task itself by reducing the complexity of the task and focusing the TA's attention (Jackson et al., 1998) i.e. intrinsic scaffolding provides normal and advanced options for the user. In other words, this support includes providing options to help the TA make decisions in the preliminary stage. If s/he ignores to choose the provided options, this would be called fading. Employing interpersonal scaffold in McFeSPA as a dialogue offer option could help the TA to request or ignore help generated by the system.

5.3.2 Interfaces of Scaffolds

Scaffold/help interface can be presented in different forms, such as, text, graphics, and sound. A scaffold interface can be unchangeable and presented at all times (stable) in order to help the learner in self-explanation, or it can be changed in two general ways: adaptable and adaptive (Jackson et al., 1998). Stable scaffold in McFeSPA can be achieved by employing a glossary. By providing a glossary, McFeSPA allows the learner control of self-explanation. Design of stable, adaptive and adaptable help in McFeSPA, adapted from PACT Geometry Tutor (Aleven & Koedinger, 2000), consists of intelligent help and unintelligent help. In McFeSPA, offering any guidance depends on the TA's performance and the record of previous help from the system to the TA as adaptive help but s/he can accept help depending on their need for adaptable help. The kind of help in McFeSPA, is different from Help in PACT Geometry tutor that offers the hint button to the learners to control their need of help.

Intelligent help is the hint message that will pop up according to the TA's skill step regarding an inappropriate answer. The specific hint advice increases when the TA's action does not follow the correct step. The system starts at the first level and will display the next level even when the TA is still not on the right path. When the TA succeeds each step, the skill measurement of a particular feedback given to the students will increase in the form of bar chart. Another help – unintelligent help (using glossary), adapted from PACT Geometry tutor- can be requested by the TA when he/she is either confronted with one error or have had no errors occurring for each step.

However, there is evidence that users rarely use a glossary, which leads to a number of errors being made (Aleven & Koedinger, 2000). Thus, our hint messages include a level that encourages the TA to read the glossary when required.

In order to scaffold the TA, adaptive scaffold can be accomplished by offering help from the system according to the TA's action whereas adaptable scaffold can be achieved by offering an option if the TA needs scaffolding by the system. In general, for scaffold interfaces, McFeSPA adopt SCI-WISE approaches according to White, Shimoda, & Frederiksen (1999).

5.3.3 Scaffold Fading

McFeSPA can be designed to provide information and advice to help the TA to measure his/her progress. Once the TA masters the knowledge of giving good feedback, he/she can withdraw/fade scaffolding of McFeSPA. There are three ways of fading:

- Through less use of support
- Through student-selected level of supported use
- Through stopping immediately (Guzdial, 1995).

The first way could be if the feedback skill meter (any feedback type) is more than 50% full (i.e. the TA has nearly mastered giving such feedback), then McFeSPA provides delayed feedback of the occurrence of any errors at that time; if the feedback skill meter (feedback types) is equal or less than half full, then the system offers immediate feedback. The second way could be that the system

offers five levels of help which are available for the TA to request help at any level and any times according to his/her needs. The third way could be that once the TA has mastered any kind of feedback giving, the system will immediately stop giving help as adaptive fading. In other words, adaptive fading can be done depending on the TA's performance. However; the TA could do this if s/he does not need adaptable fading help from the system.

With respect to the type of scaffolding mentioned above, the following analysis and categorization of scaffolding systems based on Randoll and Kali (Randoll & Kali, 2002)'s criteria according to Table 5.1 is presented. These systems provide help scaffolding the learners to learn in a particular context, which is not in the context of helping the feedback giver to give feedback.

Table 5.1 Analysis and categorization of Scaffolding systems based on Randoll and Kali (Randoll & Kali, 2002)'s criteria.

Systems	Types of scaffolding				Scaffolding-Interfaces			Fade of scaffolding	
	Functional	Process	Content	Metacognitive	Stable	Adaptable	Adaptive	Adaptable	Adaptive
Cognitive tutors(Anderson, 1993) [Domain: Algebra]	(n/a)	√	√	√	√	√	-	-	-
Contingent Tutoring (Wood & Wood, 1999; Wood, 2001) [Domain: 1)EXPLAIN(Experiments in Planning And Instruction) : Tower of Nottingham 2) A Quadratic Tutor	(n/a)	√	√	√	-	-	√ √	- √	-
Ecolab(Luckin, 1998): A Vygotskian perspective on help (Luckin & du Boulay, 1999) [Domain: food web ecology] 1)VIS 2)WIS 3)NIS	(n/a)	√	√	√	-	√ √	√ √	-	-
SE Coach (Conati & VanLehn, 1999, 2000)	√	√	√	√	√	√	√	-	-
SCI-WISE (White et al., 1999)	(n/a)	√	√	√	√	√	√	-	-

Systems	Types of scaffolding				Scaffolding-Interfaces			Fade of scaffolding	
	Functional	Process	Content	Metacognitive	Stable	Adaptable	Adaptive	Adaptable	Adaptive
TheoryBuilder (Jackson et al., 1998)	√	√	√	√	-	√	-	√	-
PACT Geometry Tutor (Anderson et al., 1995)	(n/a)	√	√	√	√	√	√	-	-
McFeSPA (Kochakornjaru pong et al., 2005)	√	√	√	√	√	√	√	√	-

According to Table 5.1, McFeSPA has both adaptable & adaptive scaffold (adaptable because it allows the TA to set a scaffolding option either on or off; adaptive because it automatically pops up in the case of the TA selecting an inappropriate answer.) McFeSPA can be designed to provide help at the domain level i.e. at the level of individual actions. The help becomes available when the TA is completing his/her specific actions and makes an error. McFeSPA can be designed to provide assistance in terms of the adjustment of help according to the TA's action.

5.4 Design of McFeSPA's Behaviour

McFeSPA is a computer-support system that helps the TA to learn to give good feedback. According to McFeSPA's scaffolding design above, McFeSPA's features can be identified by using technical terms of behaviour of tutoring systems (VanLehn, 2006). These terms consist of task domain, task, step, 'knowledge component', 'knowledge event', 'outer loop', 'inner loop', and incorrect.

According to (VanLehn, 2006), the 'task domain' means the information and skills being taught by the tutor. A task means a multi-minute activity that can be skipped or interchanged with other tasks. A step means completing a task that consists of multiple steps. Each step is a knowledge event.

'Knowledge event' means a mental event. There are usually several events per step. 'Knowledge component' means a 'task domain' concept, principle, rule, fact, etc that experts use to accomplish tasks. 'Outer loop' means that an ITS behaves as if it had an outer loop (iteration) over tasks. 'Inner loop' means that an ITS behaves as if it had an 'inner loop' over steps. Incorrect means inconsistent

with the instructional objectives of the tutor. The tutoring system that contains an inner loop, is called an intelligent tutoring system (VanLehn, 2006). Thus, McFeSPA can be referred to as an intelligent tutoring system with regard to the inner loop.

5.4.1 Task domain

Task domain in McFeSPA is giving good feedback with a good feedback pattern according to the students' programming script.

5.4.2 Task

There are several tasks that McFeSPA needs the TA to accomplish for each marking. For example,

- Task 1: Giving the explanation of error/weaknesses in problems from the errors analysed by the system

- Task 2: Editing feedback report

- Task 3: Prioritising the issues for feedback

- Task 4: Encouraging TA to give feedback (e.g. immediate feedback to encourage the TA after perform the right action)

5.4.3 Step

A Step is completing a task with several steps for each TA interface event. For example,

- The steps of task 1 above can be giving feedback regarding explanation of particular errors/weaknesses made by the students in their programming assignment (using knowledge of quality feedback e.g. important feedback, detailed/elaborative feedback), and giving feedback with relation to comparison of the current student's errors with the previous student's error (e.g. using knowledge of quality feedback e.g. feedback loop, individual feedback)

- The steps of task 2 above can be giving feedback in terms of asking questions to add any questions into the feedback report,

- The steps of task 3 above can be organizing feedback with reference to a good feedback pattern by starting and ending with 'positive feedback'.

5.4.4 Knowledge component

Knowledge component in McFeSPA refers to the principles of McFeSPA (see Section 5.6) which consists of rules for quality feedback, rules for feedback pattern, and rules for tutor's hints

5.4.5 Knowledge event

A knowledge event is a mental event, i.e. if the knowledge component is "Before giving the feedback loop, the marker should check the history of student's errors", if the knowledge event is "Because the detail of current feedback does not correspond with the history of student's error, I should check the history of student's error by viewing the history of 'student's error' interface"

5.4.6 Outer loop

Outer loop, in McFeSPA, is the TA model (see McFeSPA's architecture in Section 5.5). Outer loop is to decide which task the TA should do next. Outer loop in McFeSPA can be designed with three methods. These are display menu, fixed sequences, and Macroadaptation (Corbett & Anderson, 1995).

5.4.6.1 Display menu

McFeSPA displays a menu (e.g. to select student's script) and lets the TA select the next task. Each student's script has different types and numbers of error/weakness after the scripts were analysed by the system to be used in the next task.

5.4.6.2 Fixed sequences

Outer loop assigns a task in a fixed sequence i.e. in McFeSPA, the TA has to select the student's script first then analyse and provide the explanation for error/weakness of the occurrence of each error/weakness type, select feedback template, generate feedback report, etc.

5.4.6.3 Macroadaptation

Outer loop in McFeSPA is based on a pedagogy called macroadaptation (Corbett & Anderson, 1995). Macroadaptation is defined as how to present selected knowledge. Not only can McFeSPA represent correct and incorrect knowledge components (i.e. out of the solution path), but it can also represent other stable traits of the TA such as customization to keep the TA's preference for verbal explanation (e.g. keep record of refused help, accepted help, current hint of each step, etc.). For example McFeSPA can be assigned a task that requires many knowledge components that are already mastered by the TA and just two components that are not yet mastered.

5.4.7 Inner loop

The Inner loops are steps within a task. The Inner loop in McFeSPA contains minimal feedback, error-specific feedback, hints, assessing knowledge, and reviewing solution.

5.4.7.1 Minimal feedback

McFeSPA gives minimal feedback to the TA on a step that remembers previous steps as correct. It provides minimal feedback on demand i.e. when the TA clicks the 'OK' button in the right step to the appropriate answer according to McFeSPA.

5.4.7.2 Error-specific feedback

Error-specific feedback on an incorrect step is provided by employing the contingent help method (Wood & Wood, 1999) with five levels of help. The error-specific feedback will be provided according to the TA's incorrect step e.g. help level 4 of hint#1: "This is the 2nd occurrence of an error of [type name]) which the student has been making more than previously. You should encourage the student to avoid this error. Have another go, [TA's name]." (see Section 5.7).

5.4.7.3 Hints

McFeSPA offers help with fixed help-giving policy when the TA could not enter a correct step (adaptive help) while the TA decides whether to accept or refuse

help from the system. Once the TA accepts help, McFeSPA provides a hint to the TA according to the TA's current action. Hint in McFeSPA is based on the contingent tutoring method with five levels of help that start from the weakest hint which is vague and ending at a bottom-out hint which is a clear explanation.

5.4.7.4 Assessing knowledge

McFeSPA assess the TA's knowledge of giving good feedback by representing the skill meter of each feedback type that the TA performed.

5.4.7.5 Reviewing solution

Reviewing the TA's solution is the method of giving feedback and hint between steps during problem solving. McFeSPA gives this after the TA has submitted a solution

5.4.8 Incorrect

When the TA departs from a step, a solution path of McFeSPA, i.e. the TA does not meet its current instructional objectives it is considered incorrect. McFeSPA provides a feedback message to the TA for such an incorrect step.

5.5 McFeSPA Architecture

Providing novices with mentors is a sensible way of helping TAs to learn to mark programming assignments. This is close to the method of providing cognitive apprenticeship (Collins et al., 1989)) in Chapter 2, and this is the framework we adopt. Hence we require the content, methods, sequencing, and aspects of social learning for designing McFeSPA[10]'s architecture (Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship, see Figure 5.3). The architecture of the TA system is the conceptual diagram for scaffolding framework for provision of feedback on students' assignments. The design of McFeSPA's architecture is based on several approaches e.g. Andragogical model (Knowles, 1988, 1990; Knowles et al., 1998); rules for tutor hints (rule for

[10] McFeSPA will run in two modes - scaffolding on or off - this is done for experimental reasons - see later (also see Chapter 6-8).

hints/scaffold for giving good feedback by contingent help (Wood, 2001;Wood 1999;Wood, Wood, Cheng, 1999), help seeking and help design (Aleven, et al., 2003); knowledge of feedback pattern and knowledge of quality feedback based on design of feedback in Chapter 3; knowledge of scaffolding (e.g. Hints/Guide/Prompt/Scaffold information for the right quality feedback) by investigating Heffernan's (2001)work, part of the scaffolding approaches in Ecolab (Luckin, du Boulay, 1999); knowledge of error/weakness messages by providing a list of choices for the TA to choose, also the TA can add/edit/delete this message based on the types of errors in Chapter 4; knowledge of individual TA by using customisation of the system (from the interface and click event that the TA performs from individualized instruction (Gagnè, Briggs, and Wager, 1988) and from the principle of instructional design).

Furthermore, we diagnosed TA's lack of experience in giving feedback (see Appendix A) to design a model of training TAs to give quality feedback. This model consists of several variables of giving feedback e.g. delayed timer (from the system), return of voice e.g. "you did that wrong" how to help TA giving useful feedback in terms of what went wrong (- student went wrong) "Do you think they (students) need some positive feedback", "why don't you put this first?". The system gives hints about giving feedback, which is received from the decision of the rule of dialogue response.

5.5.1 Content

McFeSPA has two kinds of domain knowledge: about feedback including knowledge of feedback patterns, scaffolding, and about quality feedback based on analysis of the level of feedback content in Chapter 3; and the programming domain - including knowledge of errors/weaknesses as in Chapter 4 such as problems with program design, implementation and program style. McFeSPA also has heuristic knowledge encoded as rules for feedback pattern, for providing quality feedback, tutor's hints, and control strategies such as dialogue responses required to move to the next stage depending on the TA's action.

- *Knowledge of feedback patterns*: (presented in Chapter 3) e.g. early warning, positive feedback first, embrace correction, etc. (Eckstein et al., 2002); knowledge

from design of feedback-contents (Jang et al., 2001) such as topic contingency (TC), response contingency (RC) etc. -see Figure 5.4.

- *Knowledge of quality feedback*: (presented in Chapter 3) e.g. quality feedback such as positive feedback, individual feedback, feedback loop, important error/specific feedback, detailed/elaborative feedback etc; Draper's level of giving feedback (Draper, 1999) such as describe what was wrong (input), describe the right answer, describe which section makes the right answer, tell what the result was etc. - see Figure 5.4.

- *Knowledge of scaffolding*: we use five levels of contingency as part of the scaffolding process, ideas considerably adapted from contingent help approaches (see Figure 5.4).

- *Knowledge of errors/weaknesses*: (presented in Chapter 4) we have classified students' weaknesses into design, implementation, and style weaknesses and in sequence by the critical errors so that the TAs have some particular cases to help them learn to give quality feedback to students. In our domain we have chosen the design issue in an unfamiliar programming language as a part of Prolog program design (1994). Implementation issues are grounded on the methods of Shapiro (1983) as we have chosen the error types of termination with incorrect output to be students' weakness on implementation. PRAM (Mansouri et al., 1998) has various style metrics to measure Prolog programming style. We currently only measure blank line and indentation – i.e. measurement of layout. In future work we would include more stylistic features (more details about knowledge of errors/weaknesses, see Chapter 4).

- *Rules for feedback pattern*: These rules (presented in Section 5.6) will be provided alongside the rules for tutor's hints. For example, if the TAs provide a feedback pattern –not an appropriate one according to McFeSPA, positive feedback is given first such as "feedback sandwiches", while organizing the feedback then McFeSPA[11] provides different levels of help to scaffold the TAs.

- *Rules for quality feedback:* These rules (presented in Section 5.6), applying to all error/weakness types diagnosed by McFeSPA, will be provided alongside the

[11] McFeSPA exemplifies the feedback patterns which are to be learned by the TA.

rules of tutor's hints in order to scaffold the TAs to provide error messages into the feedback report. For example, if an important error is found automatically for a second time - for example, an error occurs which is the same as an error found in the previous assignment - and there is more than one error, then if the TA does not provide quality feedback, McFeSPA will provide a hint (the hint will be one of five levels of help and the hint will be terminated when the TA performs the right action or refuses the help-offer).

- *Rules for tutor's hints*: These rules (presented in Section 5.6) employ five levels of contingent help from the "knowledge of scaffolding" database and provide incremental support to self-explanation through different levels of scaffolding based on contingent help approaches. These rules are applied alongside the rules for providing feedback for particular error/weakness types. We have defined a number of solution paths for providing a feedback message to help TAs learn to provide quality feedback. For example, suppose A1 is a path providing only the error line number and brief error message. In this case, if McFeSPA diagnosed that the current error found is the 1st error of any particular error type, from the student's profile, but the TA still selects A1 then five levels of help/hint are provided from level 1 to level 5; however, it is unnecessary to provide all levels of help. If the TA selects the right solution while the hint message is being processed then the next level of help will be terminated according to the TA's action.

- *Rules for dialogue response:* These rules (presented in Section 5.6) provide the responses from McFeSPA to remind the TAs, for example "don't do so much" or "don't spend a lot of time on reworking the analysis of the solution", and so on.

5.5.2 Methods

The Scaffolding approach involves helping the learners to succeed in a way that they could not accomplish on their own (e.g. the Zone of Proximal Development (ZPD) (Vygotsky, 1978)). Scaffolding means providing support to allow the learner, the TAs here, to think for himself/herself. Furthermore, McFeSPA is like a tool for enculturating the TAs into the thinking pattern of experts. In addition, instructional scaffolding is an effective way to help the TAs accomplish discrete

learning tasks. In addition, TAs are like adult learners[12] and they could learn better when their needs or interests lie in improving their learning. To design teaching methods for helping the TAs, we have selected a scaffolding approach together with andragogical approaches (Knowles et al., 1998) and design principles of McFeSPA (see Section 5.6), helping them to acquire an integrated set of cognitive and metacognitive skills through the process of observation and guided and supported practice. We have also opted to implement fading within McFeSPA. This encourages the TAs' autonomy for both carrying out an expert problem solving process and defining or formulating the problems to be solved when they become 'masters'. In reference to methods, they were also proposed as McFesPA's approaches in Section 5.2.1.

5.5.3 Sequencing

We have selected a principle, increasing diversity, for guiding the sequencing of learning activities to accommodate the development of strength in problem-solving. It means the TAs can apply the approach/skill of giving quality feedback to any course of assignments marking based on their experience.

5.5.4 Social Learning

The following two important characteristics of the social aspects of learning have been selected; situated learning - this is interpreted as learning to give quality feedback in the situation of marking real assignments. In our case, the TA is like an apprentice who can learn in the context of their application to realistic problems: learning within a culture focused on and defined by expert practices (e.g. situated cognition (Brown et al., 1989), legitimate peripheral participation (Lave & Wenger, 1991), and situated learning in adult education (Stein, 1998)). We believe that the cognitive apprenticeship framework is useful for training TAs to give quality feedback. Although both students and TAs are expected to benefit from the approach, the focus of the work is on supporting TAs. This involves both helping the TAs to be efficient and helping the TAs to learn about quality

[12] **Adult learner** is a term used to describe any person socially accepted as an adult who is in a learning process, whether it is formal education, informal learning, or corporate-sponsored learning. (from http://encyclopedia.thefreedictionary.com, accessed on 1st January 2007)

feedback. The scaffolding approach has been selected as an appropriate approach for the TAs who, like adults, have little time to learn anything while engaged in marking students scripts. Although the implementation of scaffolding is difficult (Chee, 1995), scaffolding techniques have been implemented effectively in a number of systems (e.g. Ecolab (Luckin & du Boulay, 1999)). We have chosen to work on the problems faced by TAs in the realistic situation of marking programming assignments for large classes and providing feedback on the students' errors. The TAs are likely to be inexperienced in giving feedback even if they have excellent programming skills.

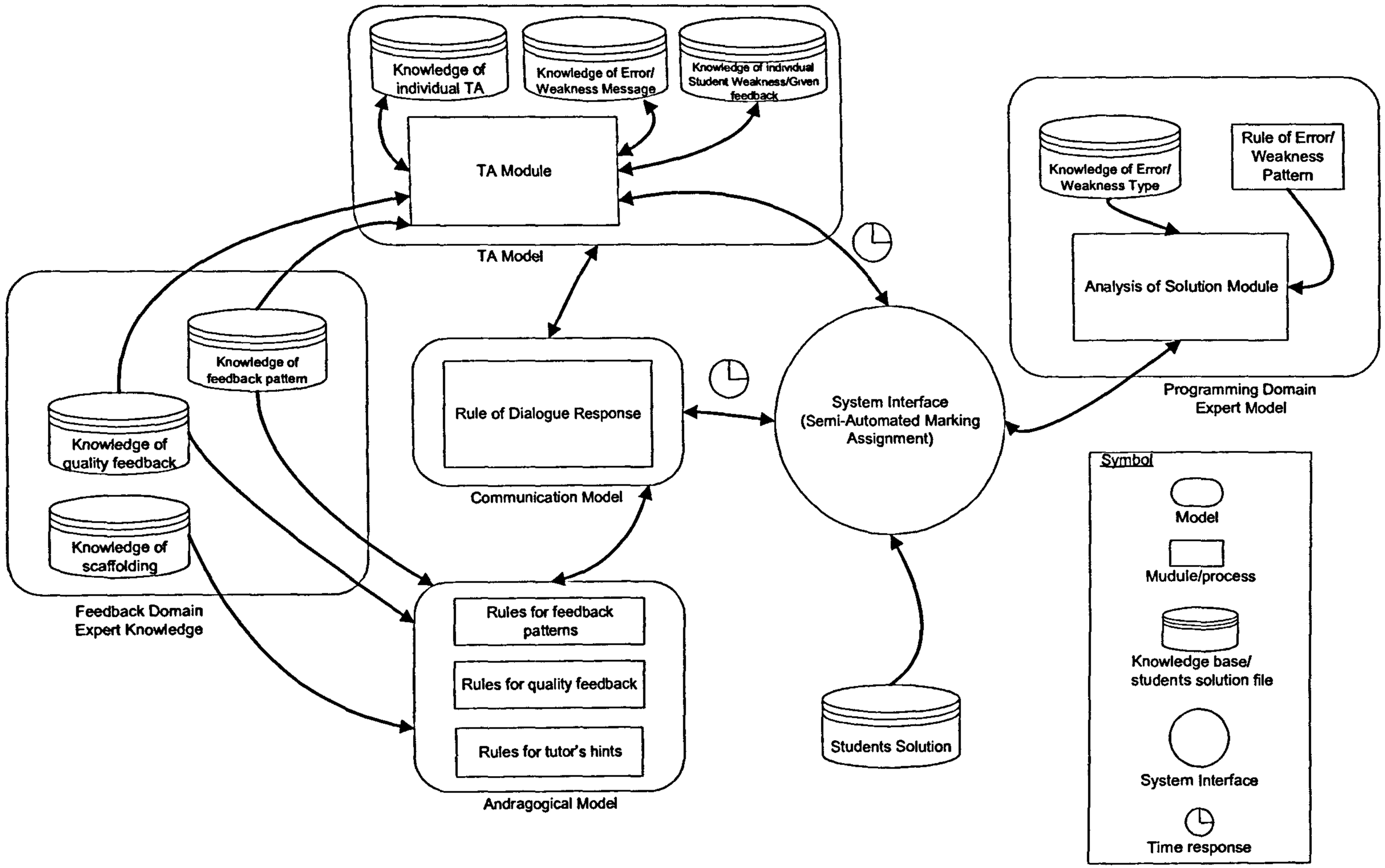
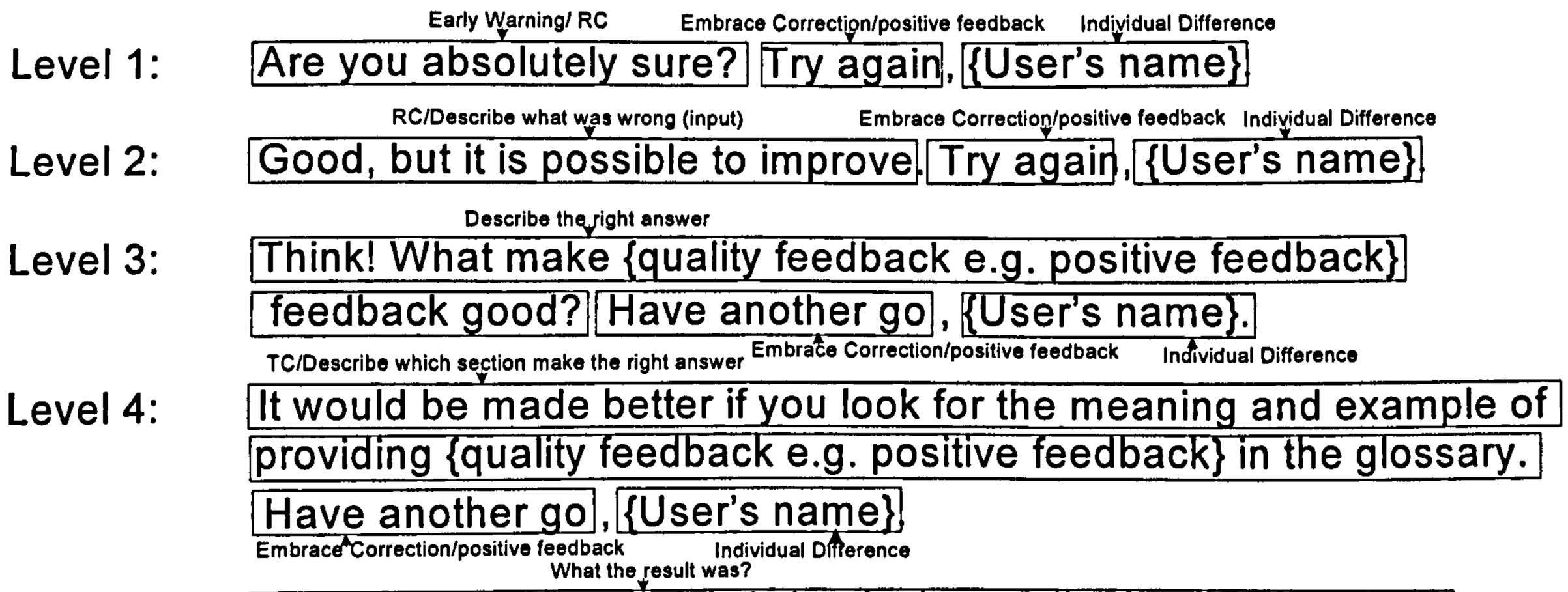


Figure 5.3 Architecture of scaffolding framework for provision feedback on students' assignment, adapted from Kochakornjarupong, Brna, & Vickers (2005)

According to Figure 5.3, when the TA obtains the students solution from the interface of the system, then the system analyses the students' solution based on the rule of error or weakness pattern by using the knowledge of weakness or error types. Thereafter, the system annotates error or weakness patterns and sends them to the TA model. In this stage the system allows the TA to add/update/delete further weakness messages beyond the system. And this module will compare each student weakness from their previous and current weaknesses in order to relay this information to the TA to provide appropriate feedback to the student. For each student, the system will only go back to review previous weakness 2 times because the system allows resubmission of assignments only twice (e.g. the student may ask the TA "Is that good enough to submit?" then the TA provides quality feedback to the students with some questions (see Table 3.2 in Chapter 3)). Therefore, considering the TA, they would mark any same that are the same no more than 3 times. The TA module stores some information that the TA does and this module will provide the information for reflection by the TA in terms of voice messages, for example "doesn't do very much or doesn't spend a lot of time on reworking of the Analysis of solution, and so on". This module depends on reflection time for the TA. It also employs knowledge of feedback patterns and knowledge of quality feedback for the TA to organise the feedback before generating the feedback report to the student. Whilst providing feedback, information is processed between the Communication model, which uses the rules for Dialogue Response and the Andragogical Model, which consists of the rules for tutor's hints, and the rules for feedback pattern. The Andragogical model utilises three knowledge bases which are the knowledge of scaffolding, the knowledge of quality feedback, and the knowledge of feedback pattern in order to scaffold the TA to provide quality feedback.



Note: RC: Response Contingency, TC: Topic Contingency

Figure 5.4 Knowledge of scaffolding: five levels of contingent help in McFeSPA

5.6 Principles of McFeSPA

McFeSPA's architecture in the previous section presents an overview of system design. This section shows the low level view of the system design. We also use a mechanism of condition-action rules for deciding what the principles of McFeSPA are according to the following. In order to complement the design of McFeSPA, we define algorithms for increasing the measurement of the skill meter (see Appendix C).

5.6.1 Rules for Quality feedback

We propose the pattern of giving an error feedback message in phase 2 (see Chapter 6) in which this includes employing quality feedback, according to our three rules in Figure 5.5. These rules belong to design error type, implementation error type, and style error type (see Chapter 4). The system will encourage the TA to provide quality feedback according to these rules.

The explanation of technical terms in the bracket is described in the following.

{Individual feedback}: (concerned with intrinsic & extrinsic - for extrinsic e.g. "very good" as a positive feedback), keeping students' history on

- how do they correct –e.g. how many praise words e.g. excellent, very good, well done-
- thinking about students' incorrect responses– which error have they ever made/haven't repeated/repeated errors

{Important error}: depends on 1) Individual difference (learner's knowledge) 2) Timing of feedback (time response); concern about quantity of feedback (e.g. do not give too many comments or to every error message for the same error type; in our design the system assists the TA to give important errors to the learner).

{Elaborative/Detailed feedback}: explanation why the answer is right/wrong

{Individual difference}: it contains more effectiveness; thinking about the learners' feeling (e.g. student's performance on their history's errors; include learner's name; avoid writing the same comment).

{Asking key question}: The TA can provide questions to the students according to their current errors and their history of errors (e.g. What problems are you having? (e.g. "go back to check your work") What are you going to do next? (e.g. "tell me how") How did you do it? (e.g. "What would make this better?") Why did you do it in that way? Can you explain this in your own words? Explain why you believe that your answer is correct or wrong? What did we learn in the class about this particular topic?). However, providing only a key question is inadequate because there might be no explanation of what to do about the key question. Thereby, the system can encourage the TA to provide such questions alongside providing a hint. In addition to ELAWAR's algorithms, they can be applied in the systems hint message. Those are

1) What is the key error?
2) What is the probable reason the student made this error?
3) How can you guide the students to avoid error in the future?
4) What did the students do well that could be noted?

Rule 1: If the TA is marking the first error found (particular error type) Then {*Thinking about student's action*}/{*Individual feedback*} (allow TA to add more details to student's profile to keep student's record)

Rule 1.1: If there are many errors (for each assignment, same type of error more than once, errors generated by the system)
Then Provide {*Important error*} & {*Elaborative/Detailed feedback*}

Rule 1.2: If there is an error (generated by the system) then Provide {*Elaborative/Detailed feedback*}

Rule 1.3: If there is no error (generated by the system) then the system asks the user to think {*Asking key question*}

Rule 1.3.1: If the TA needs to add error message (further the system)
Then the system provides {hint} (e.g. What is the key error?)

Rule 2: If the TA is marking **NOT** the first error found (same error type) Then {*Thinking about student's action*}/{*Individual feedback*} (allow TA add to more details to student's profile to keep student's record)

Rule 2.1: If there are many errors (for each assignment, same type of error more than once , errors generated by the system to compare with the previous assignment)

Rule 2.1.1: If the error type same as previous assignment (generated by the system)
Then Provide {*Important error*} + {*Feedback loop*} + { *Individual feedback*} & {*Brief Detailed feedback*}

Rule 2.2: If there is an error (generated by the system)

Rule 2.2.1: If the error type is same as previous assignment (error generated by the system)
Then Provide {*Brief Detailed feedback*} + {*Feedback loop*} + { *Individual feedback*}

Rule 2.3: If there is no error (generated by the system) then the system asks the user to think {*Asking key question*}

Rule 2.3.1: same as **Rule 1.3.1**

Rule 3: {Say nothing}: If the TA prefers {say nothing} and this is not the first error then OK (allow the TA {say nothing})
Else prompt the TA to provide other type of quality feedback

Note: each assignment can be the same assignment but resubmit

Figure 5.5 Rules for quality feedback in McFeSPA

{Feedback loop}: if the learners have not improved their learning in the next assignment then the markers should collect all given feedback for the next assignment e.g. "This is the 2nd, 3rd, ... same error found; hopefully, next time you could improve your script to avoid this type of error"

{Brief Detailed feedback}: this is like a short feedback message, which does not explain enough details about the error.

{Say nothing}: don't put any message detailing the error found in the feedback message.

Further information of the characteristics of quality feedback can be found in Chapter 2.

5.6.2 Rules for Feedback Pattern

To help TAs in providing quality feedback, the rules for feedback pattern will be provided alongside the rules for tutor's hints (see Section 5.6.3). We propose two rules for feedback pattern:

Rule 1: If the TA selects the best template to generate the feedback report the first time, the measurement of 'positive feedback' will be increased. The appropriate feedback template is via feedback sandwiches i.e. giving either 'negative feedback' or error messages between two 'positive feedback' messages.

Rule 2: If the TA provides 'positive feedback' (for a feedback sandwich), the measurement of 'positive feedback' will be increased. For example, the starting detail and the ending detail of a feedback sandwich should be 'positive feedback' not 'negative feedback'.

5.6.3 Rules for tutor's hints

To support the TAs with hints, the rules for tutor's hints use five levels of contingent help from the "Knowledge of scaffolding". These rules will be provided alongside the quality feedback rules and feedback pattern rules. Each hint provided will increase the amount of specific advice and only one level of help will be displayed at any one hint time. The help could reflect the TAs thinking in their action (Schön, 1983). We propose seven rules for tutor's hints as follows.

Rule 1: If the TA doesn't give a feedback message to students to avoid errors that the students made, for example more errors of the same kind than before; same number of errors of the same kind as before; less errors of the same kind as before. Then the system provides a hint to the TA (i.e. contingent help) and the measurement of 'individual feedback' and 'feedback loop' will not be increased.

Rule 2: If the TA doesn't give a feedback message to students to avoid errors that the students made the 1st time, then the system provides a hint as contingent help

to the TA and the measurement of 'individual feedback' and 'feedback loop' will not be increased.

Rule 3: If the TA doesn't give 'detailed/elaborative feedback' the 1st time the error is found to be made by a student. Then the system provides a hint as contingent help to the TA and the measurement of 'detailed/elaborative feedback' will not be increased.

Rule 4: If there are a number of the same kinds of error found whether such errors happened the 1st time or not but the TA doesn't give a feedback message as 'important/specific feedback' and indicate to the TA that there are more errors like this. Then the system provides a hint as contingent help to the TA and the measurement of 'important/specific feedback' will not be increased.

Rule 5: If the TA doesn't select a good template to generate the feedback report the first time. Then the system provides a hint as contingent help to the TA to help him/her select a good feedback template for generating feedback report and the measurement of 'positive feedback' will not be increased. (The good feedback template is giving feedback sandwiches, giving either 'negative feedback' or error messages between two 'positive feedback' messages)

Rule 6: If the TA doesn't provide the right student name on the feedback report. Then the system provides a hint as contingent help to the TA to help him/her give 'individual feedback' with regard to giving the right student's name from student's marking script in the feedback report and the measurement of 'individual feedback' will not be increased.

Rule 7: If the TA doesn't provide 'positive feedback' (of feedback sandwiches) with regard to, for example, the starting detail of 'positive feedback'; the ending detail of 'positive feedback'; the positive detail of the starting 'positive feedback'; the positive detail of the ending 'positive feedback' in the feedback report. Then the system provides a hint as contingent help to the TA to help him/her give 'positive feedback' and the measurement of 'positive feedback' will not be increased.

5.6.4 Rules for dialogue response

To help the TAs provide quality feedback in a short period of time with giving help/suggestion to help the TA think about their performance, we propose the following rules for dialogue response.

Rule 1: If the TA repeats taking into account the analysis of the student's solution (e.g. clicking the analysis button several times) of the same student more than three times in one minute, then the system provides a help message to the TA (e.g. "doesn't do very much or doesn't spend a lot of time on reworking of the Analysis of solution, and so on").

Rule 2: If the TA repeats taking into account generating of final report (e.g. clicking the general final report several times) for the same student more than three times in one minute, then the system provides a help message to the TA (e.g. "doesn't do very much or doesn't spend a lot of time on the reworking of generating the final report, and so on").

5.7 Context of Hints

The previous section illustrates the low level view of McFeSPA's design. In this section, we propose the context of hints employed in McFeSPA. Currently, we provide 12 hints, 12 different contexts for which contingent help is available, in the main context of learning how to provide quality feedback. The contexts, the purposes, and the forms of the hint for all hints in McFeSPA in the current version can be seen in Table 5.1.

Table 5.1 Contexts, purposes, and forms of the hint for all hints in McFeSPA (excerpted from Appendix D)

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
1	When the student has made **more** errors of the same kind than previously.	Help the TA to give a feedback message to an individual student who has made **more** errors of the same kind than	Level 1: "Are you absolutely sure? Try again, [TA's Name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]."

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
		previously to avoid errors with regard to student's error history from student's profile i.e. help the TA to give 'individual feedback' and 'feedback loop'.	Level 3: "Look for the meaning of 'Feedback loop' in the glossary. Have another go, [TA's name]." Level 4: "This is the 2nd occurrence of an error of [type name]) which the student has been making more than previously. You should encourage the student to avoid this error. Have another go, [TA's name]." Level 5: "The right answer is the 2nd choice which gives you a good 'Feedback loop'."
5	When the student has made a particular error for the 1st time	Help the TA to explain more detail feedback i.e. help the TA to give 'detailed/elaborative feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Think! What makes 'Detailed/Elaborative feedback' good? Have another go, [TA's name]." Level 3: "Look for the meaning of 'Detailed/Elaborative feedback' in the glossary. Have another go, [TA's name]." Level 4: "It would be better to provide 'Detailed/Elaborative feedback' the first time (of this type error found). Have another go, [TA's name]." Level 5: "The best answer which gives you a good 'Detail/Elaborative feedback' should be the 'Yes' option -provide 'Detailed/Elaborative feedback."

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
6	When there are a number of the same kinds of error found whether such errors happened for the 1st time or not	Help the TA not to give too many comments or to every error message i.e. help the TA to give 'important/specific feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Think! What makes 'Important/Specific feedback' good? Have another go, [TA's name]." Level 3: "Look for the meaning of 'Important/Specific feedback' in the glossary. Have another go, [TA's name]." Level 4: "It would be better to provide 'Important/specific feedback' (of this type error found) only once. Have another go, [TA's name]." Level 5: "The best answer which gives you a good 'Important/Specific feedback' should be the 'Yes- just once' option -provide 'Important/Specific feedback' only once."
7	When the TA does not select the "feedback sandwich" template which is when the error message is between two positive feedback messages	Help the TA to select the appropriate feedback template for generating feedback report. The appropriate feedback template is giving feedback sandwiches, giving either 'negative feedback' or error messages between 'positive feedback' i.e. help the TA to give 'positive feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5: "The best feedback is the error message between two 'Positive

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
			Feedback', select upper rightmost button - that is the best feedback pattern."

Giving hint#1-4 is aimed to help the TA to consider an individual student with regard to the student's error history from their profiles; this is associated with providing a 'feedback loop'. Giving hint# 9-12 is aimed to help the TA to consider detail and position for giving 'positive feedback' (of feedback sandwich). Giving hint#5-8 has different contexts and purposes of each hint (see Appendix D). There are similar forms of hints but the contexts and purposes are different. We are aware that a later version could be improved and the language used and the systematics could be changed. Our first version is aimed to provide 12 hints to test usability. In the current version, some details of levels of hints had been changed according to the suggestions of evaluators in the usability evaluation in Chapter 7 (see Appendix I). All details of the levels of help of each hint can be seen in Appendix D. In later versions the number of hints can be either decreased or increased from the current amount of hints depending upon the TAs' response.

5.8 Andragogical model for training TA improving giving feedback

In order to design a help-giving feedback system to support the TAs, who are like adults as mention earlier, in learning to give feedback and to help them improve giving feedback, we need to take into account adult learning theory. Knowles (1988; Knowles, 1990) defined andragogy as 'the science and art of helping an adult to learn'. He defined five elements of the learning process:

- the concept of the learner
- the role of the learner's experience
- readiness to learn
- orientation to learning and
- motivation.

He defined andragogy as a theory of learning for adults as opposed to pedagogy, which focuses more consistently upon learning of children.

Andragogy, helping adults learn, is based on a different set of beliefs. Adults learn better when their needs and interests, life experience, self-concepts, and individual differences are taken into account. There are six core principles, as a basic theory underpinning adult learning, which were developed by theorists (Knowles et al., 1998). The design of the scaffolding feedback system is based on these principles with regards to helping adults learn. These are:

1) Adults are motivated to learn as they experience needs and interests that learning will satisfy so the TAs, will be motivated to learn how to improve giving feedback by using McFeSPA when they want / need to improve their feedback giving skills.

2) The learning environment is characterised by physical comfort, mutual trust and respect, mutual helpfulness, freedom of expression, and acceptance of differences. Consequently, the learning environment of McFeSPA will be satisfied when McFeSPA has improved all usability problems according to the TAs' suggestion.

3) The learners perceive the goals of the learning experience to be their goals. Adults' orientation to learning is life-centred. Hence, the TAs who used McFeSPA will satisfy the system when the TAs know what McFeSPA will help them learn and help to improve their weaknesses of giving some kinds of feedback and can give better feedback to the students.

4) Adults have a deep need to be self-directing. The learners accept a share of the responsibility of planning and operating a learning experience, and therefore have a feeling of commitment toward it. The learners participate in the activity of the learning process. Thus, the TAs who intend to improve giving feedback will concentrate on giving feedback and try to learn to use McFeSPA to gain knowledge of giving better feedback as much as s/he can.

5) Experience is the richest source for adults' who are learning. Accordingly, McFeSPA could help the experienced TA to learn to give improved quality feedback faster than a novice TA.

6) The learning process is related to and makes use of the experience of learners. McFeSPA could help the TA improve giving better feedback when the process of giving feedback in marking programming assignments is similar to the real situation of marking and the system provides help to the TA to overcome any difficulty of using the system. Therefore, rules for tutor's hint in principles of McFeSPA (see Section 5.6.3) could help the TAs conquer any obstruction while using the system.

5.9 Contextual Design

To help learning to give feedback, the system supports the teachers to learn to give feedback to students in which the contextual design can be designed as per the flowchart in Figure 5.7. As can be seen from figure 5.7, first of all, the system analyses the student's solution by using the design/implement/style analyser that retrieves facts from three knowledge domains of error/weakness according to Figure 5.6. These knowledge domains will be decided by the weight of error. Then all errors will be sorted together to be the input to the next stage. After the system receives the input file of all error/weakness from the analysis stage, then it will go to the process of annotate summary of all kinds of error/weakness messages. After that the system goes to the process of annotated quality feedback (e.g. 'Detailed/Elaborative feedback', 'Important/Specific feedback', 'Feedback loop', 'Individual feedback') upon which the TA makes a decision or requests help from the system. This is followed by the process of annotated feedback pattern upon the TA's decision or help offer from the system. There after it goes to the process of annotate weakness message with detailed feedback. At this stage, the TA can add/manage any error message beyond the system. The next step is to organise and prioritise weakness/error state for each problem. Finally, it is the process of generating feedback report and sending an e-mail to the student. All processes after obtaining the output of analysers can be done by the TA himself or with the support of the scaffolding system.

D.K. Design problem	D.K. Implementation problem	D.K. Style problem
Domain Knowledge of Design problem	Domain Knowledge of Implementation problem	Domain Knowledge of Style problem
e.g. table D - error type D1 = 1 - error type D2 = 2 - error type D3 = 3	e.g. table I - error type I1 = 4 - error type I2 = 5 - error type I3 = 6	e.g. table S - error type S1 = 7 - error type S2 = 8 - error type S3 = 9 ... ...

Figure 5.6 Domain Knowledge of Error/Weakness types (adapted from Kochakornjarupong (2003))

5.10 Summary

Various scaffolding systems have been employed in different context/domains. McFeSPA is another such scaffolding system that adopted several approaches that enabled it to be designed as a scaffolding system to help TAs in giving good feedback to students and at the same time to improve given feedback. The design of the leaning environment of McFeSPA is the design of the process of using the system till achieving the final task – i.e. there are several cases which lead to several outputs and there is only one appropriate final solution that is the right answer of each case. McFeSPA is an assistant system that provides contingent hints leveled from 1 to 5 for a particular situation depending on the TA's skill. The hint increases by offering a greater amount of specific advice. If the TA uses the system with scaffolding and the TA does not follow the system solution path, the system will automatically produce a popup help-offer which starts at hint level 1 (after the TA has logged into the system) to ask the TA whether he/she needs any help from the system. If the TA requests help at first, McFeSPA will display the next level of hint when the TA is still out of the system's solution path. Nevertheless, carefully 'Phrasing feedback' in each contingent hint is important. If the feedback is not phrased well, the TA my not accept the feedback i.e. help messages may distract the TA if he/she does not understand or becomes frustrated with them. We believe McFeSPA represents a new genre of software that allows the users to express their metacognitive ideas and sociocognitive practices as they undertake complex tasks – depending on students errors; however, the problem in designing McFeSPA is determining a good method for representing ideas about

how to carry out the feedback report, how to scaffold them, and how to monitor either their progress or performance. In the next chapter we draw out the scenario-based scaffolding system to elaborate on the environment for using McFeSPA.

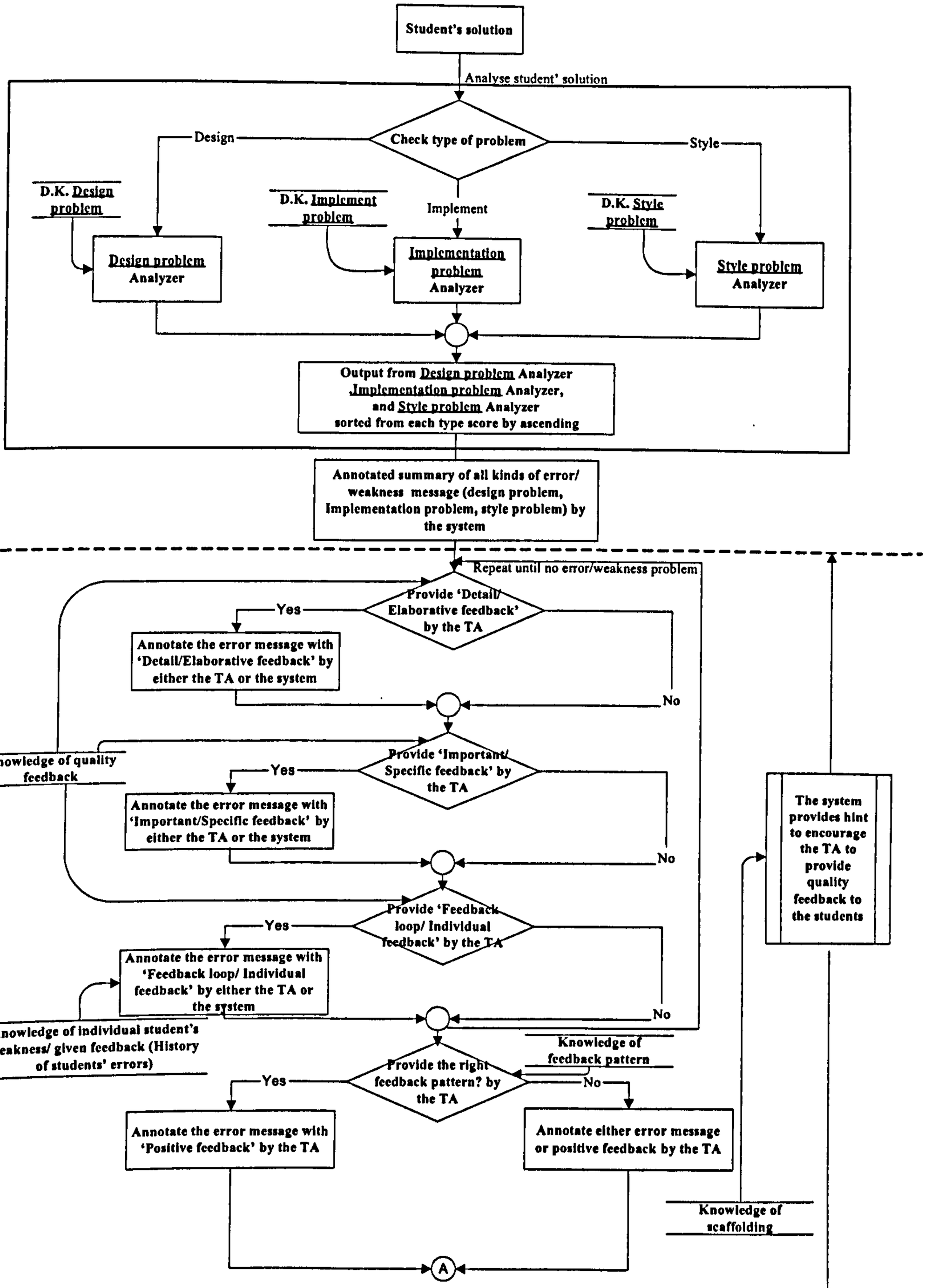


Figure 5.7 Contextual Design of Semi-automatic help system (adapted from Kochakornjarupong (2003))

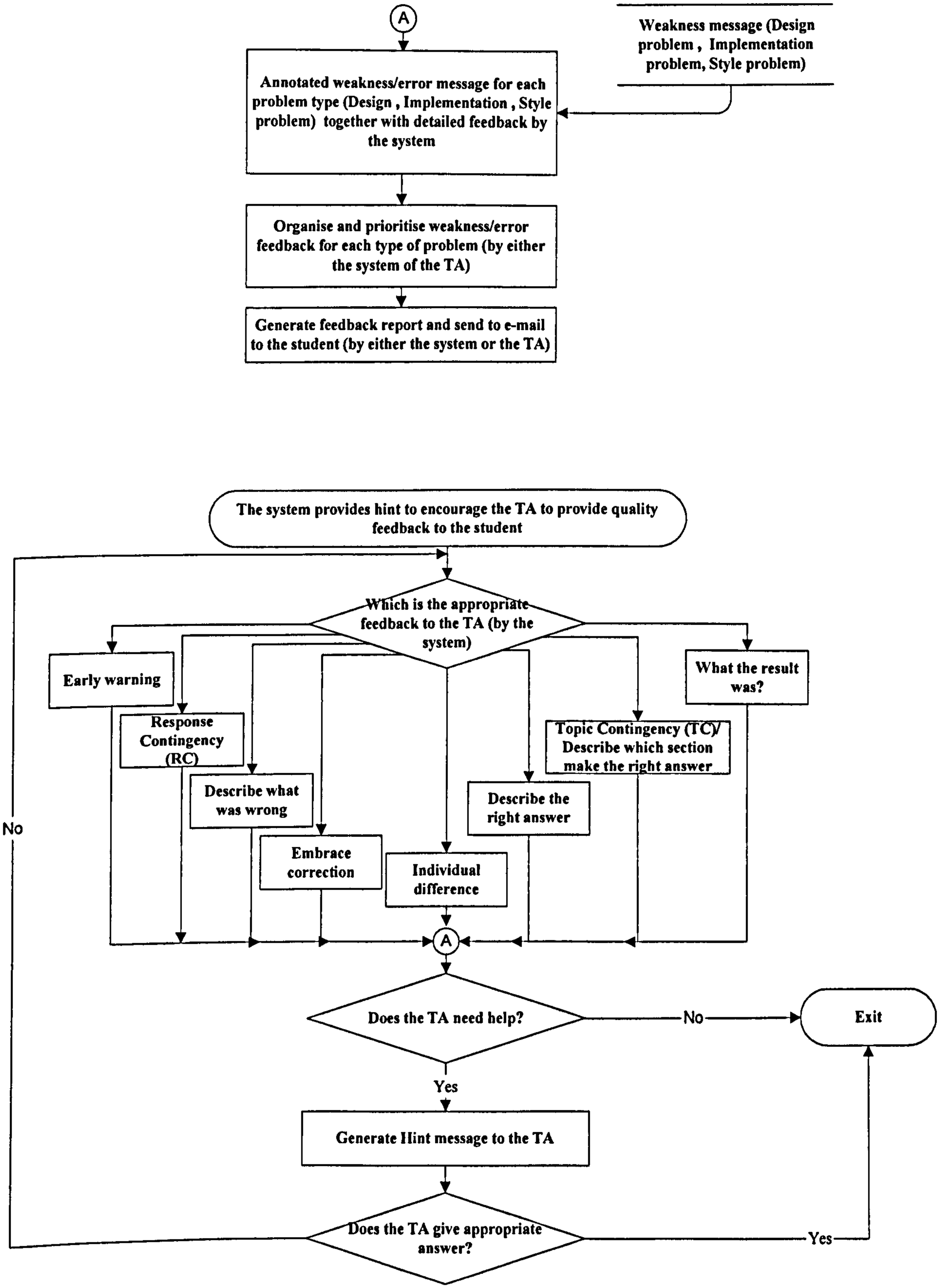


Figure 5.7 Contextual Design of Semi-automatic help system (adapted from Kochakornjarupong (2003)) (cont.)

Chapter 6

Scenario-Based Scaffolding System Design

6.1 Introduction

In the previous chapter, we described the design of the scaffolding system. In this chapter we consider the scenarios in which the scaffolding system is to be utilised in order to develop the interface design, and ensure that the interface is usable (see Chapter 7). In the following, we present the design structure, the origin of scenarios, building scenarios, design principles for using scaffolding, using scaffolding for different TA needs, and the design of the content alongside scaffolding.

6.2 Design structure

Requirement engineering and usability engineering are essential parts of software development, which are brought together in scenario-based approaches. Accordingly, in order to design a scaffolding system we found that the scenario-based approach is a potential methodology to help either designers or analysts to reuse or redesign a system according to the users[13] concern. Scenarios can be represented in a way that is accessible to both TAs and designers providing an effective tool for communication for both groups (Carroll, 1995). Carroll & Rosson (1992) noted that detailed scenarios establish a narrative theory of the artifact in use. The benefit of usage scenarios is that they can be generated and developed "even before the situation they describe has been created" in which use-scenarios can be the principal design representation of an artifact (Carroll & Rosson, 1992) in the design of various tools (Carroll, 2000).

In creating a state-of-the-art system, designing TA training requires a set of TA interaction scenarios, documentation, and a usability test. Scenarios are used

[13] Users in McFeSPA are either novice teachers or novice lecturers or novice teaching assistances (TAs) or novice tutors.

in a number of approaches to theory-building in HCI, according to Carroll's (1995) taxonomy of the natural event empirically. Besides, in software engineering, implementation of the application functionality can be supported directly by a scenario-based analysis. Using a scenario can help people better understand and provide better use of documentation and training if users are in the context of the task that they need to achieve. Scenarios also provide a framework for evaluation of the functionality, usefulness, and usability of the system (2002). Scenarios help us to design the context aspects of the system. In order to carry this out we adopt Randoll & Kali's (1992) approaches to design principles for the use of scaffolds (see Section 6.5). Thus, this chapter basis the design on the methodology of Carroll & Rosson (2000) as well as a combination of scaffolding approaches.

In this thesis, we divided our design into three levels which consists of:

- high level or abstract level or contextual level;
- schemata level;
- low level or prototype level.

In the high level description of the system, we have selected a scenario approach to generate the contextual description of the system because such approaches pertain to the psychological design rationale of an artifact-in-use in terms of causal schemas –claims- under the scope of a basic task usage situation. Regarding the schemata level, we ground our work on the cooperative evaluation approach. For the low level, we apply HCI approaches to implement the prototype which is described in Chapter 7.

6.3 Origin of Scenario-Based Approaches

According to Carroll (2000) scenarios can come from ethnographic field studies, participatory design, reuse of prior analyses, scenario typologies, theory-based scenarios, technology-based scenarios, and transformations,. The scenarios used for the design of McFeSPA are derived from ethnographic field studies, scenario typologies, and transformation according to the following.

- **Ethnographic field studies**

Ethnographic field studies are derived by the observer who builds the ontology of the agent, goals, actions, events, obstacles, contingencies and outcomes from scratch. Examples of such an ontology can be seen from the following questions.

- What events occur in the domain? (e.g. TAs are marking assignments and giving feedback to the students, etc.)

- What types of roles do people play? (e.g. TA as a novice marker/lecturer/tutor)

- How do people do the work together? (e.g. Usually TAs mark an assignment alone- Individual working)

- What action do they take? (e.g. read student's solution, give quality feedback, etc.)

- What are the common obstacles in achieving the goal and action? (e.g. TAs may not have much experience in providing any new error messages for the system, however this is not the main obstacle. In addition TAs may take some time in the preliminary state of learning to use the system until they gain experience and then they do not need any help from the system.)

-What depends on what? (e.g. error messages depend on student's scripts)

- What sorts of variations occur in actions and events, and with what consequence? (e.g. support/ help/ hint/ prompt/ scaffold from the system to help the TAs to give quality feedback. The level of help depends on the TAs action and his progress.)

- What are the typical and significant outcomes that occur in the domain? (e.g. feedback report to the students)

It would appear that these studies are good for discovering "exotic" error scenarios.

- **Scenario typologies**

The way to support scenario reuse is to identify categories or types of scenarios and domain –called scenario typologies. There are six categories of scenario typologies – a typology of user concern (see Section 6.4.2); however, this may be useful for designers who work in a particular domain e.g. educational technology for classrooms. Scenario typologies consist of Orienting to appropriate goals (e.g. orienting to the novel task situations, identifying and analysing appropriate goals); Interacting with the environment opportunistically; Searching under a description (e.g. looking for the menu item that will allow the TA to create the feedback report); Following procedures; Seeking and using explanations (e.g. finding the meaning of quality feedback from the glossary); Reflecting upon and crafting one's own work.

- **Transformations**

Scenario transformation is to support hypothetical "what could go wrong" lines of reasoning. It is to change points of view and ask how each scenario would appear to another actor. In addition, it is the exchange of the tools and other task objects occurring in a given scenario with another set of tools and objects (easy to criticise and easy to improve). Scenario transformation in this research was adopted to build the interface (see Chapter 7) and some interfaces were changed according to the evaluators' and participants' suggestions (see Appendix I).

6.4 Building Scenarios

Building scenarios by employing the scaffolding approach (see Section 5.2.1, Chapter 5) requires an empirical approach, analytic approaches, and design situations while the TA is marking assignments with a semi-automated marking system. We will explore these further below.

6.4.1 The Empirical Approach

This approach is basically derived from general problems that arise with a given task as it is currently carried out based on observation, investigation, or

interviews. This refers to the task-artifact cycle –ontology of HCI, as shown in Figure 6.1.

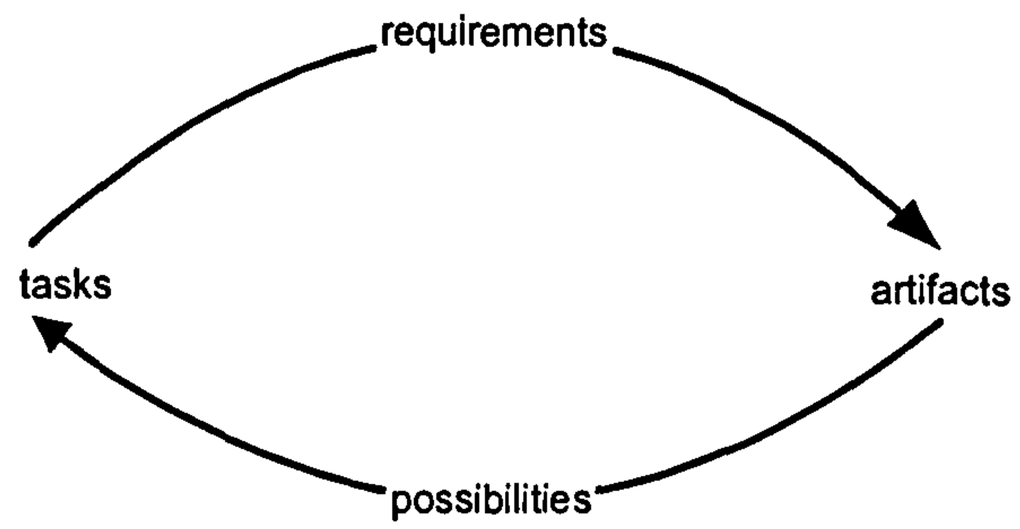


Figure 6.1 The task-artifact cycle

The current tasks for McFeSPA are obtained from both the investigation in Chapter 2 and from the interview based questionnaire survey of three lecturers and two teaching assistants (see Appendix A). Thus, the requirements of McFeSPA are:

- ***feedback designed to encourage the TA***, for example, encouraging the TA to provide feedback patterns and quality feedback to the student (e.g. 'individual feedback', 'important feedback', 'positive feedback', etc.). For further details see "Scenario of Contingent support depending on the TA's action" in Section 6.4.3.5 for more details.

- ***support for showing error/weakness in problems***, for example, a design, implementation or style problem, see phase 2 in Section 6.4.3.2, and "Scenario of Contingent support depend on the TA's action" in Section 6.4.3.5 for more details.

- ***support for prioritising the issues for feedback*** i.e. TA can organise/dominate feedback message, see phase 3 in Section 6.4.3.3 for more details.

- ***support for editing feedback report***, for example, help TAs change some information in the feedback report, see phase 3 in Section 6.4.3.3 for more details.

- ***the provision of contingent support***, for example, the support system itself depends on the given situation in learning to give feedback. Contingent help/support is a kind of scaffolding. In addition, the role of scaffolding is giving feedback from the system to the TA (e.g. the system may tell the TA when given

feedback has gone wrong.), see phases 2 and 3 in Section 6.4.3.2, 6.4.3.3), and "Scenario of Contingent support depending on the TA's action" in Section 6.4.3.5 for more details.

- ***The value of the system for giving feedback*** that comprises a) conditional factor for feedback pattern (see "Scenario of Contingent support depending on the TA's action" in Section 6.4.3.5 for more details), b) feedback organisation, see phase 3 in Section 6.4.3.3 for more details.

- ***In order for McFeSPA be considered an intelligent system***, a few rules can be added for giving quality feedback, see "Scenario of Contingent support depend on the TA's action" in Section 6.4.3.5 for more details.

6.4.2 The Analytic Approach

The analytic approach involves organising the collected scenarios by using the theory of scenarios. Scenarios can be classified by multiple levels of abstraction, which are derived from the user's concerns. With regard to the typology of user concerns of McFeSPA as usability requirements, we present the typical scenarios in Figure 6.2 and Figure 6.27. For each scenario –interacting with the environment – we pay attention to our characteristics of quality feedback.

6.4.2.1 Orienting to appropriate goals

Encouraging the TA to pay attention in his/her performance can be done by offering orienting to appropriate goals. For example,

When the TA decides to organise feedback, there will be a feedback template, offered as a feedback sandwich, which consists of six choices; however, only one choice is the right answer which is the appropriate feedback pattern to provide to the student. If the TA does not choose the right one, a scaffolding message pops up with five levels of help. The choices are:
Choice 1, feedback open-faced sandwiches, are error messages followed by 'positive feedback'.
Choice 2, feedback open-bottom sandwiches, are 'positive feedback' followed by error messages.
Choice 3, feedback sandwiches, are the error messages covered by 'positive feedback' – error messages are between 'positive feedback'.
Choice 4, feedback layered sandwiches, are design messages, implementation messages, and style messages covered by 'positive feedback'. For example, start with 'positive feedback' then design messages then 'positive feedback' thereafter implement message and so on.
Choice 5, feedback open-all sandwiches, are error messages only.
Choice 6, feedback unfilled sandwiches, are 'positive feedback' only.
In the first selection if the TA selects choice 1, choice 2, choice 4, choice 5, or choice 6 which are incorrect with regard to giving quality feedback, help level 1 will be provided by the system as below.

Level 1[14]: Are you absolutely sure? Try again, {TA's name}.

In the second selection if the TA selects choice 1, choice 2, choice 4, choice 5, or choice 6 which are incorrect with regard to the quality feedback, help level 2 will be provided by the system

Level 2[15]: Good, but it is possible to improve. Try again, {TA's name}.

In the third selection if the TA selects choice 1, choice 2, choice 4, choice 5, or choice 6 which are incorrect with regard to the quality feedback, help level 3 will be provided by the system

Level 3: Think! What makes {feedback e.g. 'positive feedback'} good? Have another go, {TA's name}.

In the fourth selection if the TA selects choice 1, choice 2, choice 4, choice 5, or choice 6 which are incorrect with regard to the of quality feedback, help level 4 will be provided by the system

Level 4: Look for the meaning of providing {quality feedback e.g. 'positive feedback'} in the glossary. Have another go, {TA's name}.
In the final selection if the TA selects choice 1, choice 2, choice 4, choice 5, or choice 6 which are incorrect with regard to the quality feedback, help level 5 will be provided by the system

Level 5: You should select choice 3 which gives a good feedback pattern.

Figure 6.2 Orienting to appropriate goals -Content scaffolding – hint for the process of feedback template to organise the feedback module

After the TA has selected the right feedback template the create feedback report pane is then shown in which it contains a sub-pane – feedback pattern pane and final feedback report pane. In feedback pattern pane, the sequence of information appears which is the header button, 'positive feedback' button, the frame organise error messages –these consist of design button, implement button, and style button that the system allows the TA to organise these buttons according to the importance of the marking context, then followed by 'positive feedback' button, and the final sequence is the footer button. Each button the system allows the TA to double click in order to edit it. However, if the detail of the top 'positive feedback' equals the bottom 'positive feedback' detail, there will be a prompt from the system to remind the TA to rethink about his/her action again. For example "It might be better to give different 'positive feedback' to the student".

Figure 6.3 Orienting to appropriate goals -Content scaffolding – prompt to organise feedback module

6.4.2.2 Opportunistic interaction with the environment

Facilitation in McFeSPA can be done by offering opportunistic interaction with the environment. For example:

- The TA can select the type of assignment, the class, and the course before marking any assignments.

- If the TA selects 'student's solution' menu item then he/she can tell the system to analyse all error types which are classified by design,

[14] The previous message of help level 1, before updating by using the results from the usability evaluation in Chapter 7, is 'Something in your selection doesn't work for you, try again' "

[15] The previous message of help level 2, before updating by using the results from the usability evaluation in Chapter 7, is 'This may not be the best feedback, try again'"

implementation, and style; or to analyse the particular error type (e.g. only design, implementation, or style).

- To facilitate using McFeSPA, the TA can customise content/wording/add new error/create header and footer for the feedback report.

- The TA can request help from the system so that not only can he/she continue with the right process but he/she can also ask for an explanation of functionality for each item

The TA can customise the system at any phase of processing the system. When the TA is on the customisation menu item which offers various "menu item" (e.g. manage more error messages, favorite wording, favorite content, setting scaffolding in order to inform the system whether the TA needs help or not from the system.) a highlighted prompt will appear to remind the TA when they are spending a significant amount of time customising the system and the system will encourage the TA to continue the process on the next step. The TA can customise choice of consequence of error messages so that they appear in a temporary report from analyzing the student solution process.

Figure 6.4 Interacting with the environment opportunistically, as exemplified in customise option choice scenario

In the first marking of the students first assignment , if the TA decides not to indicate any detailed feedback message, there will be a dialogue box to encourage the TA to provide elaborative feedback "It might be better if you explain the error messages in the student 's feedback report at the first time of marking". The TA can set the default for generating error messages to the other students' script; however, for each student the TA can be asked by the system to teach him/her to give quality feedback because each student has individual differences.

Figure 6.5 Interacting with the environment opportunistically, as exemplified by prompting the TA to provide elaborative feedback.

The TA may interact with the system by error response. There are several option loop error scenarios such as login failure and exit failure. As can be seen from Figure 6.6, we integrate all typical error loops to one scenario.

Login Failure: A dialogue error message pops up as an error loop when the TA types either the wrong login or wrong password. The TA is allowed to try again for up to three times; otherwise he/she cannot access to the system e.g. "Sorry! You cannot enter to the system". In the case of the wrong password, the system asks a secret key question, which is set when the system is installed.
Exit Failure: The TA specifies menu item and chooses tab menu and another menu item over and over, changing messages and then back again. The TA expresses frustration and helplessness, can't see either the way out or the way to process the right process/step, and feels that he/she fails in his/her original goal of choosing the right item.

Figure 6.6 Interacting with the environment opportunistically, as exemplified in the login error loop and exit loop

The system may interact with the TA if he/she spends a lot of time in a process rather than giving feedback as can be seen from Figure 6.7.

In the situation that the TA provides feedback to the student alongside scaffolding by the system, at this process if the TA spends a considerable amount of time selecting students' solutions or the same students to analyse their solution, without continuing giving quality feedback and generating feedback reports to the students, then a dialogue response will inform the TA in terms of text or voice such as "Do not spend too much time reworking the student's solution".

Figure 6.7 Interacting with the environment by reminding the TA to carry out the process of marking assignments properly

According to the categorisation of the kinds of scaffold (Wood et al., 1999), we propose two kinds classified as types of scaffolding and scaffold interfaces. We have four types of scaffold, namely functional scaffolding as in Figure 6.8; process scaffolding as in Figure 6.11 – Figure 6.16; metacognitive scaffolding as in Figure 6.18 - Figure 6.22, Figure 6.24, and Figure 6.25; and content scaffolding - as can be seen from Figure 6.2, and Figure 6.3.

Functional scaffolding is a type of scaffolding that helps the TA understand how to use and/or interpret the system e.g. explanation of representation.

When the TA drags a mouse to any menu items or any buttons, there is a description of how the menu items and buttons work e.g. When the TA drags the mouse to the setting scaffolding menu item, there will be the following description "This is the setting option you may request help from the system or continue without system assistance".

Figure 6.8 Interacting with the environment - Functional scaffolding – dragging the mouse to the scaffolding setting menu item

In the case of the system finding only one design error, in 'Didn't use for 1st letter of variable', there will be the following dialogue box "The student has a design error of 'Didn't use for 1st letter of variable'. Would you like to include any detailed feedback message in feedback report?" followed by three choices:
1) No – "indicate the error line number with brief error messages" option;
2) Yes – "indicate the error line number with brief error messages" option and "include an example" option;
3) Say nothing.
In the case of the system finding more than one design error in'Didn't use for 1st letter of variable', there will be the following dialogue box "The student has [number of errors] design error of 'Didn't use for 1st letter of variable'. Would you like to include any detailed feedback message in the feedback report?" followed by three choices:
1) No the – "indicate the error line number with brief error messages" option;
2) Yes (only 1) the – "indicate the error line number with brief error messages" option and "include an example" option;
3) Yes (Always) –with "indicate the error line number with brief error messages" option and "include an example" option;
4) Say nothing.
In the case of any implementation errors or style errors being found, they will be similar to the above.

Figure 6.9 Interacting with the environment –the system offers a choice for giving Adaptive support module to provide a level of help for providing quality feedback and adapts the next help according to the TA action.

6.4.2.3 Information searching

Supporting the TA to find useful information in McFeSPA can be done by providing information searching, for example:

- The TA can search for appropriate types of quality feedback according to the situation (e.g. in the situation of organising feedback, the TA should not provide the same 'positive feedback' message in the same feedback report.). Otherwise, if the TA spends a lot of time on such a situation, then the system will encourage the TA to perform the task correctly.

In the stable scaffold interface, McFeSPA supports finding the description of quality feedback.

TAs can seek the meaning of information regarding quality feedback which is displayed in the list box, which is an electronic glossary. After clicking any type of feedback list, there will be a description displayed below the list together with an example of providing such feedback e.g. asking key questions: description: In order to provide constructive or focused feedback and encouragement, there should be a key question to ask in the tutorial. For example
(a) What are you trying to do?
(b) What have you been doing?
(c) What problems are you having?
(d) What are you going to do next?
However, providing only 'asking key question' feedback may appear to be negative feedback so it should be provided with a hint of 'positive feedback' together with asking the key question e.g. "Be careful, what are you trying to do?"

Figure 6.10 Information searching -Stable support module to provide a description of quality feedback

6.4.2.4 How-to-do-it procedure

Helping the TA learn the sequence for using McFeSPA can be achieved by offering help in the form of a how-to-do-it procedure. For example:

- The system helps/scaffolds the TA to provide quality feedback (e.g. there will be a help dialogue displayed/shown when the TA provides the same 'positive feedback' message in the same feedback report.)

- The system helps/scaffolds the TA to process according to the sequence of the systems. Those are ReadStudentSolution module, AnalyseStudentSolution module, GenerateErrorMessage module, OrganiseFeedback module, GenerateFeedbackReport module, ReturnFeedback module.

The how-to-do-it procedural information in McFeSPA is presented in process scaffolding. This type of scaffolding helps the TA understand his/her path

within the system e.g. "Where am I? & What should I do next?". The system keeps the TA's record as a history of the TA's action.

When the TA becomes unsure about what to do next, he/she can press the "Need help" button to ask for help from the system. There will be two choices of help. These are
1) What should I do next?
2) How do I solve the current problem?
In this manner, the TA needs to know what to do next so if he/she decides to select choice 1. Then press OK button. After that, the system will gradually scaffold the TA to complete the next process in terms of the five help levels. If the TAs do not complete the process correctly during the first four levels of help, the final/bottom level of help will present the correct answer that the TA should use to complete the process.

Figure 6.11 How-to-do-it procedure -Process scaffolding – preliminary process scaffolding

The initial scenario of process scaffolding for the process of selecting student's solution, analysing student's solution, organising feedback, and generating feedback reports are similar to Figure 6.11, except the level of help/scaffold, which is described in Figure 6.12 – Figure 6.16. In content scaffolding, it is a type of scaffolding that helps the TA discovers the correct answer. For each step of process scaffolding McFeSPA provides five levels of help as contingent help approaches (Carroll, 2000). These levels of help can be seen from Figure 6.12 - Figure 6.16. In addition, according to feedback design in Chapter 3 we propose our pattern of help for providing quality feedback to the TA that consists of five levels of help as shown in Figure 5.4.

Level 1: {the name of the 1st click of any button/menu item} doesn't work for you. Try again, {TA's name}.
Level 2: {the name of the 2nd click of any button/menu item} is not the next step that you should process. Try again, {TA's name}.
Level 3: {the name of the 3rd click of any button/menu item} is not the next step. Think about what is a suitable button/menu item after this process. Try again, {TA's name}.
Level 4: {the name of the 4th click of any button/menu item} is not quite right. I'll show you what to do next. Try again, {TA's name}.
Level 5: You should click "Select Students' Solution button/menu item"

Figure 6.12 How-to-do-it procedure -Process scaffolding – hint for the process of selecting the student solution module

Level 1: {the name of the 1st click of incorrect file (not file.pl)} doesn't work for you. Try again, {TA's name}.
Level 2: {the name of the 2nd click of incorrect file (not file.pl)} is not the right student's solution. Try again, {TA's name}.
Level 3: {the name of the 3rd click of incorrect file (not file.pl)} is not the right student's solution. Think about what is the suitable file extension that you have to mark. Have another go, {TA's name}.
Level 4: {the name of the 3rd click of incorrect file (not file.pl] is not quite right. Think about what is a suitable file that you have to mark in {the name of course e.g. prolog}. I'll show you what to do next. Have another go, {TA's name}.
Level 5: You should select filename.pl

Figure 6.13 How-to-do-it procedure -Process scaffolding – hint for the process of selecting the correct student's solution file module

The level of help in order to provide the TA with a hint to process and analyse student's solution modules are similar to the level of help in Figure 6.12 except the detail of help level 5 "You should click 'Analyse solution' button/menu item"

Figure 6.14 How-to-do-it procedure -Process scaffolding – hint for the process of analyzing the student's solution module

The level of help in order to provide a hint to the TA to continue the process of generating the feedback report module is similar to the level of help in Figure 6.12 except the detail in help level 5 "You should click "Create Report" button/menu item.

Figure 6.15 How-to-do-it procedure -Process scaffolding – hint for the process of generating the feedback report module

In the process of generating error messages, there will be dialogue to remind the TA if there are no error messages generated by the system. The system will scaffold the TA to reconsider the student's script "Have you reconsidered the student's script?" or "Think about what is the key error. Let us see the student's solution window again"

Figure 6.16 How-to-do-it procedure -Process scaffolding – hint for the process of generating the error messages module

6.4.2.5 Intelligent concern (making sense)

- Reminding the TA when they spend too much time on any one process i.e. the system will prompt the TA into action for each over delayed process.

- Adaptive supporting to provide a level of help and adapt the next help according to the TA action.

- Reminding the TA when the TA has not filled/selected student class, course, assignment number, and marker name.

In the adaptive and adaptable scaffold interface can be seen from the scenario below.

Adaptive support will be executed behind the system in order to help the TA at the appropriate time such as when they get stuck but do not request any help. Thereby, when the TA requests help – as adaptable support- at any process of the system it can predict how to help the TA because the system always keeps a record of the TA's actions. Such a record can be deduced from the previous action in order to help the TA to process the next appropriate action. For example, when the system tells the TA that the student's script has four errors of the same type, the TA is then unsure how to generate a feedback message to add in the feedback report. Then he/she asks for help from the system by clicking the "Need help" button. The system knows that the TA is generating a number of feedback messages for the same number of error types. Thus, the system generates contingent help to scaffold the TA to provide feedback on an important error only once e.g. the help level 1 of this scaffolding is "You should be aware that you are providing too much of the same error type" In other words, the TA does not summarise feedback when the students perform a number of the same error type. The system may also help with scaffolding messages e.g. "don't report the same feedback messages such as the single variable" The system may suggest that the TA directs the student to visit the webpage of the tutorial.

With regard to adaptive support, the system will provide each level of help sequentially. This depends on the TA's profile, for any action that the TA performs the appropriate level of help will be available in the pop up window.

Figure 6.17 Intelligent concern -Adaptive & adaptable support module to provide the level of help and adapt the next help according to the TAs action.

6.4.2.6 Reflecting upon one's own work

Metacognitive scaffolding is a type of scaffolding that helps the TA to perceive his/her own learning through reflection, monitoring, etc. as reflective learning to the TA. This is in order to assess the TA's understanding and the progress through his/her learning process for each phase of the system. The metacognitive scaffolding in McFeSPA can be viewed and will be displayed in the form of a bar chart –called a skill meter. McFeSPA has several aspects that can be metered which are 1) understanding of providing 'positive feedback'; 2) understanding of providing important error feedback; 3) understanding of providing elaborative/detailed feedback; 4) understanding of providing asking questions; 5) understanding of providing a feedback loop; 6) understanding of providing individualised feedback. If there is a skill area in which the TA cannot reach the required level, the bar chart representing the skill will not increase.

For every situation in which the TA completes the appropriate performance relating to giving quality feedback, and the feedback pattern, the skill meter for the particular feedback will be increased alongside a message that explains the event e.g. the scenario in Figure 6.18.

When the TA selects the appropriate feedback pattern the 'positive feedback' skill meter will increase together with a message informing the TA of his performance such as "Well done, you performed well in giving 'positive feedback'. Your skill meter has increased 30%" This message can also be viewed in the skill meter in customise menu item. There will be an explanation in such view panes for every feedback meter below the feedback meter bars.

Figure 6.18 Reflecting upon one's own work -Metacognitive scaffolding – Reporting the TA's performance

Viewing of the skill meter could help the TA to reflect his/her current skill. Other scenarios of metacognitive scaffolding can be seen from Figure 6.19 - Figure 6.25.

In a situation where the TA has marked 10 of a particular student's scripts but the 'important feedback' hasn't been increased, how can the system help the TA to recognise his/her achievement of learning to give feedback? The system will provide a message to the TA such as "It looks like your students perform unique errors, doesn't it? If not, you should go back to see what was inappropriate in your feedback giving process" This situation can happen when the scaffolding mode is set at off, and also when the scaffolding mode is set at on in the case that the user refuses the help offer from the system to process giving quality feedback. This could not happen when the user accepts a help offer from the system because the system always observes the TA's behavior

and helps the TA to provide 'important feedback' to the student when there are more than one of the same kind of error found.

Figure 6.19 Reflecting upon one's own work -Metacognitive scaffolding – Remind the TA of their performance in giving feedback

When the scaffolding mode is on or off, and the system is increasing the skill meter the system also provides a pop up message to let the TA know how they are progressing e.g. "Excellent! Your giving of detailed feedback is progressing."

Figure 6.20 Reflecting upon one's own work -Metacognitive scaffolding – Reflecting on the TA's performance in giving feedback

If a TA views the skill meter 10 times while marking a student's script, the system then gives the TA the following prompt "It's good to check your performance; however, don't spend too much time viewing your skill. It would be better if you view your skill after finishing each feedback report."

Figure 6.21 Reflecting upon one's own work -Metacognitive scaffolding – Remind the TA not to view his/her performance too often.

If the TA marks a script for 10 minutes and none of the feedback skill meters increase, the system then informs the TA that "There might be something wrong in your feedback. Let's check your work again."

Figure 6.22 Reflecting upon one's own work -Metacognitive scaffolding –Remind the TA to check the whole work

The TA processes the error messages whether they are a type of design error, implementation error, or style error. Each error type consists of a specific type e.g. a design error contains four types of error ('Didn't use for 1st letter of variable', 'unused variable', 'unreachable goal', 'non-existent goal'); implement error contains three types of error ('missing cut(!)', 'writing separate symbol (; not;)', 'missing parenthesis'); style error contains two types of error ('missing indentation', 'missing blank line').

If the student does not perform any type of error, the system shows a dialogue box which contains the following question "We did not find any errors from the student's script. Would you like to add any further error messages to the system?" If the TA decides to add any error messages, there will be a prompt from the system to remind the TA "Think about what is a key error. Let's see the student's solution pane again" This message can be removed by the TA by pressing the "stop reminding me" option.

If the student does not perform any design errors, the system will generate a dialogue box with the following question "We have not found any design errors from the student's script. Would you like to add any further design error messages to the system?" If the TA decides to add any error messages, there will be a prompt from the system to remind the TA "Think about what design error will further enhance the system. Let's see the student's solution pane again" This message can be stopped by the TA by choosing the "stop reminding me" option.
However, if the TA decides not to add any error messages, there will be a prompt from the system to remind the TA "If you think that there have been some design problems but you do not want to tell the student directly, it might be better to ask the student to think about what could be the design problem in his/her script". This is a kind of quality feedback –asking a key question. The same dialogues will appear if the student does not perform either any implementation error or any style error.

Figure 6.23 Reflecting upon one's own work -Metacognitive support module to remind the TA to think about the key error

In a situation where the TA generates the final feedback report, the system will remind the TA to check his feedback report with the following prompt "have you double checked the feedback report? You may have missed some useful feedback"

Figure 6.24 Reflecting upon one's own work -Metacognitive scaffolding – Remind the TA to check the given feedback report

If the scaffolding mode is on, and the TA often refuses help from the system and has also been marking the script for 10 minutes without the skill meter of any feedback increasing, the system will inform the TA that "You need improve giving [Type of feedback] feedback"

Figure 6.25 Reflecting upon one's own work -Metacognitive scaffolding – Remind the TA to think about the given feedback

Reporting the TA's performance from the skill meter comes from the principle of informing changes in the skill meter according to the percentage shown of each TA's feedback skill as described below:

0% "Your skill meter hasn't increased, try to check your performance in giving [feedback type] feedback"

1-39% "Good, you are progressing in giving [feedback type] feedback"

40-79% "Well done, you produce a lot of [feedback type] feedback"

80-99% "Very good, you perform very well in giving [feedback type] feedback"

100% "Excellent, you have mastered giving [feedback type] feedback"

As a result of evaluators' comments the principle of informing about changes in the skill meter was developed. See Section 7.5, Chapter 7 for further details.

If the TA forgets to select/fill in the details of the student class, course, assignment number, or marker name, an automatic prompt message will appear to remind the TA. For example "You may have forgotten to fill in/select the assignment number, you need to do this otherwise you can't process the marking."

Figure 6.26 Reflecting upon one's own work -Prompting the TA when the TA has not filled in/selected student class, course, assignment number, marker name.

Helping TAs by interrupting can interfere with the TA's action (Wood, 2001) so the appearance of a highlighted text message could be a signal to inform them whether or not they are performing the right solution path for the right answer at the cognitive level.

When the TAs performs the right solution path, a text message in the bottom pane of the system will appear that will assess the TAs progress for each phase. In contrast, when the TA processes the wrong solution path, a text message in the bottom pane of the system will appear to indicate to the TAs that they are going the wrong way. These messages are colour coded green for the right solution and red for the wrong solution.

Figure 6.27 Reflecting upon one's own work -Assessing the TAs progress for each phase of the system.

Considering the five levels of given feedback in Figure 6.2, Figure 6.12, and Figure 6.13 together with feedback design in Chapter 3 and contingent tutoring (Jackson et al., 1998), we propose the definition of contingent help organised in five levels, as per the examples shown in Figure 5.4 in Chapter 5.

Five levels of help used in McFeSPA differs from a normal help system because using one level of help alone can not help the TA learn -usually a help system provides a direct answer as in the final help level in McFeSPA. So far, in order to achieve our aim some user concerns that could improve the design may be implemented later. In the following, we propose our design principle of the use of scaffolding as a scenario description.

6.4.3 Design Situation: TA Marking Assignments with Semi-Automated marking system

Based on the requirements for McFeSPA, we present the design situation of a TA marking assignments with a semi-automated marking system. We divide the situation into three phases as shown in the following.

The design situation of McFeSPA is similar to abstract level design for usability. Even though we design a number of situations, we selected to implement only the most essential one for the pilot study. The main situation is divided into 3 phases as described below.

6.4.3.1 Phase 1

This semester an inexperienced TA has several responsibilities for three courses, Prolog, Java, and C++. Each course consists of a number of registered student classes, 1st & 2nd year in computer science, 2nd year in Statistics, and 2nd year in Mathematics. For each course the teacher sets the maximum number of assignments at 10. The TA is using a computer support system to select a type of assignment, course, and group of students to mark students' assignment respectively. At that time, the TA selects the Prolog course then the system displays the class that has registered for the course. The TA selects the class of 2nd year computer science students.

Thereafter the TA chooses the 1st assignment. The TA then starts to mark a student's solution by selecting the select 'students' solution' menu item. The student's solution path and file path is displayed at the same time with the student's solution detail on the left hand pane of the screen.

6.4.3.2 Phase 2

The TA selects the 'analyse student's solution' menu item so that the system can automate marking the students' solution as far as it is able and then report all error details to the TA. All the same cases of a particular error type analysed will be grouped together and reported to the TA in brief in a pane under the student's solution pane. Then the TA can double click each error group in the pane to make choices for generating error messages. The system provides a choice for generating error messages to help the TA generate the feedback report. Thereafter it will be up to the TA to provide quality feedback to the student. There are several choices given for the TA to generate the feedback report; however when the TA needs help the system will suggest the appropriate one to the TA. Not only does the system provide the TA the choice of error, but the system also allows the TA to generate his/her own error messages to extend the system. Any error messages generated by the system may be judged inadequate by the TA, so the TA can add more errors as classified by error types of the system and save such messages to be used in the future.

6.4.3.3 Phase 3

The TA can select the 'create feedback report' menu item and the system offers a feedback template for the TA to organise feedback before generating the feedback report. Help/guidance from the system will appear here so that the TA can understand the provision of appropriate feedback pattern to the feedback report. When the template for generating feedback report has been shown, the TA can select 'positive feedback' message that can be either offered by the system or managed (added/updated/ deleted) by the TA. The system allows the TA to organise the feedback message before generating the final feedback report and sending the report to the student via their e-mail address. The TA can edit the

final feedback report in a similar fashion to using a word processor which includes cut, copy, changing font format, etc.

6.4.3.4 All Phases

In this phase, having explained the broad process which the TA goes through, the system provides a variety of ways of taking the interaction forward. While the TA is using the system, there are a number of options in the customisation menu item to help the TA customise the feedback message according to his/her needs. For example:

- The TA can select their favourite animated agent from the customised list.

- When scaffolding is "on", the TA can set the system so that either intervention is by an agent with text only or includes both text and voice or non-intervention but with a pop-up window with text (to appear in every successful step).

- In addition, the system offers the customising of the wording/content (error messages) for the TA to select favourite word/content before analysing the student's solution according to the error messages generated automatically by the system. For example, the TA selects the favourite word before the system generates automatic error messages to the temporary report.

- The TA can customise the report template before generating the feedback report.

- The system allows the TA to add more error messages which are classified by the system error types (design/implementation/style) in which case the TA needs to keep/update/delete such messages.

- To generate a quick feedback report, the TA can also customise the choice of error message consequences to appear in the temporary report from analysing the students' solution (e.g. there are three cases for generating error messages i.e. 1) every same type of errors; 2) only once; 3) ignore). When the TA decides not to generate an explanation for every case of the same type of error message, the TA may not say anything according to such errors. The TA can change his/her default at any time while the system is generating the error messages. For any particular case found the TA can set this default to be shown

permanently for the same students' marking solution. At any stage the TA can see the error messages generated by the system. The consequence of displaying the temporary report can be displayed by the system automatically or by the TA selecting/editing appropriate error messages. This facility is useful when the TA becomes a master in giving quality feedback.

- The TA can select their favourite screen colour for each window pop-up or the dialogue response window.

- The system also provides a view of the student's profile for helping the TA to see the history of student's errors, which is classified by the system error types as well as providing a view of his/her performance via the skill meter.

- The TA can circle a particular error and the system is intelligent enough to find the same type of such errors by the error pattern from the system's database or the TA can add new error patterns that the system is intelligent enough to identify within the system.

6.4.3.5 Scenario of Contingent support depend on the TA's action

The scenario of contingent support depends on the TA's action, in that the TA is out of solution path i.e. the TA does not achieve the cognitive level. Out of system's solution path is the scenario when the TA does the wrong thing i.e. the TA does not get the right answer at the cognitive level. In the situation that the TA responds to an item from the system, trying the functionality offered, setting and changing the customisation, and learning something from the interaction before moving on. When TAs go wrong – do not follow the system's path, we could consider generating an error scenario.

In the situation the system helps/scaffolds the TA to provide quality feedback (e.g. there will be a help dialogue displayed/shown when the TA provides the same 'positive feedback' message in the same feedback report), not only does the system provide many chances for rehearsing, but the system also encourages the TA to think and reflect on what they are doing. If the same error type appears a hundred times (such as between line 1 and line 3: warning! You are missing indentation of the body of predicate run/3; between line 5 and line 8: warning! You are missing indentation of the body of predicate solve/3, ... between 220 line and line 222: warning! You are missing indentation of the body

of predicate final/0) but the TA generates the same error messages for all errors that appear in the system. The system will then display a pop-up help dialogue to scaffold the TA (e.g. important/specific feedback is a kind of quality feedback). Under these circumstances, the system provides many opportunities for practice as well as encouraging the TA to think and reflect on what he/she is doing. Other circumstances can be seen in Section 6.5 (Rule 1- Rule 14).

6.5 Design principles for using scaffolding

The above user concern leads to a question which needs to be answered in our context, that is how to train/help TAs give quality feedback on students programming assignments? With respect to the context of design principle we desire answers to the following questions: What is the role of the TA?; - the answer is that a TA is marking assignments and learning to give quality feedback; How do TAs interact with the environment? –the answer is intervention – intervening when the TA performs any errors or is encouraging giving 'positive feedback' after finishing any process (step); Group or individual work? –the answer is individual work; How does the environment integrate with additional curricular materials? – the answer is reading students script from the monitor; where does it fit in the sequence of learning? – the answer is when the TA produces a final feedback report. According to the design situation in Section 6.4.3, when the TA doesn't follow the system's solution path to the right answer at the cognitive level (does not follow the system's rules), the system provides help to the TA as can be seen from the following principles.

Rule 1. If the TA does not give feedback messages to students to avoid errors that the students made, for example more errors of the same kind than before; same number of errors of the same kind than before; less errors of the same kind than before. Then the system provides a hint to the TA (i.e. contingent help) and the measure of 'individual feedback' and 'feedback loop' will not increase.

Rule 2. If the TA does not give feedback messages to the students to avoid errors that the students made 1st time, then the system provides a contingent help hint to the TA and the measure of 'individual feedback' and 'feedback loop' will not increase.

Rule 3. If the TA does not give 'detailed/elaborative feedback' at the 1st time of finding the error made by student, then the system provides a contingent help hint to the TA and the measure of 'detailed/elaborative feedback' will not increase.

Rule 4. If there are a number of the same kinds of error found, whether such errors happened the 1st time or not, but the TA does not give an 'important/specific feedback' message and indicate to the student that there are more errors like this, then the system provides a contingent help hint to the TA and the measure of 'important/specific feedback' will not increase.

Rule 5. If the TA does not select the best template to generate the feedback report first time around. Then the system provides a hint as contingent help to the TA to help him/her select the best feedback template for generating the feedback report and the measure of 'positive feedback' will not increase. The best feedback template is giving 'feedback sandwiches', giving either 'negative feedback' or error messages between 'positive feedback'.

Rule 6. If the TA does not provide the right student name on the feedback report, then the system provides a contingent help hint to the TA to help him/her give 'individual feedback' with regard to giving the right student's name from the student's marking script in the feedback report and the measure of 'individual feedback' will not increase.

Rule 7. If the TA does not provide 'positive feedback' (or feedback sandwiches) for example the starting detail of 'positive feedback'; the ending detail of 'positive feedback'; the positive detail of the starting 'positive feedback'; the positive detail of the ending 'positive feedback' in the feedback report, then the system provides contingent help hints to the TA to help him/her give 'positive feedback' and the meter of 'positive feedback' will not increase.

In the case of the TA not selecting 'select student's file' menu item while the system is running the TA cannot select either 'analyse' menu item or 'create report' menu item. If the TA chose 'select student's file' menu item without selecting 'analyse' menu item then the TA cannot select 'create report' menu item. How can we help the TA? After the TA selects 'analyse' menu item while the system is processing giving a feedback message e.g. provide choice of

feedback message; add any error messages. If the TA tries to click either any menu item in the menu list or any temporary error feedback pane (design/implementation/style) in which they do not process while the TA is in the process of giving feedback. How can the system help the TA? We propose rules to help the TA who has gone off the right direction as can be seen from the following paragraphs.

Rule 8: If the TA selects 'analyse' menu item before selecting 'select student's file' menu item then the system relays this message "You should select 'select student's file' menu item before selecting 'analyse' menu item"

Rule 9: If the TA selects 'Re-analyse design/ implementation/ style' menu item before selecting 'analyse' menu item then the system relays this message "You should select 'analyse' menu item before selecting 'Re-analyse design/ implementation/ style' menu item"

Rule 10: If the TA selects 'Re-analyse design/ implementation/ style' menu item before selecting 'select student's file' menu item and 'analyse' menu item then the system relays this message "You should select 'student's file' menu item first then select 'analyse' menu item afterwards before selecting 'Re-analyse design/ implementation/ style' menu item."

Rule 11: If the TA selects 'create report' menu item before selecting 'select student's file' menu item then the system relays this message "You should select 'select student's file' menu item before selecting 'analyse' menu item"

Rule 12: If the TA selects 'create report' menu item before selecting 'analyse' menu item then the system relays this message "You should select 'analyse' menu item before selecting 'create report' menu item."

Rule 13: If the TA selects 'create report' menu item before selecting 'select student's file' menu item and 'analyse' menu item then the system relays this message "You should select 'select student's file' menu item first then select 'analyse' menu item afterwards before selecting 'create report' menu item."

Rule 14: If the TA clicks any temporary error feedback pane (design/implementation/style) while in the process of giving feedback by either generating from a choice of feedback or adding any error messages then the

system relays this message "Don't waste time doing other things. You should concentrate on providing error messages".

6.6 Using scaffolding to a different of TA needs

There are complexities of system design concerning various learning style needs, ability ranges, as well as the variation in the amount of content knowledge. As the categorisation of scaffolding into "types" and "interfaces" above, the types of scaffolding can answer the question "what does the scaffolding help to do?" and the interfaces of scaffolding can answer the question "how is the scaffolding presented to the TA?" In terms of interfaces of scaffolding, there are several disadvantages on stable interfaces because they do not address the various TA needs. Although adaptive interfaces can provide varieties of levels of scaffolding according to the TA performance, it is not easy to think about all the possible cognitive paths the TAs might take. In offering scaffolding for various TA skill levels, there is a danger in providing too much, or too little information in the form of scaffolding. For that reasons, offering adaptable scaffold interfaces could help TAs control help according to their needs. Our design employs both adaptive and adaptable interfaces –moderate scaffolding.

6.7 Design of content alongside scaffolding

Most designers have argued that providing scaffolding- assistance- in the learning process is the most difficult aspect of the learning process obtained from the environment. Therefore the integration of different types of scaffolding (i.e. functional, process, content, and metacognitive), the combination of scaffold interfaces – stable, adaptable and adaptive-, and the different levels of scaffolding should be consolidated within the software. We believe that individual TAs have individual differences so it is difficult to anticipate the amount of either scaffolding types or scaffold interfaces. For this reason, our framework could be redesigned as discussed in Chapter 7 and Chapter 9.

Moreover in designing the content incorporated with scaffolding, with regard to verifying and clarifying the TAs understandings, the system could explain why the TAs choose an incorrect answer (from a choice list). Thereafter, the system

could allow the TAs to perform the process independently. However, sometimes we could take into account ways to prevent a TAs' action going the wrong way by a stopping rule. Finally, the system could support the TA to contribute knowledge of how to give quality feedback to the students; however, this might be difficult to do because it depends upon the TA's skill, knowledge of using the tool and knowledge of giving quality feedback i.e. learning experience -knowledge acquisition.

6.8 Summary

In this chapter, we have produced a general design in helping people learn to give quality feedback. To implement the design, we have selected the design scenarios to create a scaffolding system which will be efficient for testing the hypotheses (in Chapter 1) for which the Pilot study is intended. Most hypotheses are concerned with testing how various type of scaffolding in our system can help the users improve their knowledge about giving good feedback.

Thus, we presented the interface designs for scaffolding (stable, adaptive, and adaptable) for each type of scaffolding (functional, process, content, metacognitive, interpersonal). Most scenarios are derived from the main requirement of the system as presented in this chapter. The key implementation work includes the development of

- functional scaffolding (e.g. Figure 6.8)
- content scaffolding (e.g. Figure 6.2, Figure 6.3)
- metacognitive scaffolding (e.g. Figure 6.18, Figure 6.20, Figure 6.23, Figure 6.24);
- stable scaffold interface (e.g. Figure 6.10);
- adaptive scaffold interface (e.g. Figure 6.9, Figure 6.17);
- adaptable scaffold interface (e.g. Figure 6.17);
- interacting with environment (e.g. Figure 6.5, Figure 6.8).

For adaptable scaffolding, we have chosen to let the user control the amount of scaffolding by means of a simple on/off switch. The alternative would have involved developing mechanisms to manage the degree of scaffolding available during the course of the interaction. In the implementation, we provide pop up buttons in the "offer help" interface that ask the user whether he/she needs help.

In addition, to achieve the requirements of the system that are judged to be essential, it is important to implement Rule 1- Rule 7 carefully so we can reliably measure the TA's improvement in terms of giving quality feedback as well as to help him/her learn how to improve his/her skill.

Nevertheless, it has been decided that some design scenarios, which are not judged to be necessary for the system at this time, will not be implemented for the pilot study. For example, process scaffolding (e.g. Rule 8-15, Figure 6.11 - Figure 6.14, Figure 6.15); sending an electronic feedback report to the student; interacting with environment (e.g. Figure 6.4, Figure 6.7).

To achieve an efficient pilot study by implementing the essential scenarios (described above), the animated agent has not been implemented in the current system. In addition, it has not been judged essential to implement the customisation of the consequence of error messages because we aim to help the TA learn to give feedback rather than just generate a quick feedback report. Besides, customisation of the interface is not a major requirement of the system (e.g. setting the user's favourite screen color), so this has not been implemented.

To sum up, in this chapter we have presented a general design for a system to help people learn to give quality feedback. The implementation of the system of the interface and usability testing are described in Chapter 7. The evaluation of the learning environment is described in Chapter 8. Thus, we believe that we have provided a useful contribution for people who wish to build similar systems and would like to use the results of this chapter as a starting point.

Implementation & Usability evaluation

7.1 Introduction

According to the Scenario-Based Scaffolding System Design in Chapter 6, this chapter presents the implementation of McFeSPA derived from analyzing the necessary scenarios from the previous chapter and the evaluation of the usability of McFeSPA. In order to evaluate the usability, learnability, and effectiveness of the scaffolding system in helping people provide quality feedback and at the same time help people learn to provide quality feedback, the program platform being used to create a prototype of McFeSPA was developed by using Microsoft Visual Basic. While not as sophisticated a language as C++ or Java, it does nonetheless, allow an object-oriented, agent-based style of programming that can handle message passing and data tracking.

7.2 Interface Design

This section discusses the interface design, of McFeSPA as follows:

The Main interface of McFeSPA (see Figure 7.1) consists of a menu area, 'General Detail' area, 'Show Student's Solution' area, 'Analyse Student's Solution' area, and 'Current Error/Weakness in Feedback Report' area. McFeSPA was presented as a menu interface (as referred to in Chapter 6). There are seven menus in the main interface. These are 'File', 'Create report', 'Glossary', 'View', 'Customise', 'About Me', and 'Exit' menu. The 'General Detail' area displays the general detail for each student's feedback report which consists of marker's name, marking date, assignment number, course, class, student's name, student's registration number, module, and student's solution file.

The **'Show Student's Solution'** area displays the student's script[16]. This area shows the student's solution after selecting the 'File' menu and the 'Select student's file' menu item. The current version of the 'Show Students Solution' area was developed in accordance with suggestions from the evaluators (see Suggestion 13, Appendix I) to present the line number in front of each student's line of script.

The **'Analysed student's solution'** pane shows brief error messages generated by the system after selecting the 'File' menu, then the 'Analyse' menu item. The 1st column is the number of the error found (of the same error type), the 2nd column is the error type, and the 3rd column is the name of the error type. The 'Current error/weakness in Feedback report' area (consists of three sub-panes: Design, Implementation, and Style) shows all error messages derived from the user's decision in generating the error messages from the brief error messages in the 'Analysed student's solution pane' (via 'Choices for More Errors', 'Choices for One Error', and 'Taking the history of student's errors into account' interface); or add extra error messages (via 'Add Extra Design/Implementation/Style Errors' interface) in each pane. The details of any temporary feedback message generated will be displayed in each pane for each error type (Design/ Implementation/ Style). The users can also add further error messages into the 'Current error/Weakness in Feedback report' area using the button next to each error type (Design/ Implementation/ Style) area of the main interface of McFeSPA.

- The **'File'** menu contains two menu list items: 'Select student's file' and 'Analyse' menu items. The 'Select student's file' menu item allows the users to open the student's solution file. In the current version, there is only one type of file extension, which is the prolog file extension –'.pl'. The 'Analyse' menu item facilitates the users to analysis of the student's file by automatically generating a list of brief error messages and placing them into the 'Analysed student's solution' pane. Each list of brief error messages contains three columns displayed in the lower left pane of the screen provided McFeSPA analysed the script and found any errors. The 1st column is the number of the same error type found, the

[16] Script is a student's solution that is kept in a file. This thesis assumes that the script is the 2nd submission of a student's solution on the same assignment.

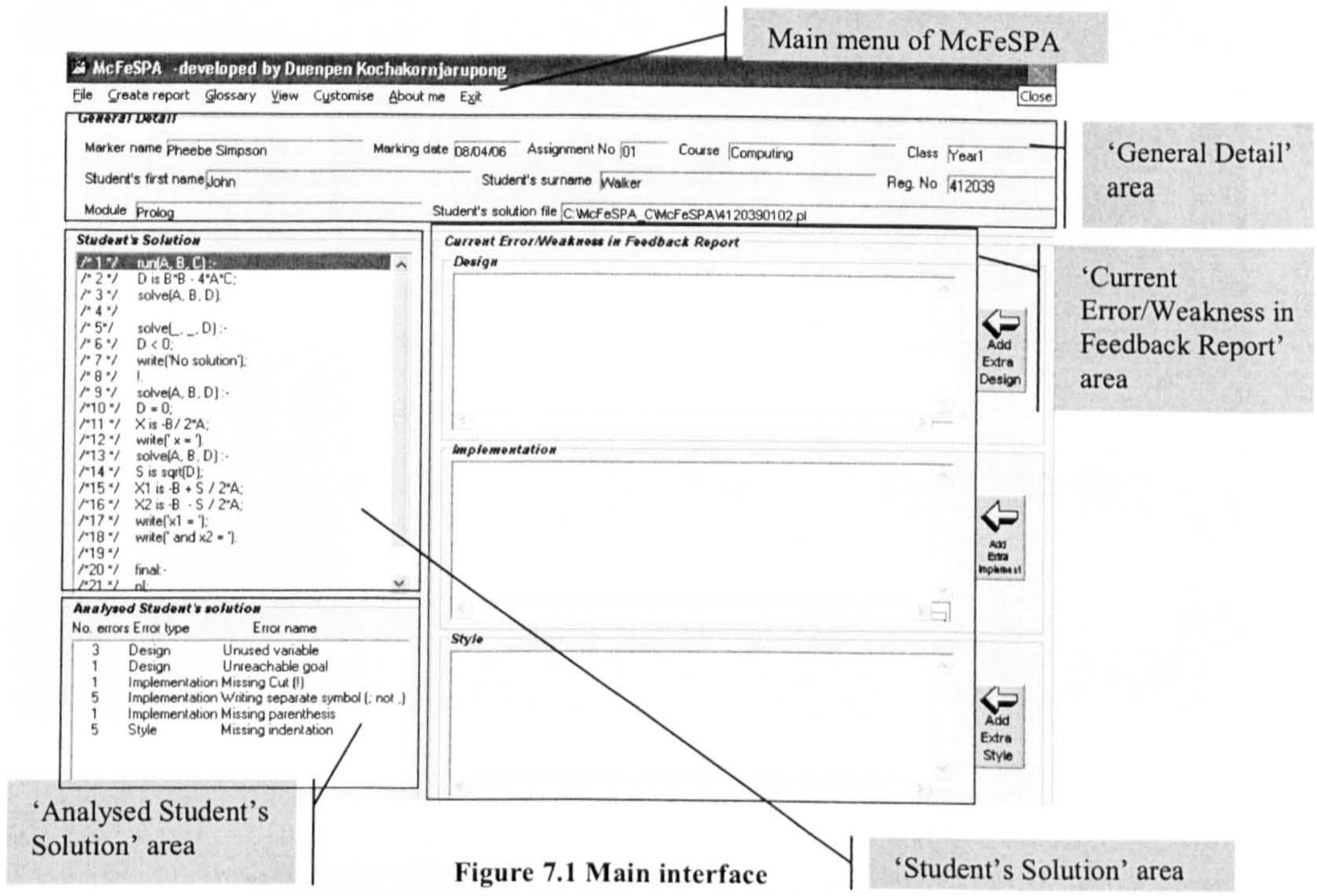


Figure 7.1 Main interface

2nd column is the error type (These errors are design, implementation, and style errors), and the 3rd column is the name of the error type. Once the users click on a list item, which the 1st column (number of errors) is 1, the 'Choices for One Error' interface (see Figure 7.20) will be displayed. In other words, the 'Choices for More Errors' interface (see Figure 7.18) will be shown if the number of errors is more than one. If no errors are generated by McFeSPA, the 'Add any error messages' interface will be displayed. If there is no design/ implementation/ style error generated by McFeSPA, the 'Add Extra Design/Implementation/Style Errors' interface will be presented (e.g. 'Add Extra Design Errors' interface in Figure 7.24).

- The '**Feedback Template**' interface (see Figure 7.2) will be displayed after the 'Create Report' menu is activated. This interface consists of six templates for generating a feedback report. Each template has a different feedback pattern for generating a feedback report. If the users are in the scaffold-on mode, McFeSPA will help the users to select the best feedback template first then organise feedback and finally generate the final feedback report. Once a template has been

selected, the particular 'Create Feedback Report' interface (see Figure 7.25-7.30) for the selected template will be displayed.

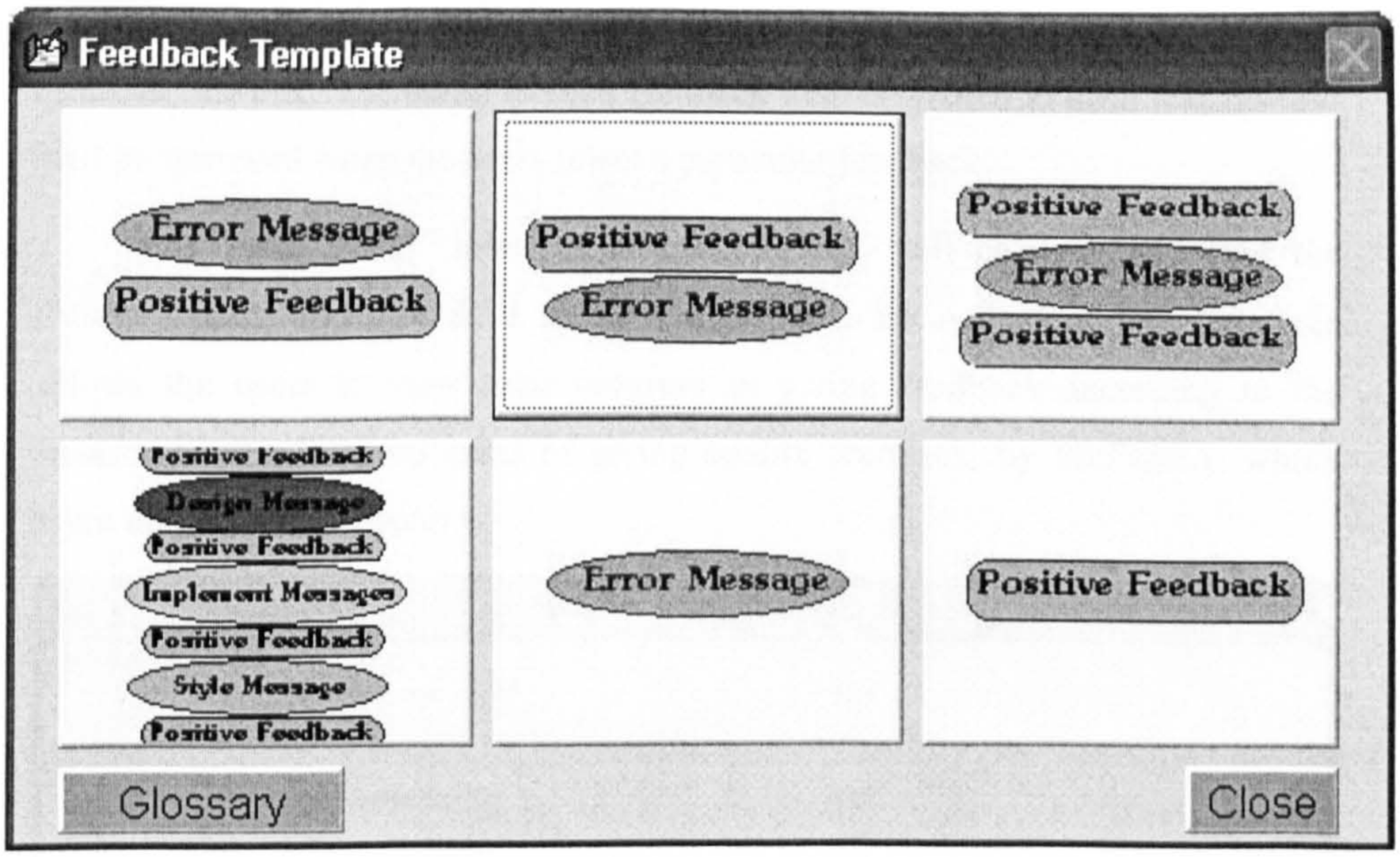


Figure 7.2 'Feedback Template' interface

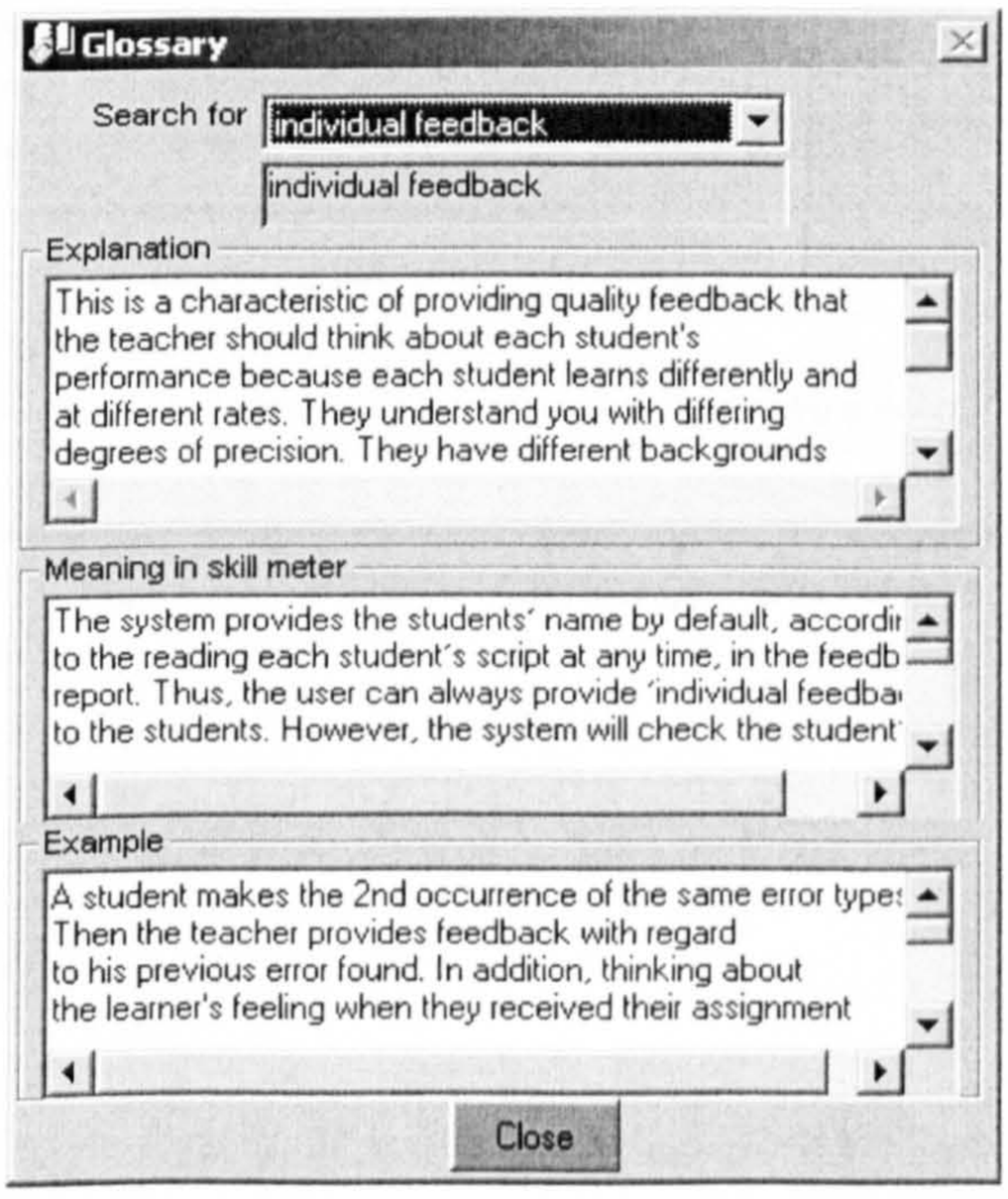


Figure 7.3 'Glossary' interface

- The **'Glossary'** interface (see Figure 7.3) will be displayed after the

'Glossary' menu has been selected. This interface facilitates the users to search the explanation of quality feedback, the meaning of giving quality feedback in the skill meter, and the example of quality feedback by selecting the needed feedback from the list box. The detail of each feedback item is written in each text file and will be retrieved when the users select a particular feedback.

- The '**Skill meter**' interface (see Figure 7.5) will be displayed after the 'View' menu, and the 'Skill meter' menu items are activated. This interface allows the users to view their progress in giving feedback according to the measurement of various areas of giving quality feedback, by McFeSPA, which were described in Chapter 6.

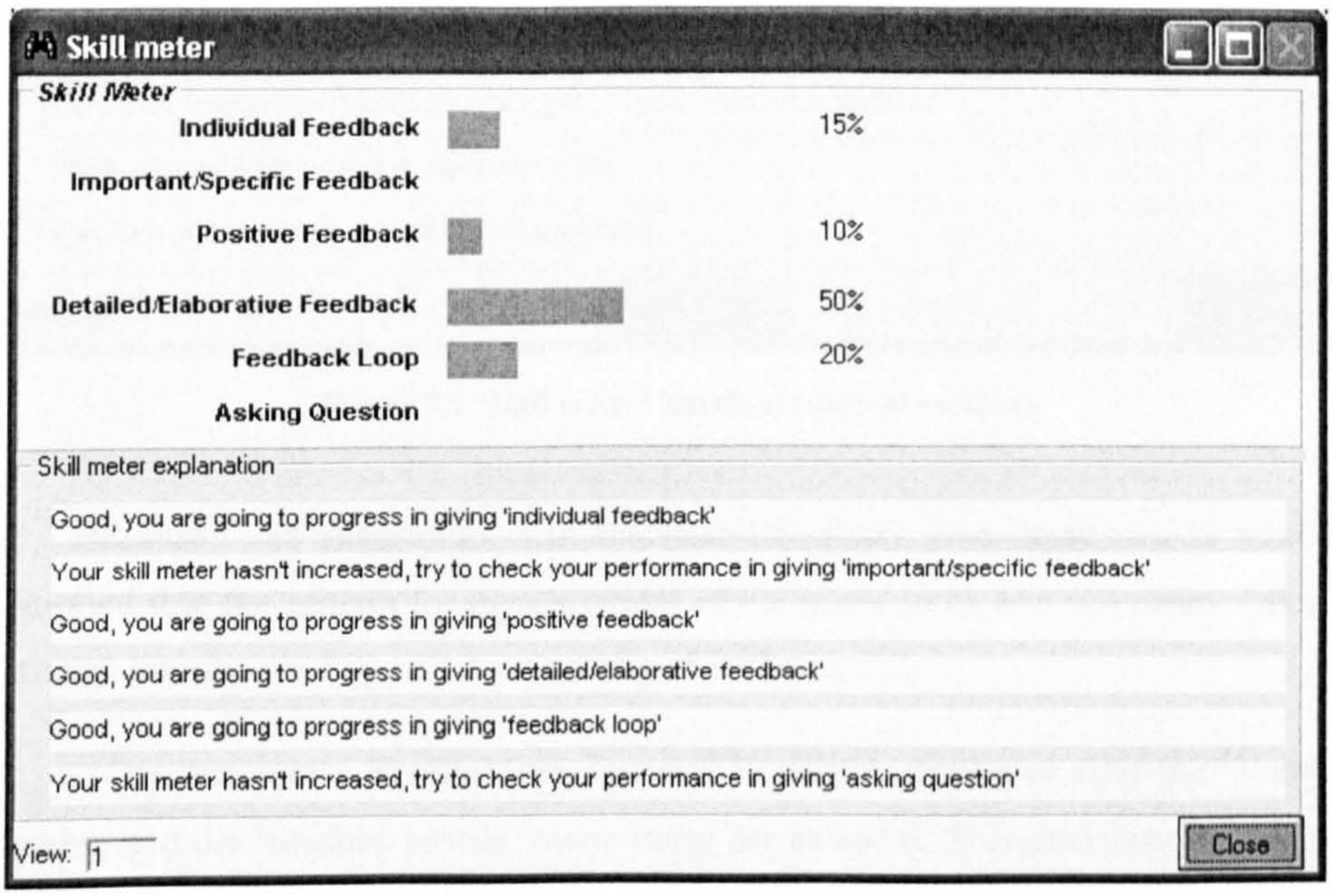


Figure 7.4 'Skill meter' interface (previous version)

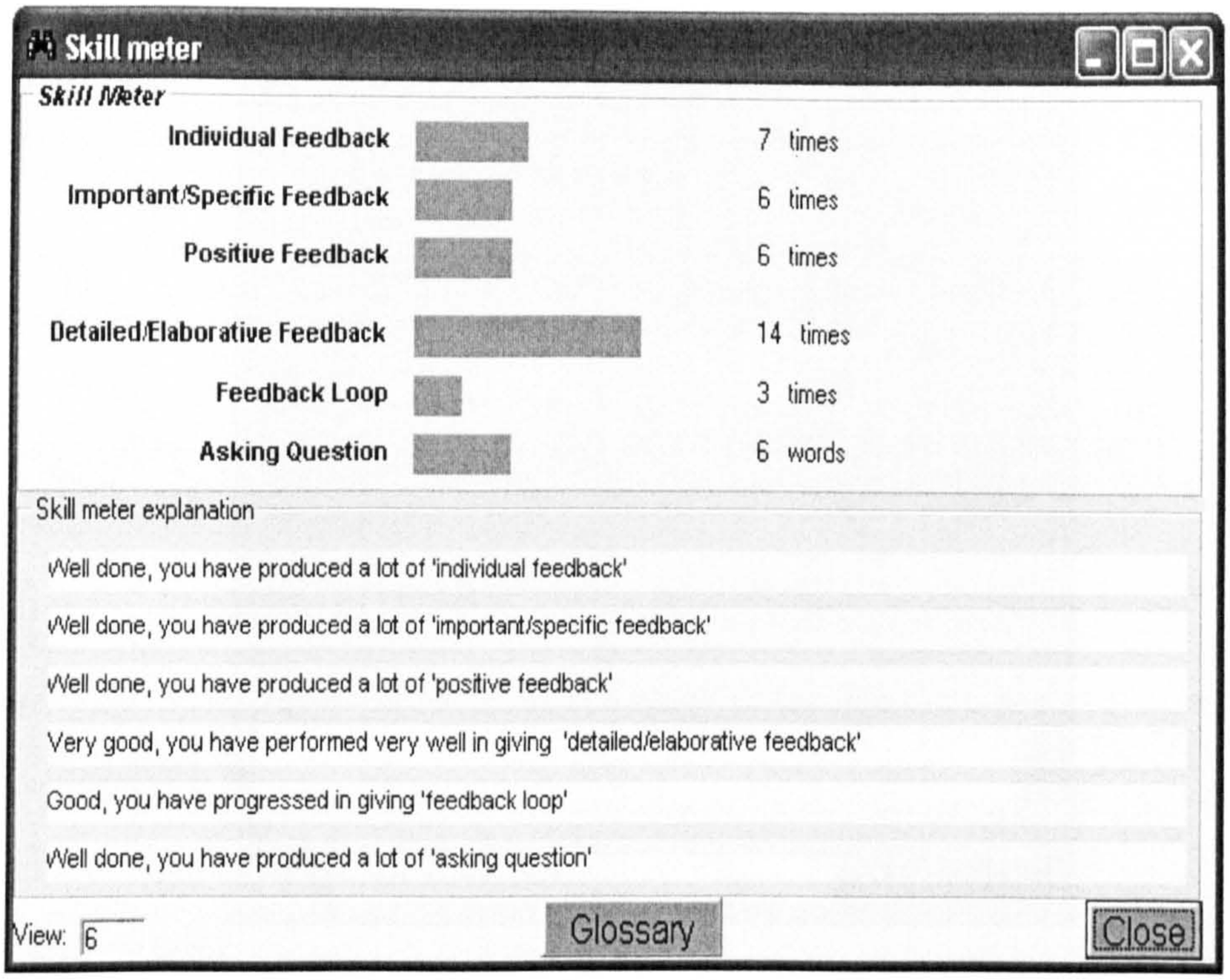


Figure 7.5 'Skill meter' interface (current version)

Because of suggestion 7 in the usability study (see Appendix I) which implied that the criteria for measurement of the users' skill was not clear, 'Skill meter' interface in Figure 7.4 was changed to Figure 7.5.

- The **'Student's Profile'** (see Figure 7.7) will be displayed after the 'View' menu, and the 'Student profile' menu items are selected. This interface facilitates the users to view each student's record regarding the history of performing errors associated with three error types (Design/ Implementation/ Style, see Chapter 4) which are automatically generated by McFeSPA. The users can add any comments to the details already displayed.

The 'Student's profile' interface in Figure 7.6 was updated to Figure 7.7 .The 'Display a student profile' interface was adjusted as a result of suggestions 4 and 30 of the usability study (see Appendix I) which implied displaying a particular student's profile of a student who was marked i.e. displaying a student's profile which was needed to take into account the history of student's errors that were useful rather than displaying all the errors.

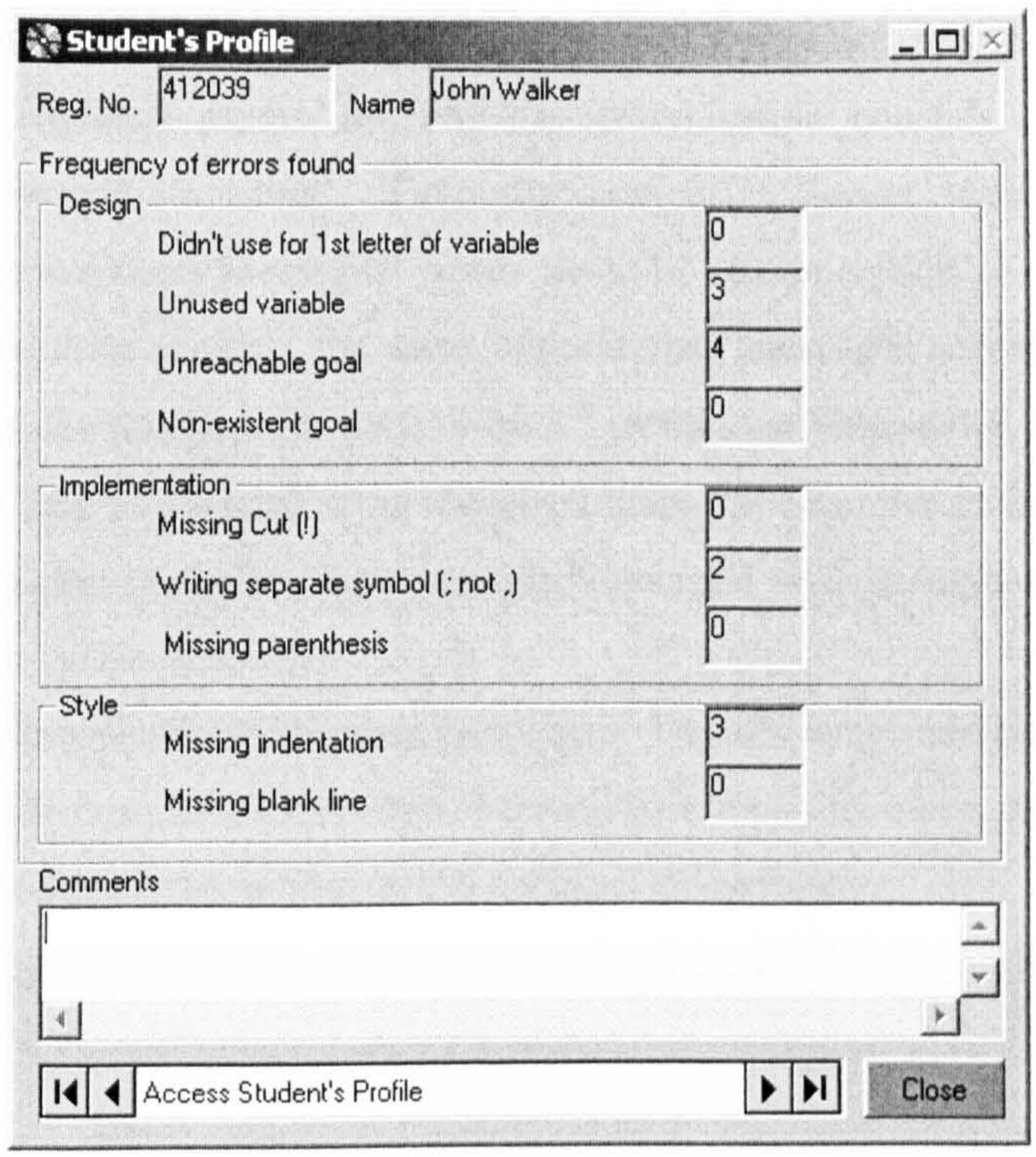


Figure 7.6 'Student's Profile' interface (previous version)

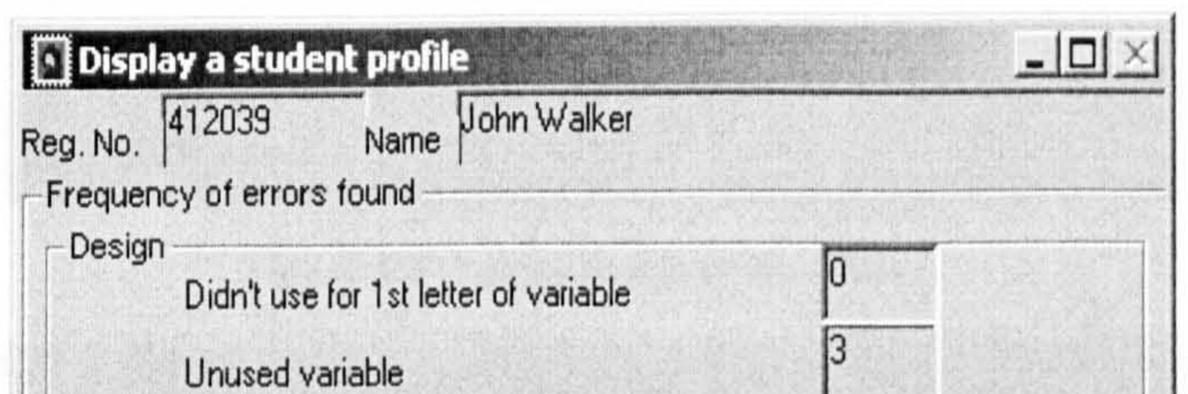


Figure 7.7 'Student's Profile' interface (current version)

- The **'Customise'** menu supports the users to customise McFesPA before generating a feedback report. McFeSPA's customisation consists of 'Setting scaffold', 'Favourite wording', 'Favourite content', 'Report template', and 'Manage error/weakness messages' menu item. In customization of 'Favourite wording', 'Favourite content', the users can edit error messages according to the error pattern of the feedback message at the 2nd level (see Section 4.6, Chapter 4) This message like the original error messages from the compiler or the warning message from the compiler. Any accepted changes will be updated in the customize table on the database.

- The **'Setting Scaffold'** interface (see Figure 7.8) allows the users to define a supporting mode from McFeSPA. Once McFeSPA is set either on or off, McFeSPA will update this setting on the database immediately.

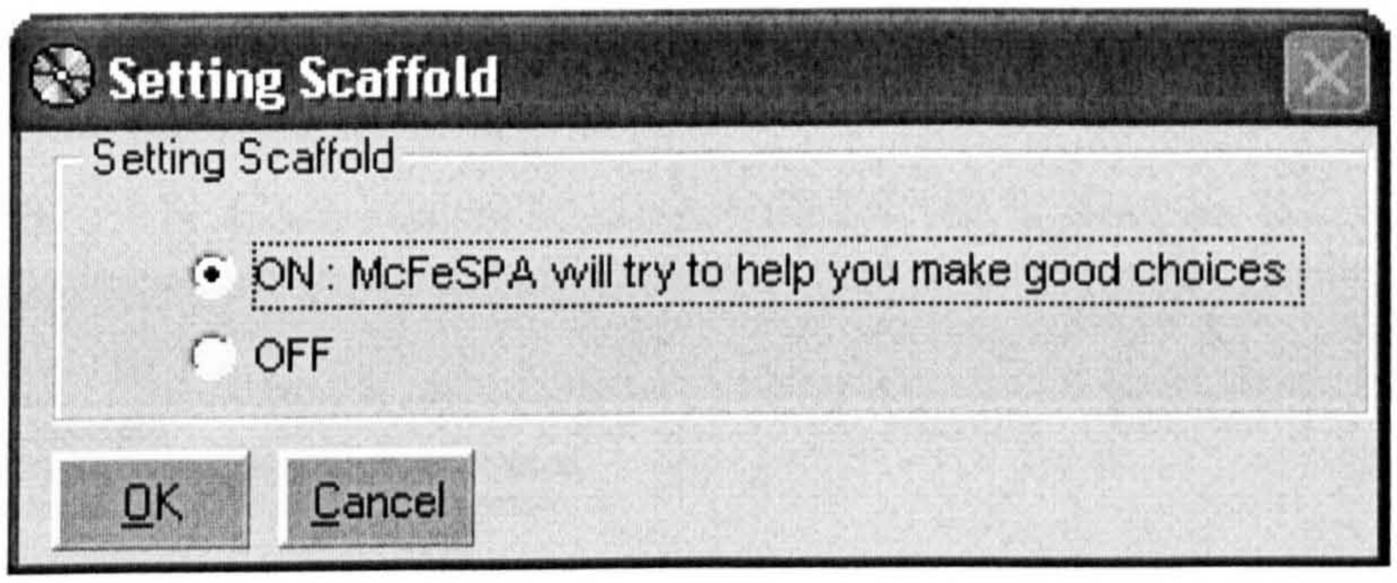


Figure 7.8 'Setting Scaffold' interface

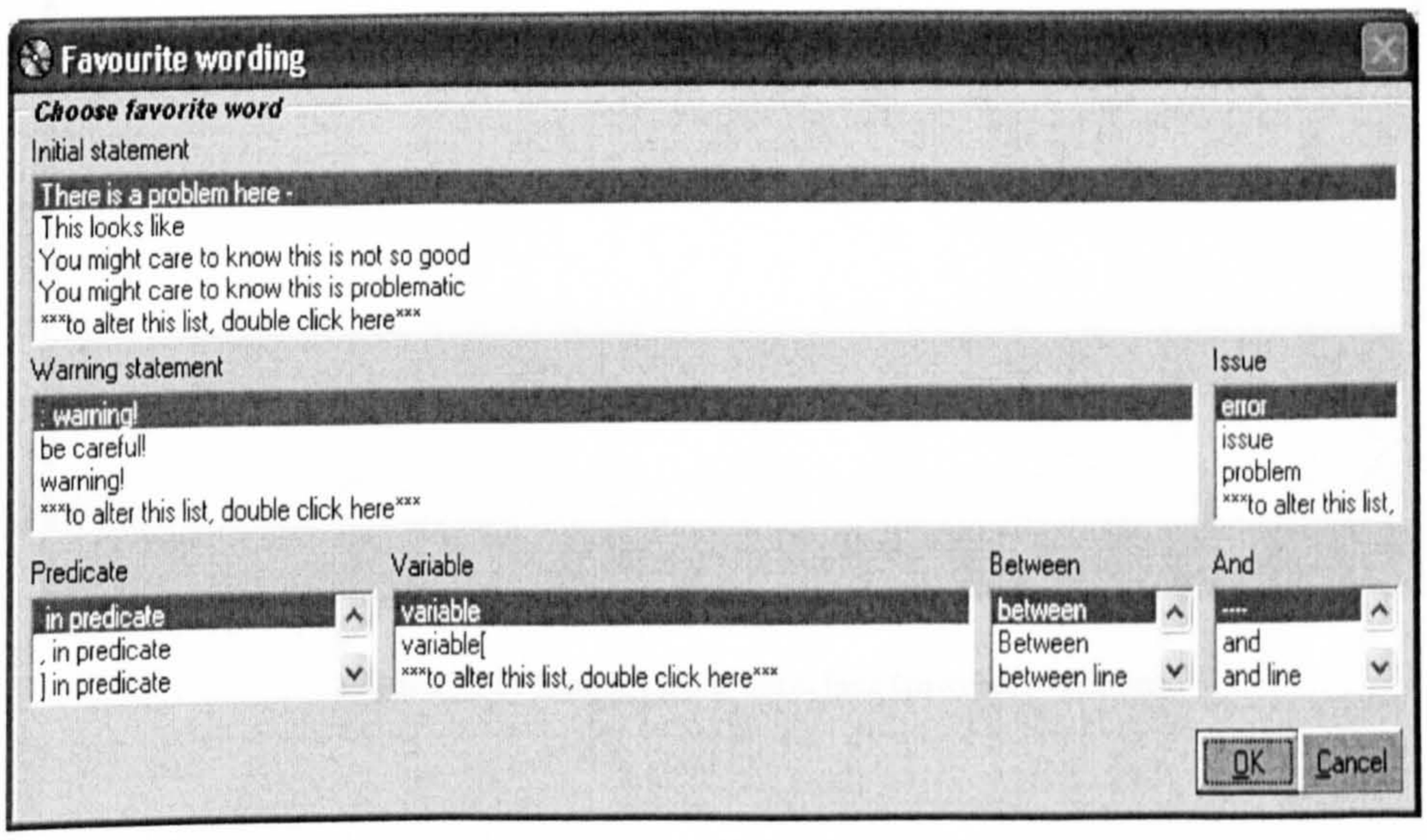


Figure 7.9 'Favourite wording' interface

- The **'Favourite wording'** interface (see Figure 7.9) consists of seven list boxes featuring words extracted from the pattern of the feedback message at level 2 (see Section 4.6, Chapter 4). Each list box contains a choice of several list items that the users can select for customization. The selected list item will appear in the error messages that are automatically generated by McFeSPA. The users can insert a new favorite word to any item lists by double clicking the bottom list item. That is '***to alter this list, double click here***'. The users can also update/delete any list item by double clicking the bottom list item.

- The **'ManageData'** interface (see Figure 7.11) will be displayed after double clicking the bottom list of each list box. This interface facilitates the users to insert/update/delete the selected list item in the 'Favourite Wording', 'Favourite Content', and 'Report Template' interface. The 'ManageData' interface in Figure 7.10 was updated to Figure 7.11 as a result of suggestion 11 in the usability study (see Appendix I) that indicated the buttons' position was not clear.

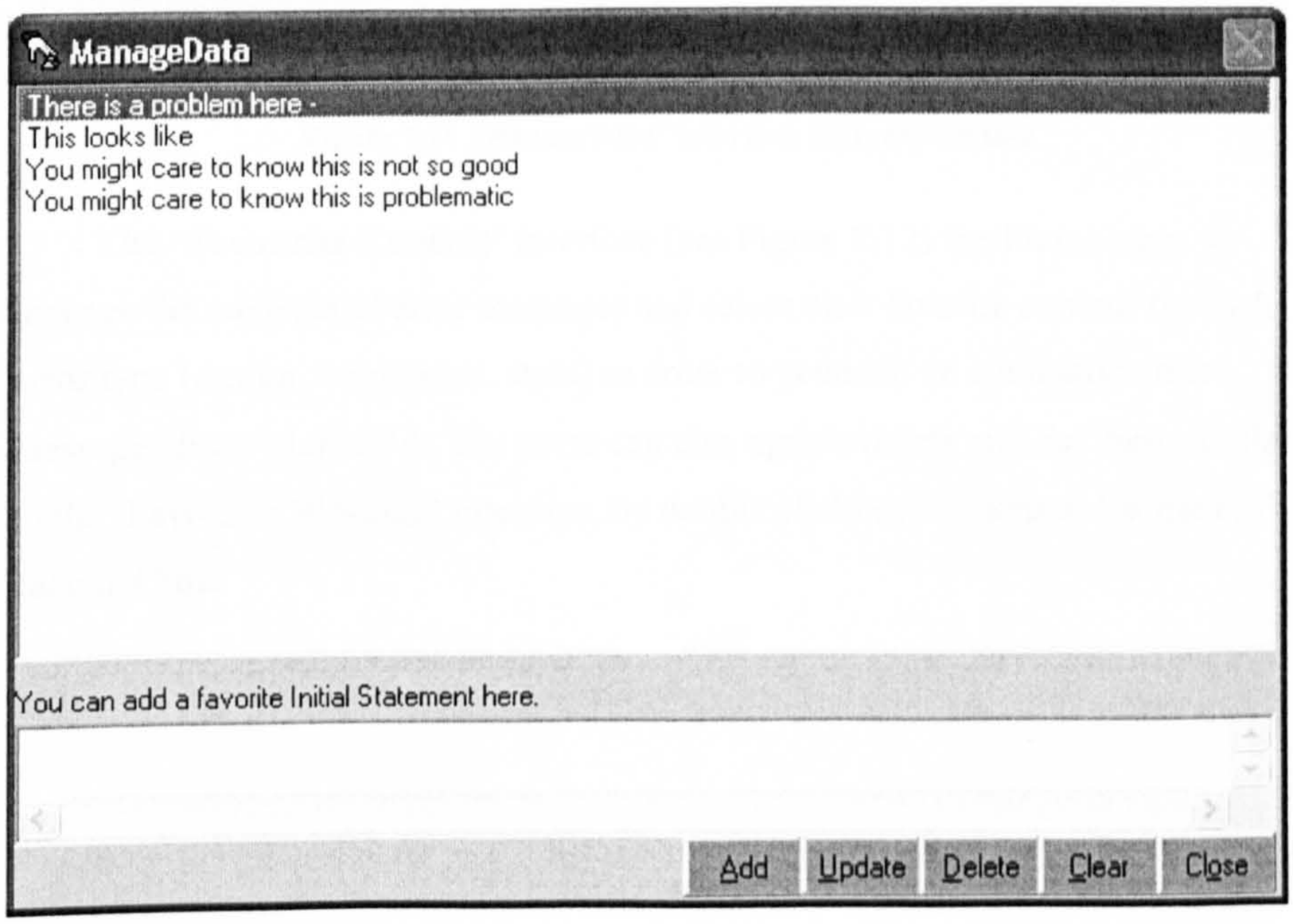


Figure 7.10 'ManageData' interface (previous version)

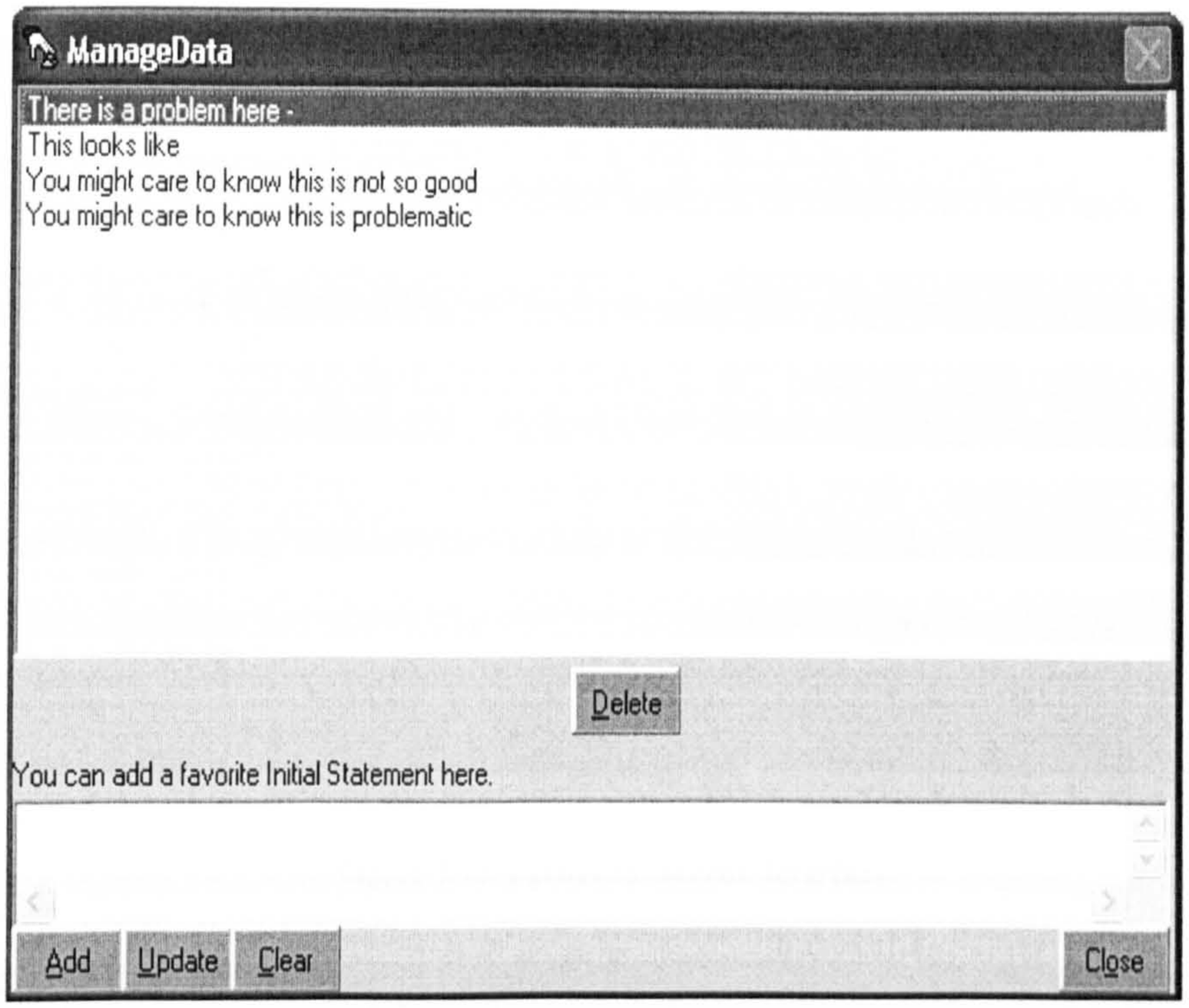


Figure 7.11 'ManageData' interface (current version

- The **'Favourite Content'** interface (see Figure 7.12) facilitates users to manage the contents of error messages and select their favorite content for each error type (design, implement, style) in order to generate an automatic error messages from McFeSPA. The users can also update/delete any list item, similar to the 'Favourite Wording' interface, by double clicking the bottom list item of each list box.

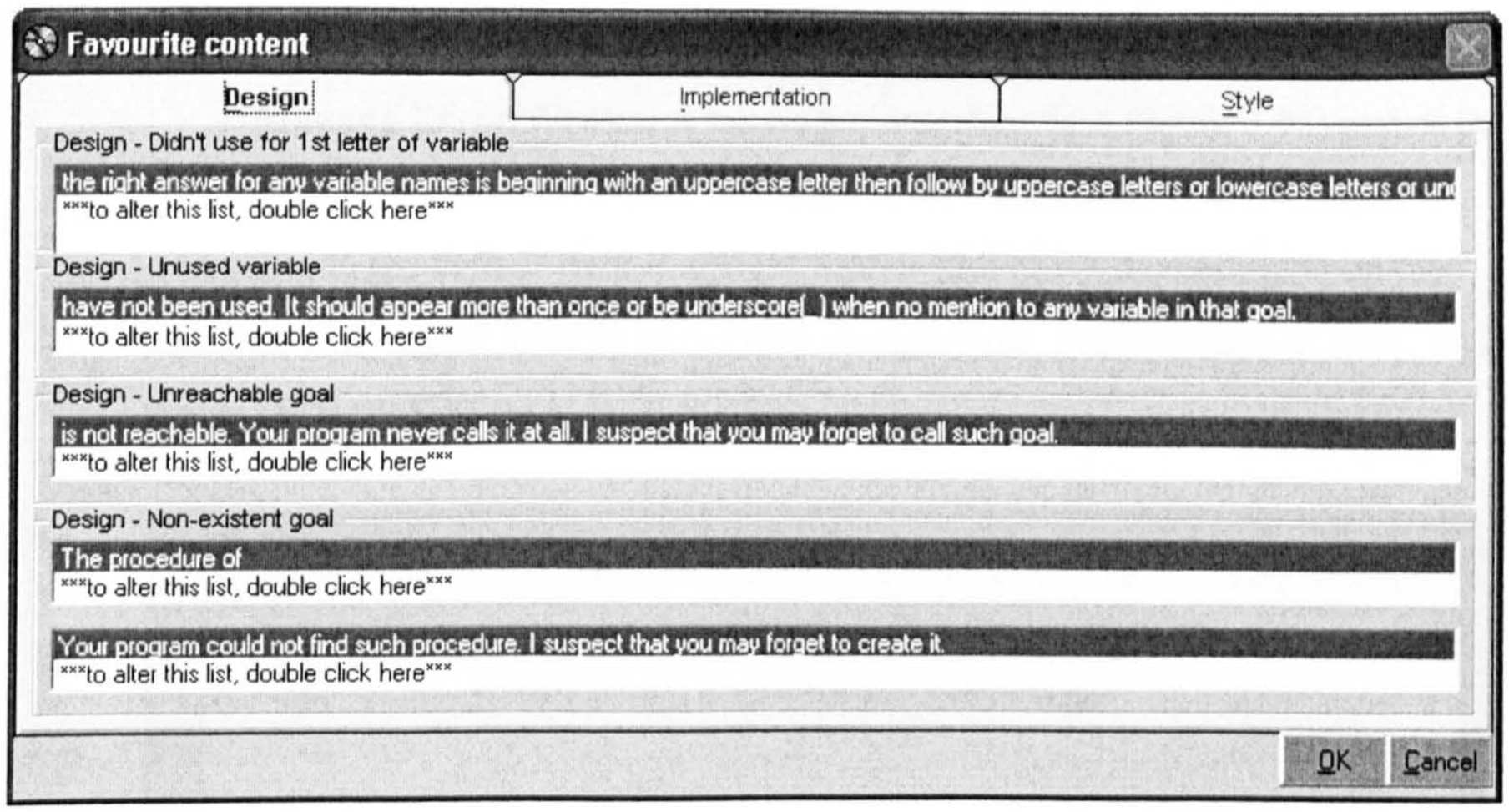


Figure 7.12 'Favourite content' interface

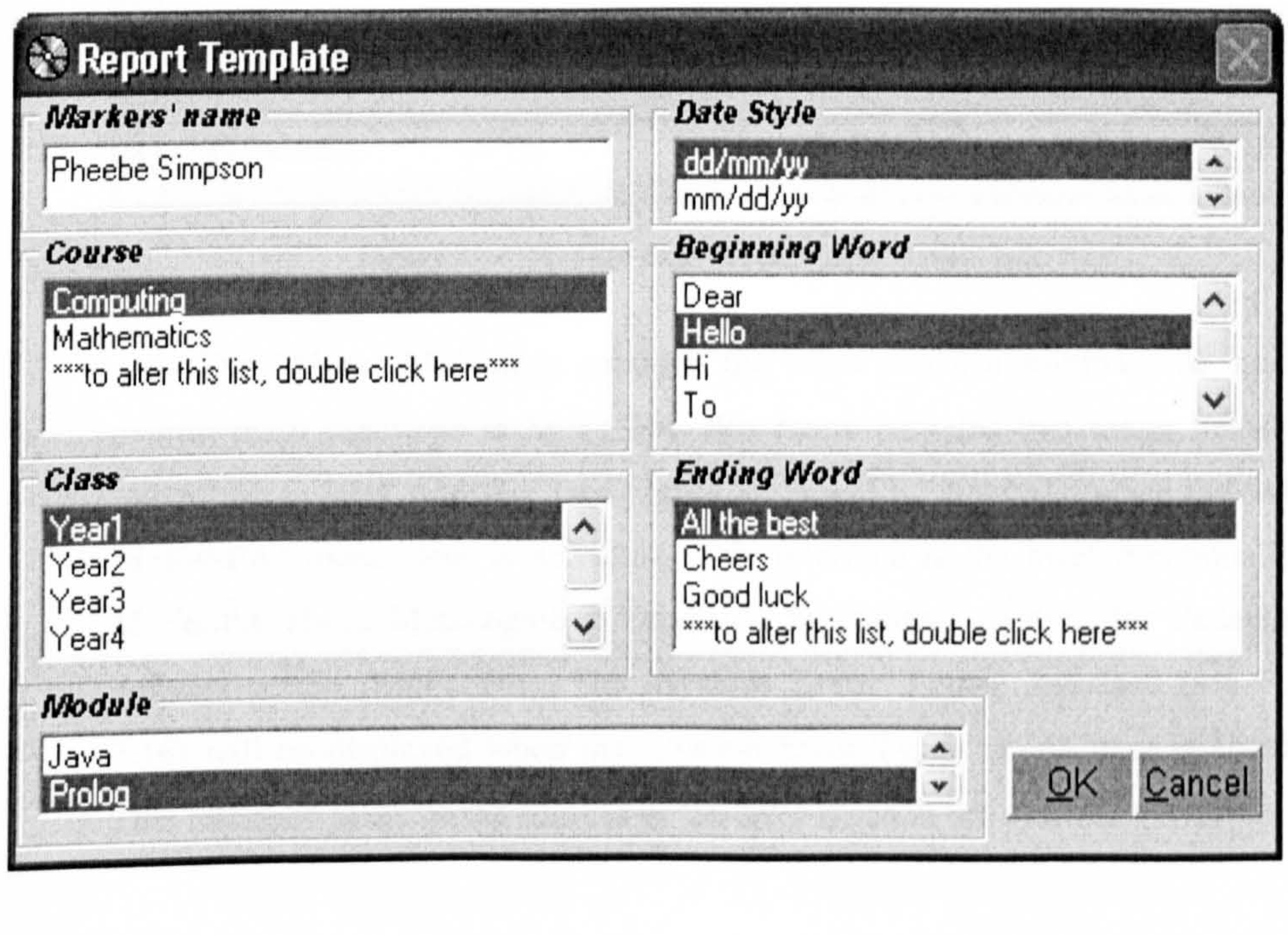


Figure 7.13 'Report Template' interface

- The **'Report Template'** interface (see Figure 7.13) supports the users to redefine the header and footer of the feedback report. The users can also

update/delete any list item, similar to the 'Favourite Wording', and 'Favourite Content' interface, by double clicking the bottom list item of each list box.

- **'Manage Error/Weakness messages'** interface (see Figure 7.14) assists the users to edit the elaborative error messages level 2 (see Section 4.6, Chapter 4) according to the default error messages further McFeSPA depending upon their decision with regard to three error types -design error, implementation error, style error- by McFeSPA.

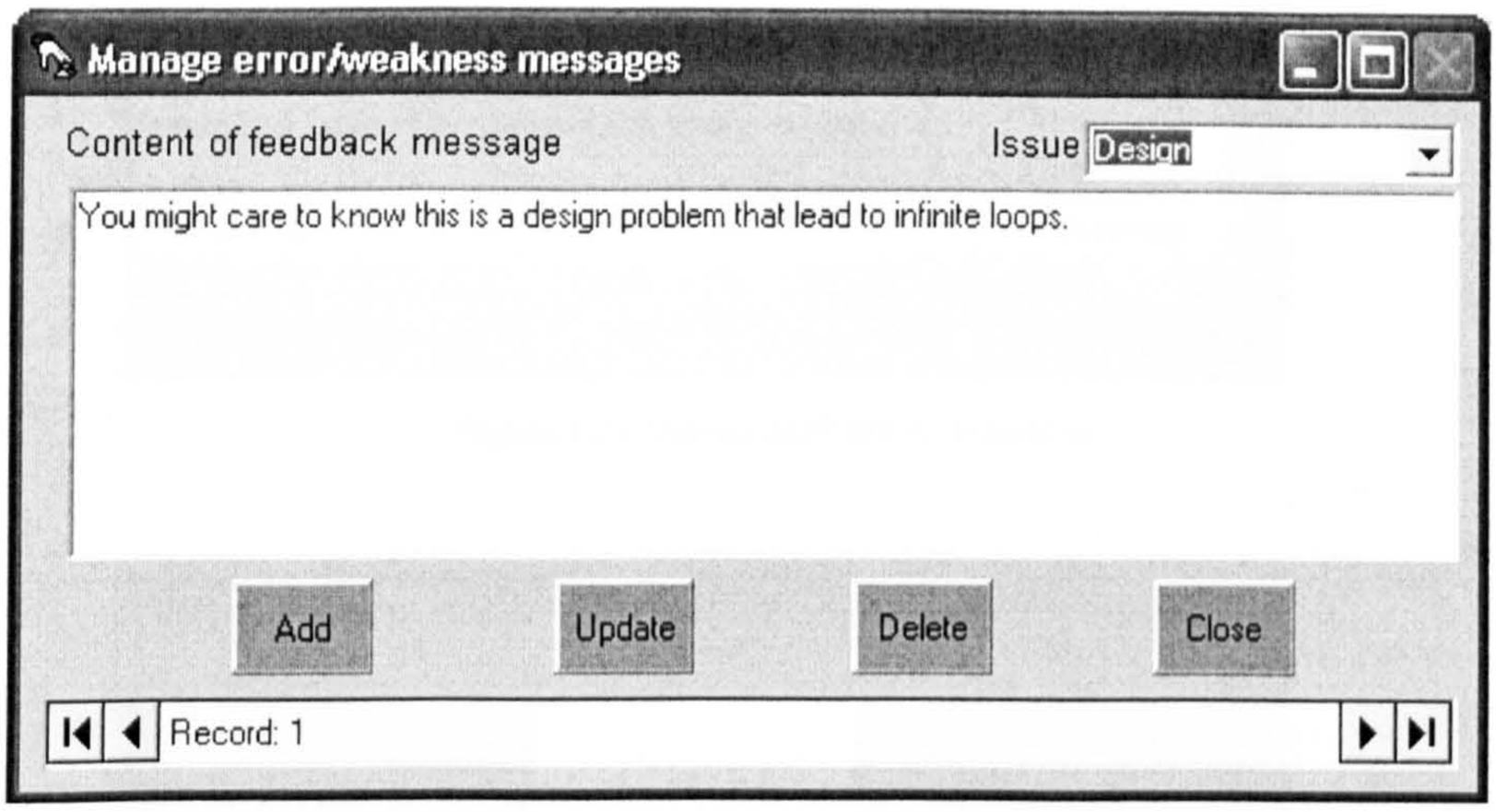


Figure 7.14 'Manage error/weakness messages' interface

- The **'About Me'** menu supports the users to see a summary of the brief overall main functions in McFeSPA, this menu contains two menu items. The 'About McFeSPA' interface (see Figure 7.15) will be displayed when the '**About McFeSPA**' menu item is activated. This interface is the brief functionality of McFeSPA about Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship (McFeSPA). The '**System Error Types**' interface (see Figure 7.16) will be displayed when the 'System Error Types' menu item is activated. This interface presents the sources of the error types in the system.

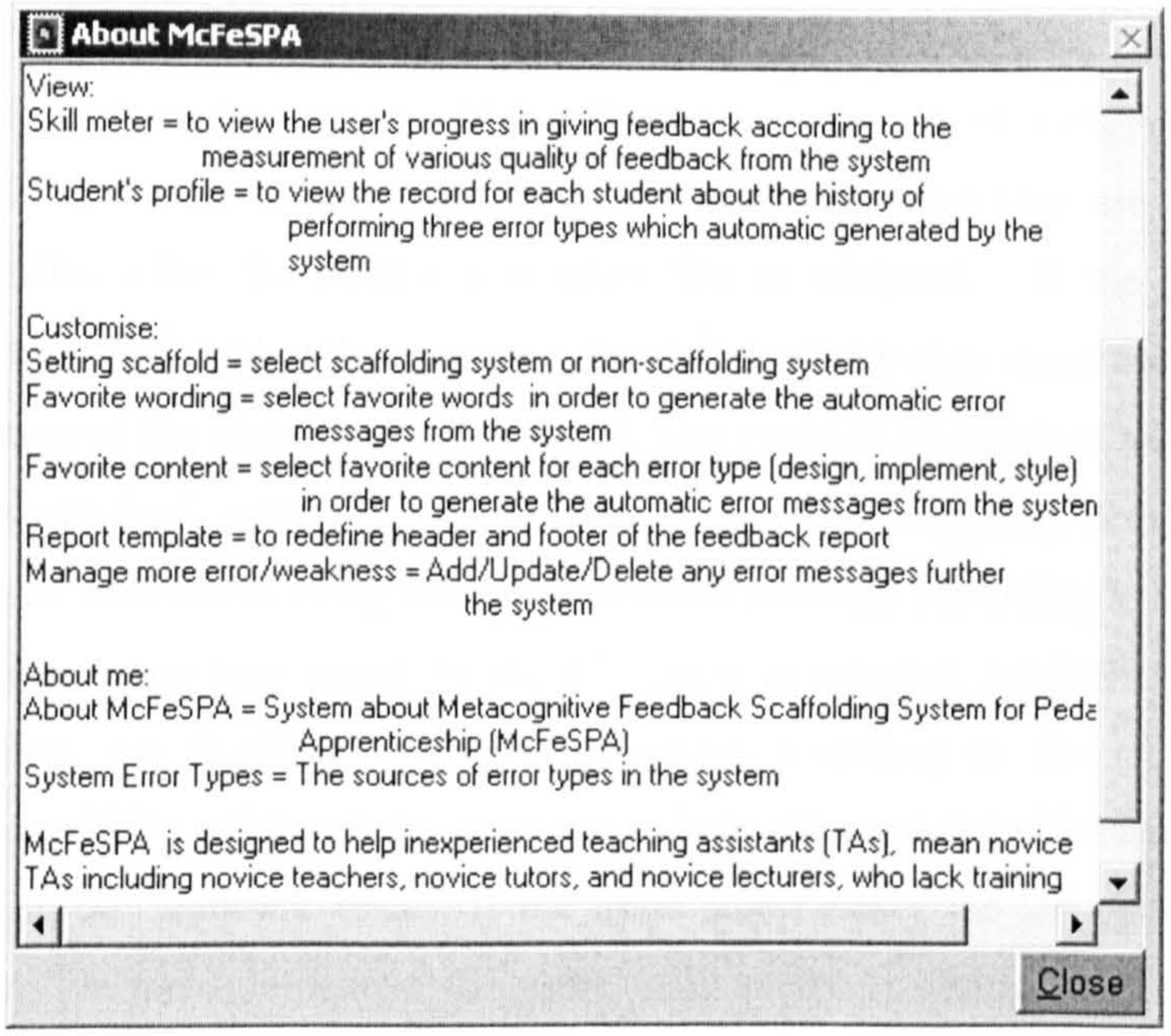


Figure 7.15 'About McFeSPA' interface

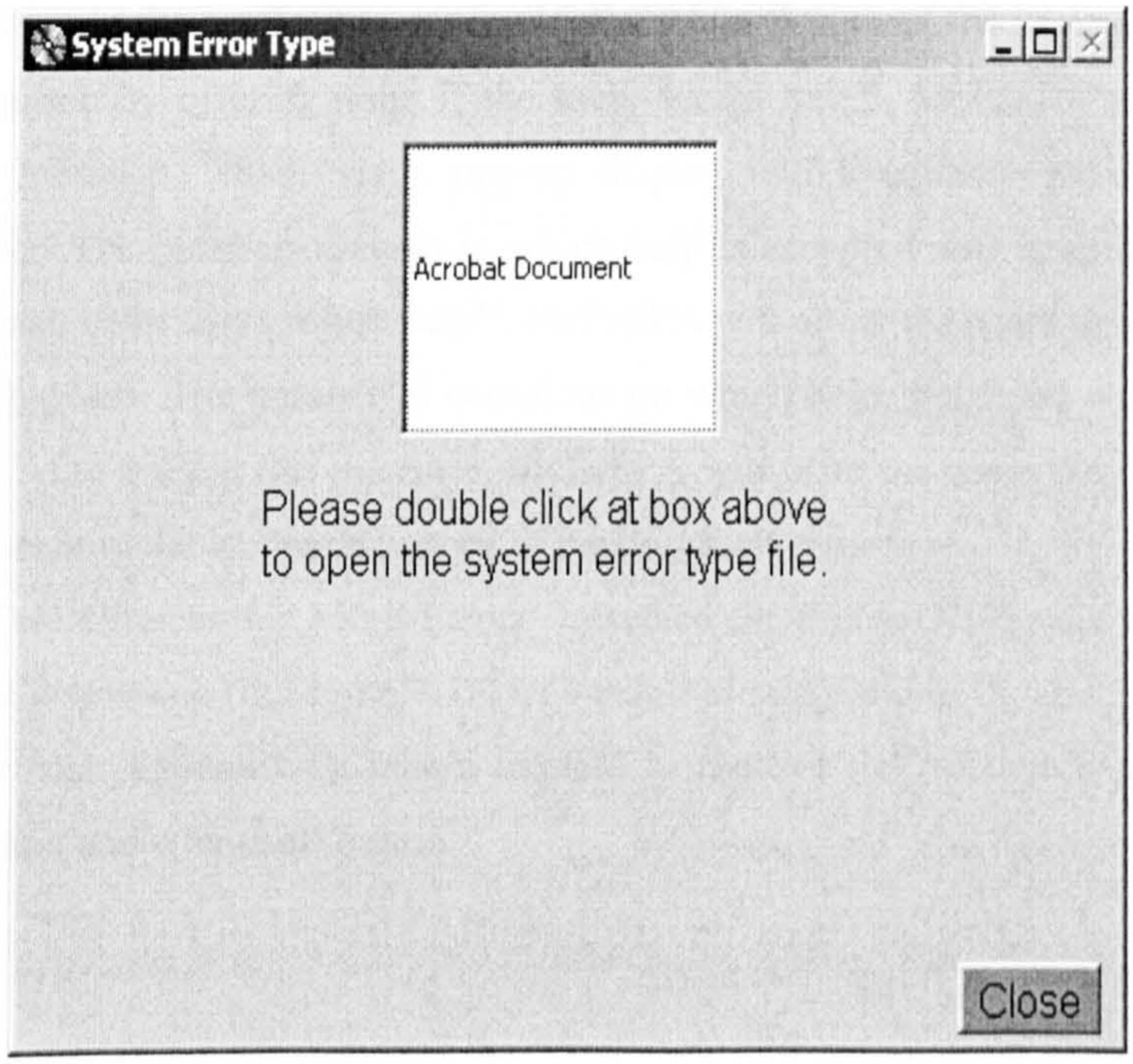


Figure 7.16 'System Error Types' interface

- The **'Choices for More Errors'** interface (see Figure 7.18) will be displayed when McFeSPA has found more than one error for the same error type

and depends on the rules for quality feedback (see Section 5.6.1, Chapter 5). This interface offers four options. If the 1st option is selected, McFeSPA will generate a brief feedback message similar to the error messages that are generated by McFeSPA after the student's solution file is analysed. If the 2nd option is selected, McFeSPA will generate a detailed feedback only once according to the 1st errors of the same error types found. The rest will be reported as "There are X errors like this" – when X is the number of the same error type. If the 3rd option is selected, McFeSPA will generate all detailed feedback according to the number of the same error type found. If the 4th option is selected, McFeSPA will not do anything. All feedback messages generated, according to the 1st, 2nd, and 3rd option, will be written in the temp file to help the users organize feedback before creating the feedback report. If the users select either the 2nd option or the 3rd option, the measurement of the users' skill meter in giving 'detailed feedback' will increase. If the users select the 2nd option, the measurement of the users' skill meter in giving 'specific/important feedback' will increase. In this module, if the users are in the scaffold-on mode, McFeSPA will support the users to provide the 2nd option by offering help. If the users accept help[17], McFeSPA will provide a help message, which via a pop-up display will eventually provide the right answer. The number of cases in which help is accepted will be counted here. In contrast, if the users refuse help[18], McFeSPA will allow the users to administer the next process. The number of occasions on which help is refused will be counted here. After ending this interface, McFeSPA will offer the users the option to take into account the student's history of making each error type.

The 'Choices for More Errors' interface (in Figure 7.17) was redesigned to create a new one (in Figure 7.18) as a result of suggestions 18, and 32 in usability study (see Appendix I), which implied to remove the 'student's profile' menu item and add a 'cancel' button.

[17] 'Accept help' is the action that the user agrees help from McFeSPA after making an error.
[18] 'Refuse help' is the action that the user denies help from McFeSPA after making an error.

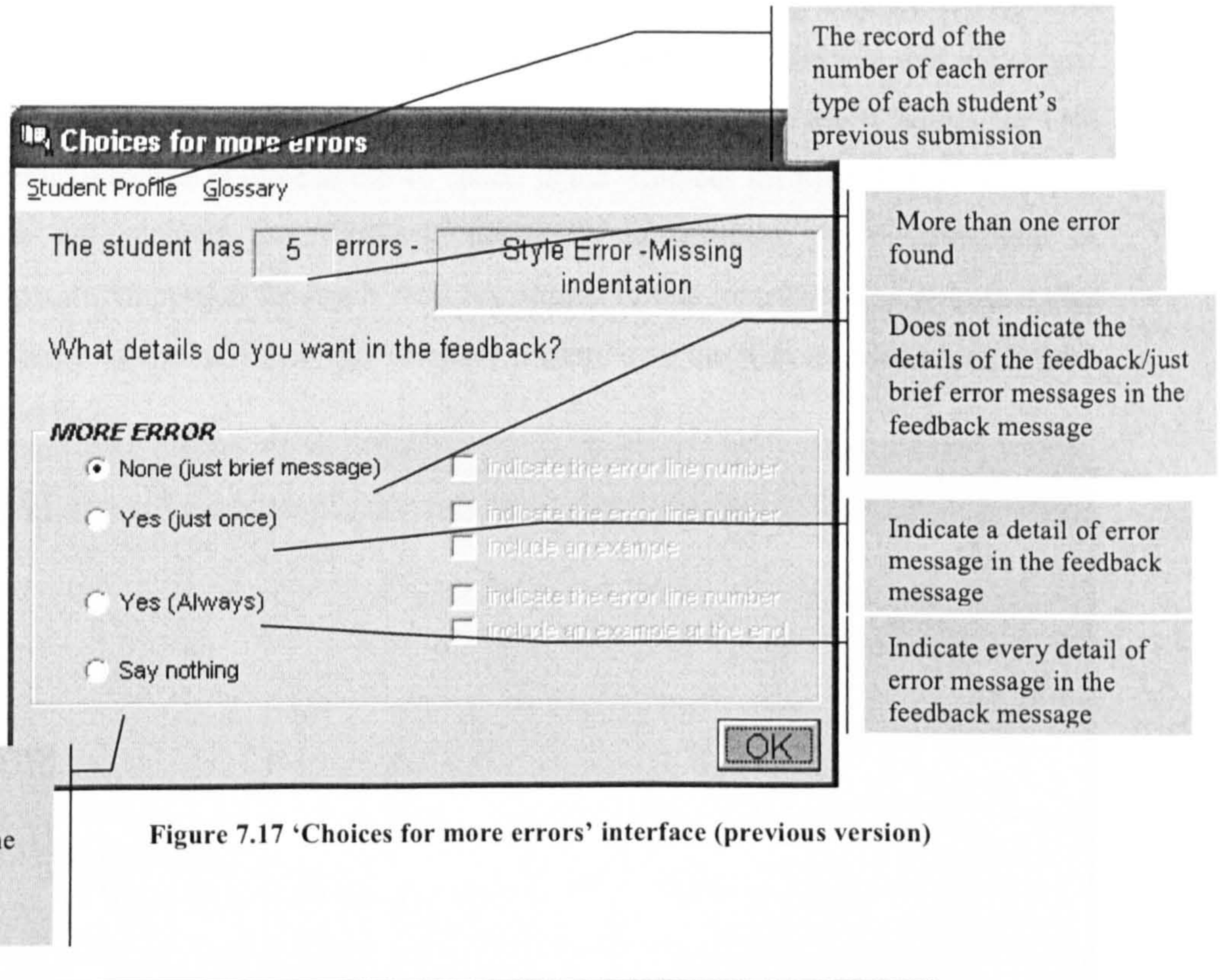


Figure 7.17 'Choices for more errors' interface (previous version)

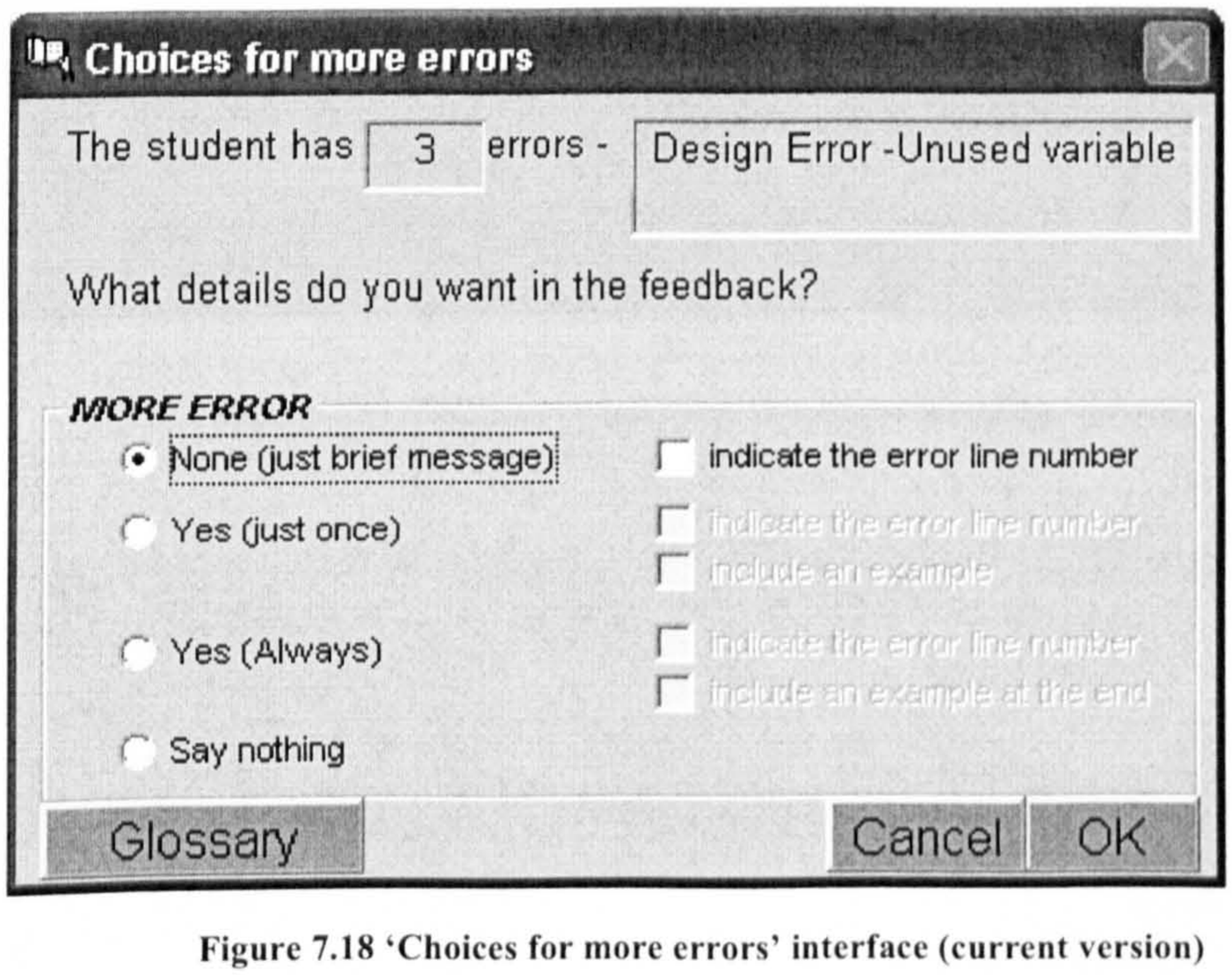


Figure 7.18 'Choices for more errors' interface (current version)

- The **'Choices for One Error'** interface (see Figure 7.20) is run similar to the 'Choices for More Errors' interface and depends on the rules for quality feedback (see Section 5.6.1, Chapter 5). There are a few differences from the

'Choices for More Errors' interface. The 3rd option (giving detailed feedback to every same error found) in the 'Choices for More Errors' interface will not appear in the 'Choices for One Error' interface. The 3rd option in the 'Choices for One Error' interface is same as the 4th option in the 'Choices for More Errors' interface (do not generate any feedback messages). In addition, the measurement of 'specific/important feedback' will not appear in this interface. The help message offered by McFeSPA in this interface is similar to the 'Choices for More Errors' interface.

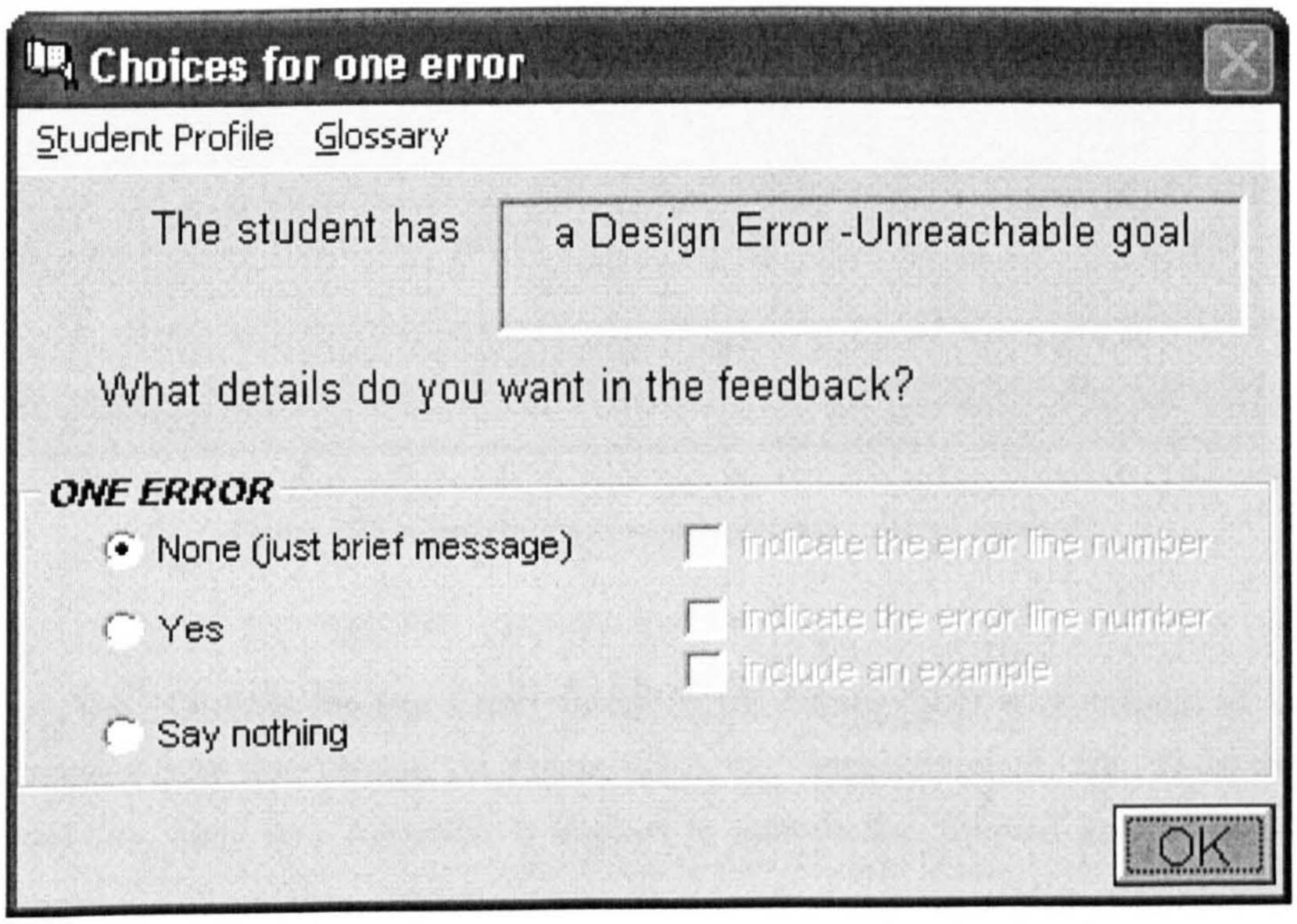


Figure 7.19 'Choices for one error' interface (previous version)

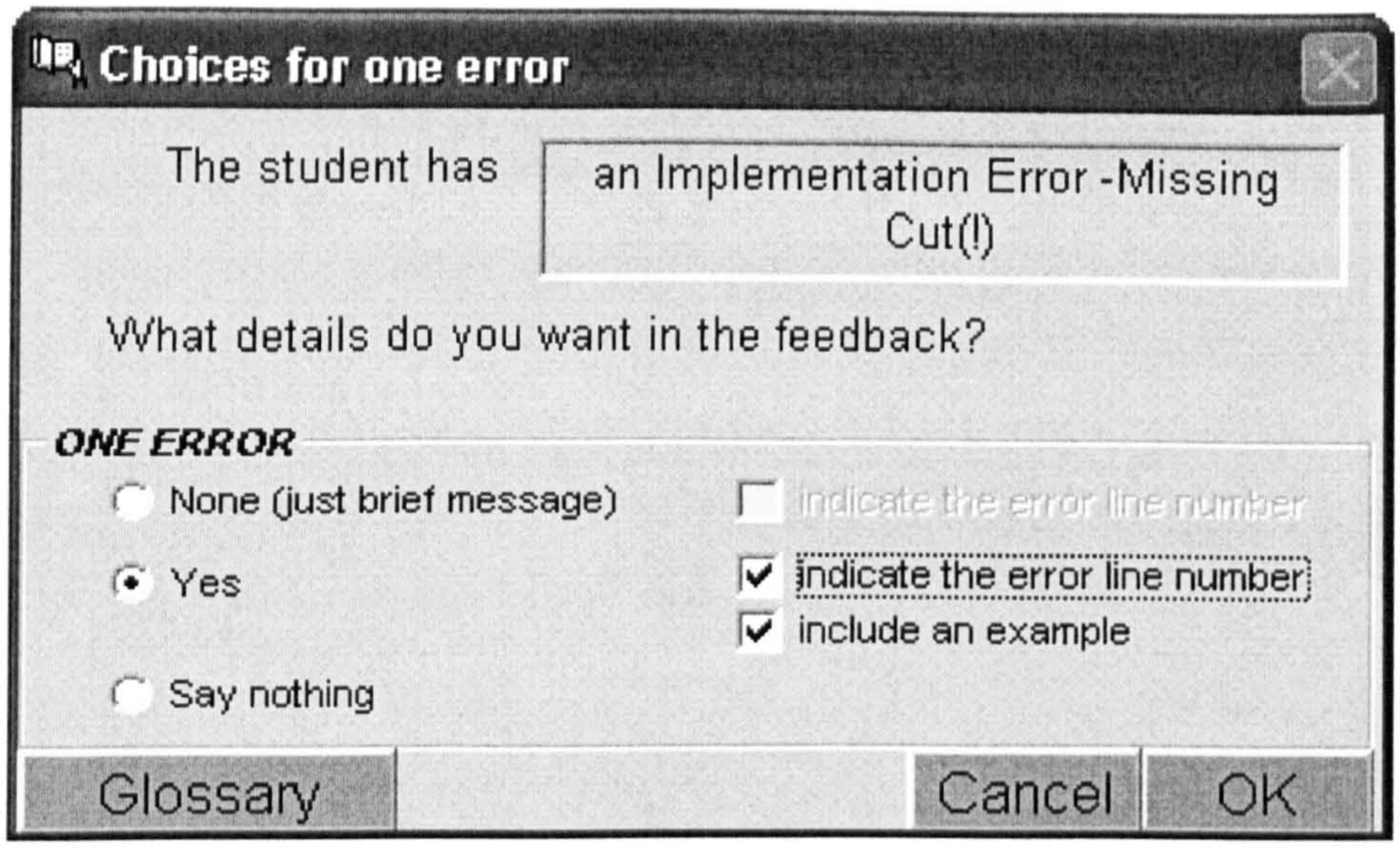


Figure 7.20 'Choices for one error' interface (current version)

The 'Choices for one Error' interface (in Figure 7.19) was redesigned to create a new one version (in Figure 7.21) as suggestions 18, and 32 in the usability study (see Appendix I) implied to remove the 'Student Profile' menu item and add a 'cancel' button.

- The **'Taking into account history of student's errors'** interface (see Figure 7.22) will be displayed when the users need to add any extra sentences after the error messages and depends on the rules for quality feedback (see Section 5.6.1, Chapter 5). In this interface, if the users select the extra message corresponding to the history of student's errors then the measurement of the users' skill meter in giving 'feedback loop' and 'individual feedback' will increase. The users can view the previous student's errors from the student's profile to compare it with the current error to consider which option should be selected. Help messages are also offered by McFeSPA in this interface. Despite the help messages in this interface being different from the 'Choices for More Errors' interface and the 'Choices for One Error' interface, the process of giving help, accepting help and

refusing help are quite similar.

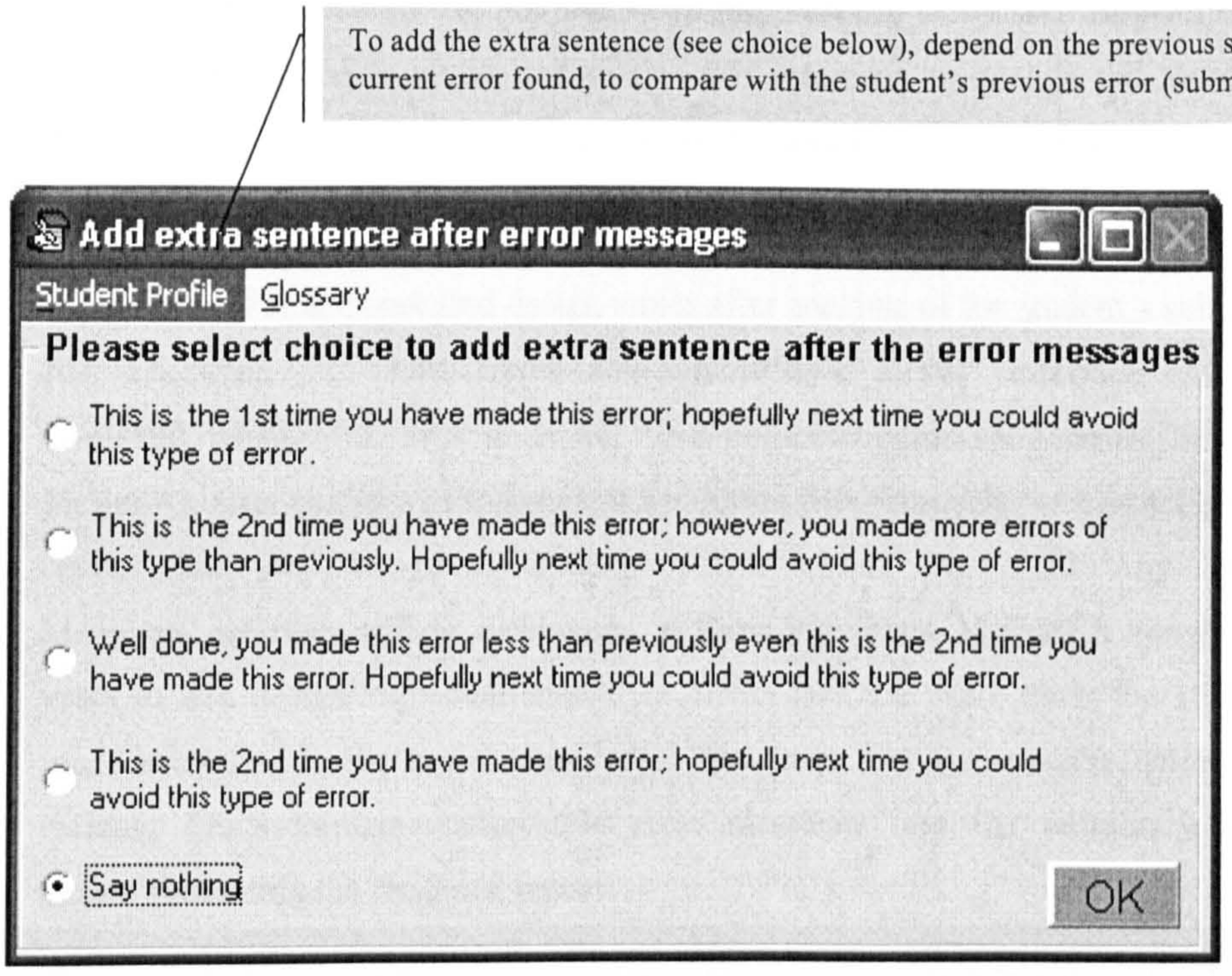


Figure 7.21 'Add extra sentence after error message' interface (previous version)

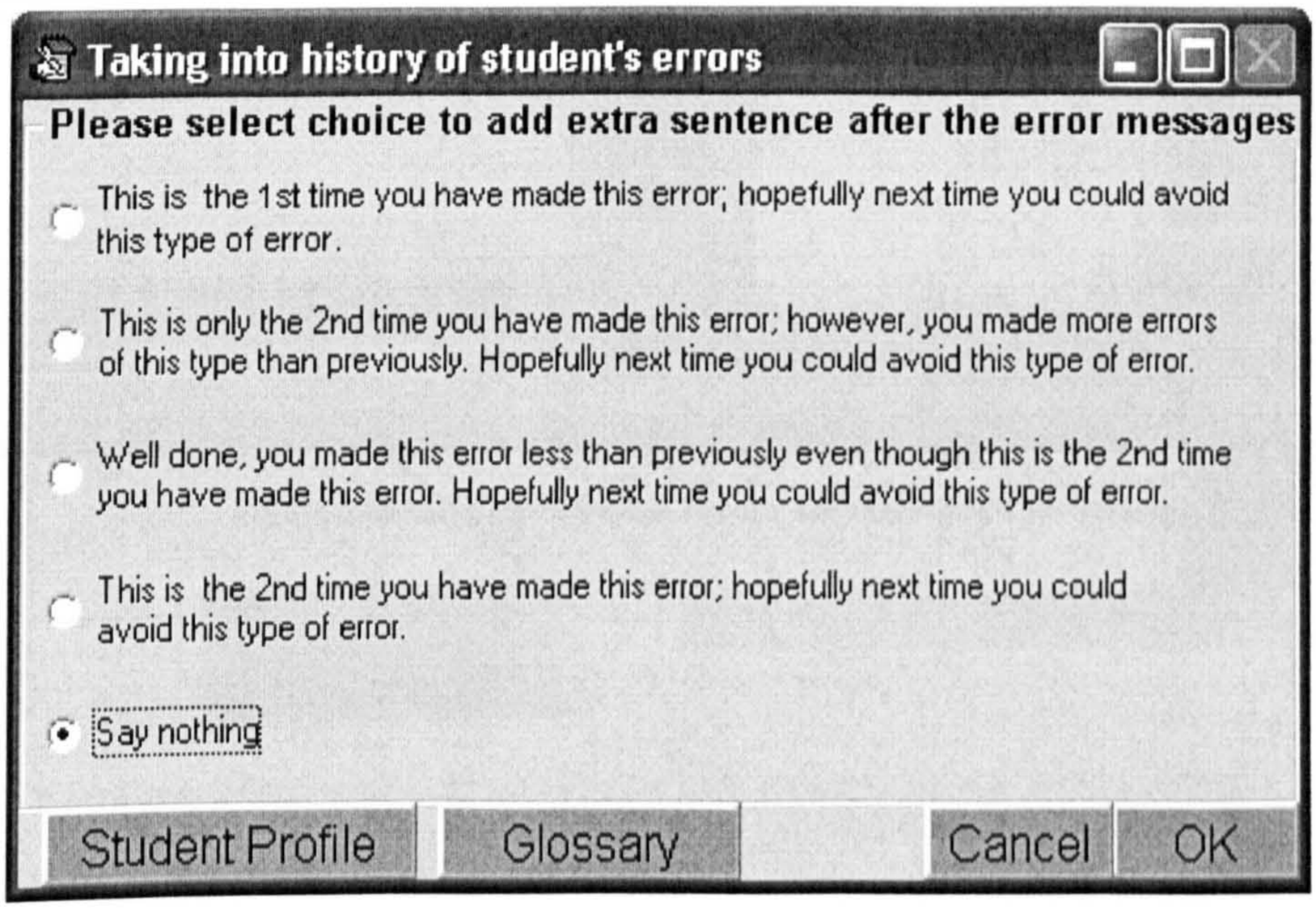


Figure 7.22 'Taking into account history of student's errors (current version)

The 'Add extra sentence after error message' interface (in Figure 7.21) was

updated to 'Taking into account history of student's errors' interface (in Figure 7.22) due to suggestions 20, 30, and 32 in the usability study (see Appendix I). These were implied to update the title of this interface, to change the 'Student Profile' menu item to the 'student profile' button and to add the 'cancel' button.

- The **'Add Extra Design Error'** interface (see Figure 7.24) will be displayed when McFeSPA does not find design errors after analysis of the student's solution file. Likewise, the **'Add Extra Implement/Style Error'** interface will be displayed when McFeSPA does not find implementation/style errors beyond McFeSPA after analysis of the student's solution file. Similarly, when McFeSPA cannot find any design/implementation/style errors, the 'Add Any Error Messages' interface will be displayed. In these interfaces, McFeSPA allows the users to add design/implementation/style errors that the users think the student made according to the users' knowledge. The users can also update/ delete the existing design/implementation/style error messages into the temporary area before generating the feedback report.

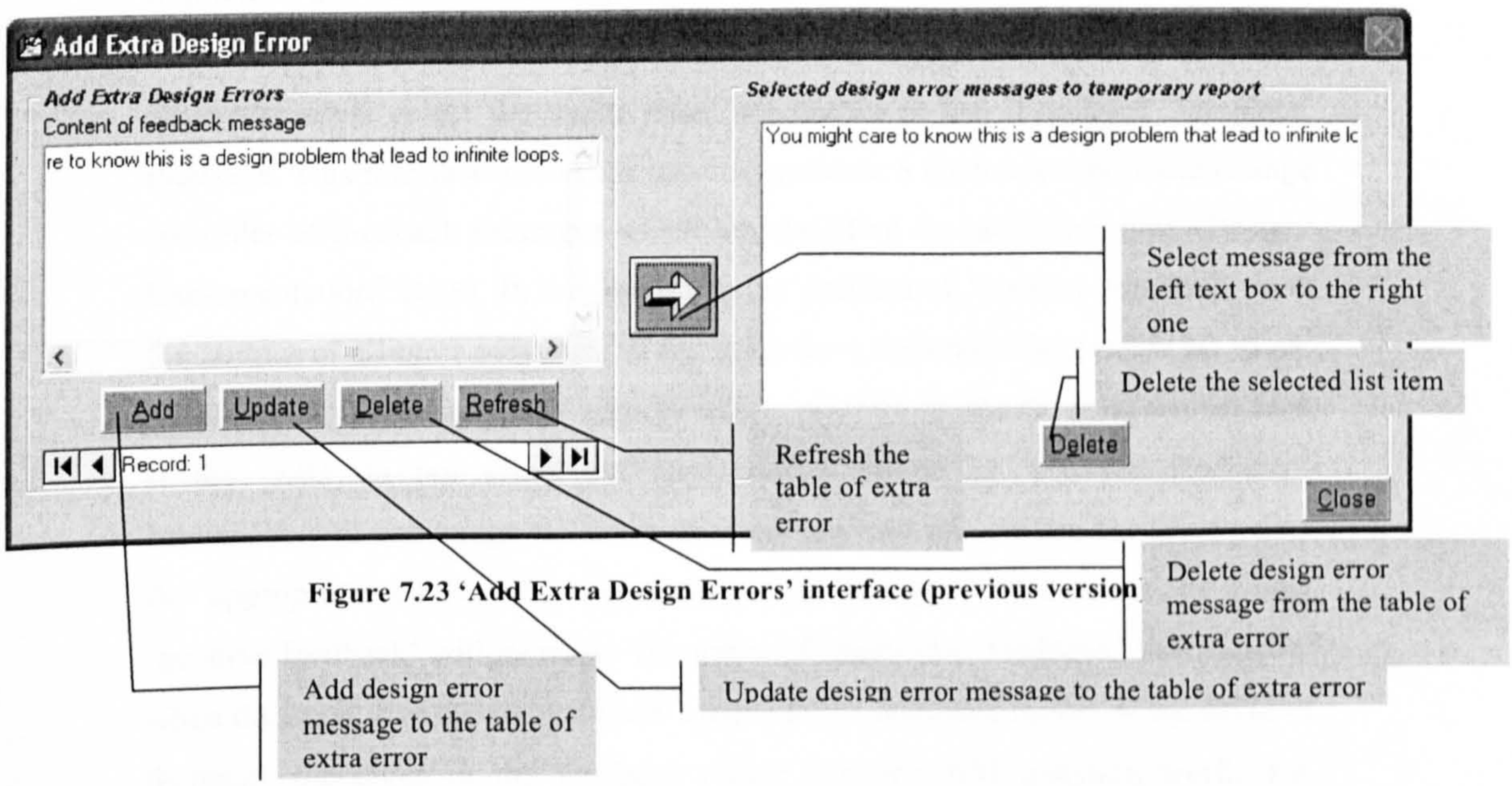


Figure 7.23 'Add Extra Design Errors' interface (previous version)

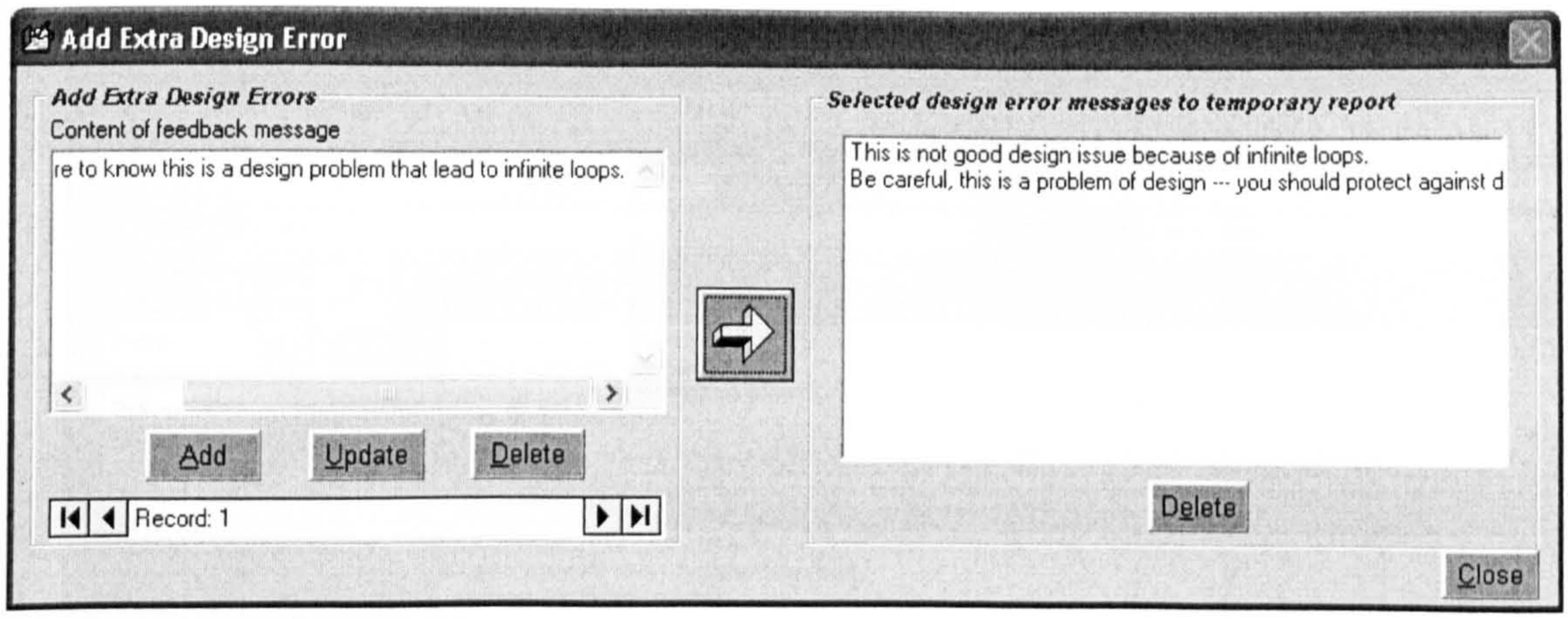


Figure 7.24 'Add Extra Design Errors' interface (current version)

The 'Refresh' button in the 'Add Extra Design Error' interface was removed because of the suggestion 21 in the usability study (see Appendix I) which implied its use was not necessary (Figure 7.23 & 7.24).

- The **'Create Feedback Report'** interface (see Figure 7.25) will be displayed when the users select the upper most left button in the 'Feedback Template' interface. This interface allows the users to generate a feedback report and change the order of feedback messages which are classified by each error type (Design/ Implementation/ Style). In this interface, the position of 'positive feedback' is at the bottom of all error messages. If the users have the scaffolding mode set to on, McFeSPA will encourage the users to select 'positive feedback' from the list box. If the users do not select the appropriated choice of 'positive feedback', McFeSPA will encourage the users to select the best one. When the users select the appropriate choice, the measurement of the users' skill meter in giving 'positive feedback' will increase. The meter of 'individual feedback' will increase when the users give the right student's name in the feedback report. If the detail of feedback messages in the feedback report contains WH question words e.g. 'why'; 'when'; 'where'; what; 'how', the measurement of the users' skill meter will increase. This interface will ask the users to double check the details of the feedback report before generating the final feedback report.

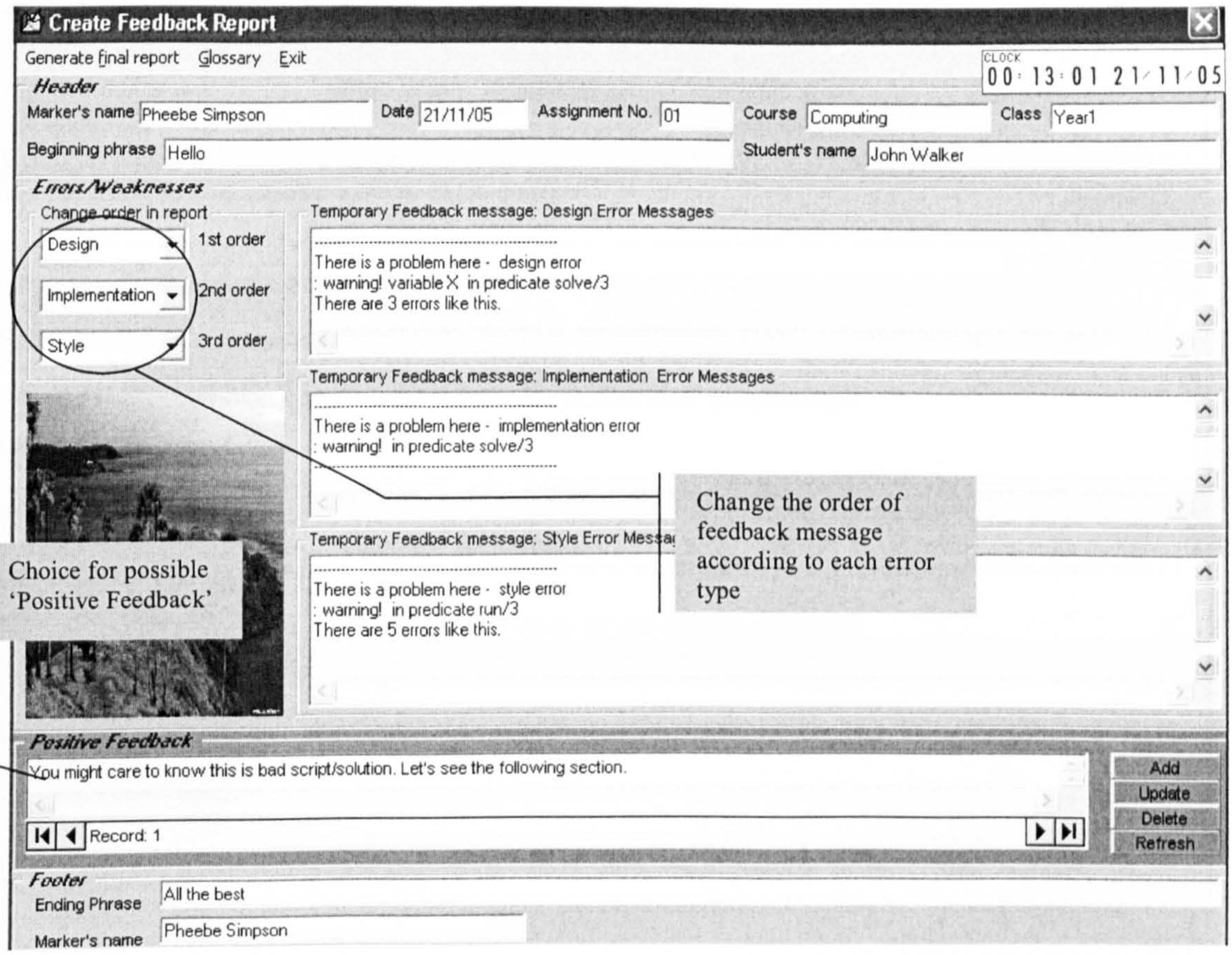


Figure 7.25 'Create Feedback Report' interface (Design error/Implementation error/Style error/ 'positive feedback')

The 'Create Feedback Report' interface (in Figure 7.25) was updated to a newer version. The 'Refresh' button was removed because of suggestion 21 in the usability study (see Appendix I) which implied its use was not necessary.

When the users select the upper middle button in the 'Feedback Template' interface, the **'Create Feedback Report'** interface (see Figure 7.26) will be displayed. This interface allows the users to generate a feedback report similar to the interface in Figure 7.25 but in this interface, the position of 'positive feedback' is on the top of all error messages. The other objects in this interface are similar to the interface in Figure 7.25.

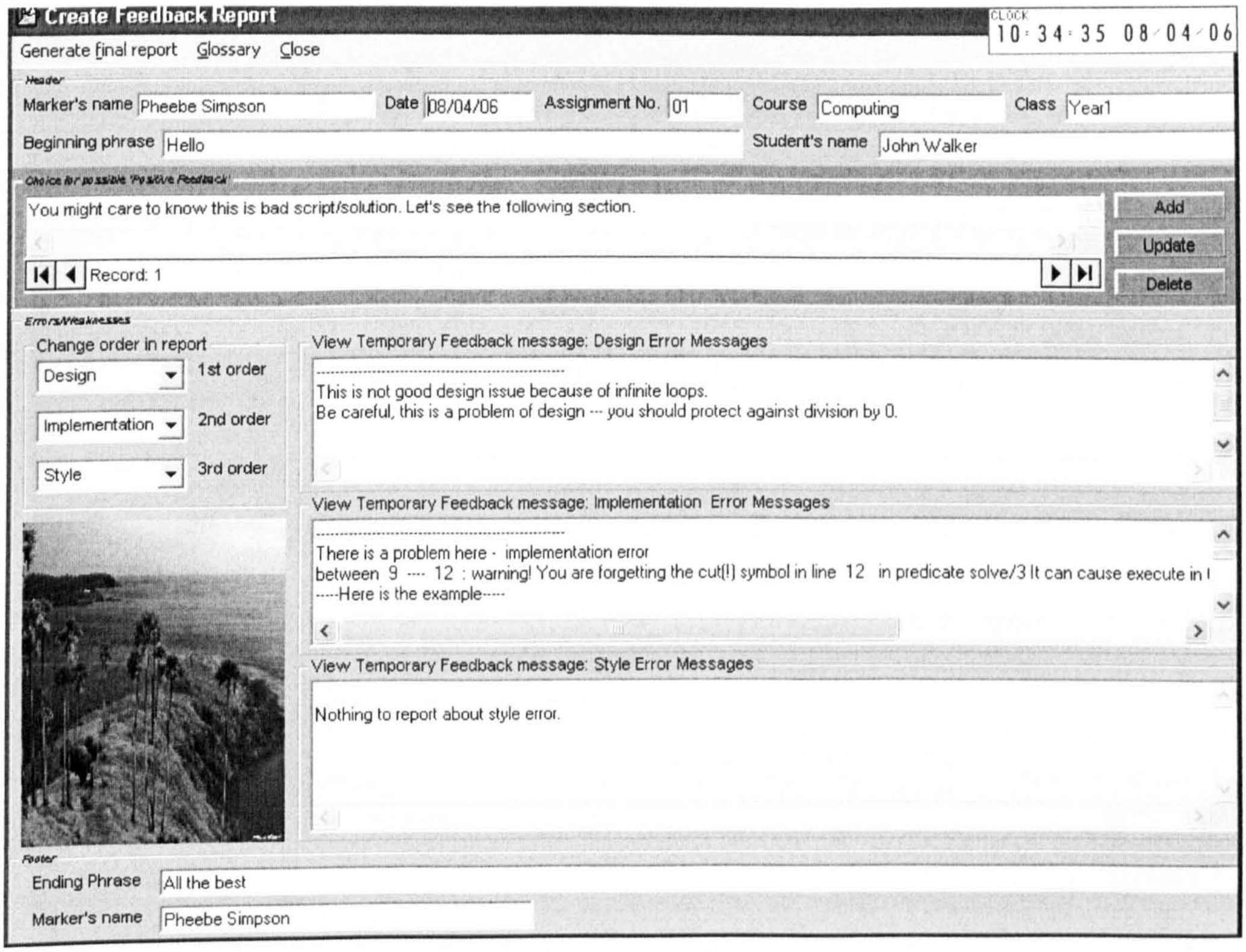


Figure 7.26 'Create Feedback Report' interface ('Positive feedback'/ Design error/ Implementation error/Style error)

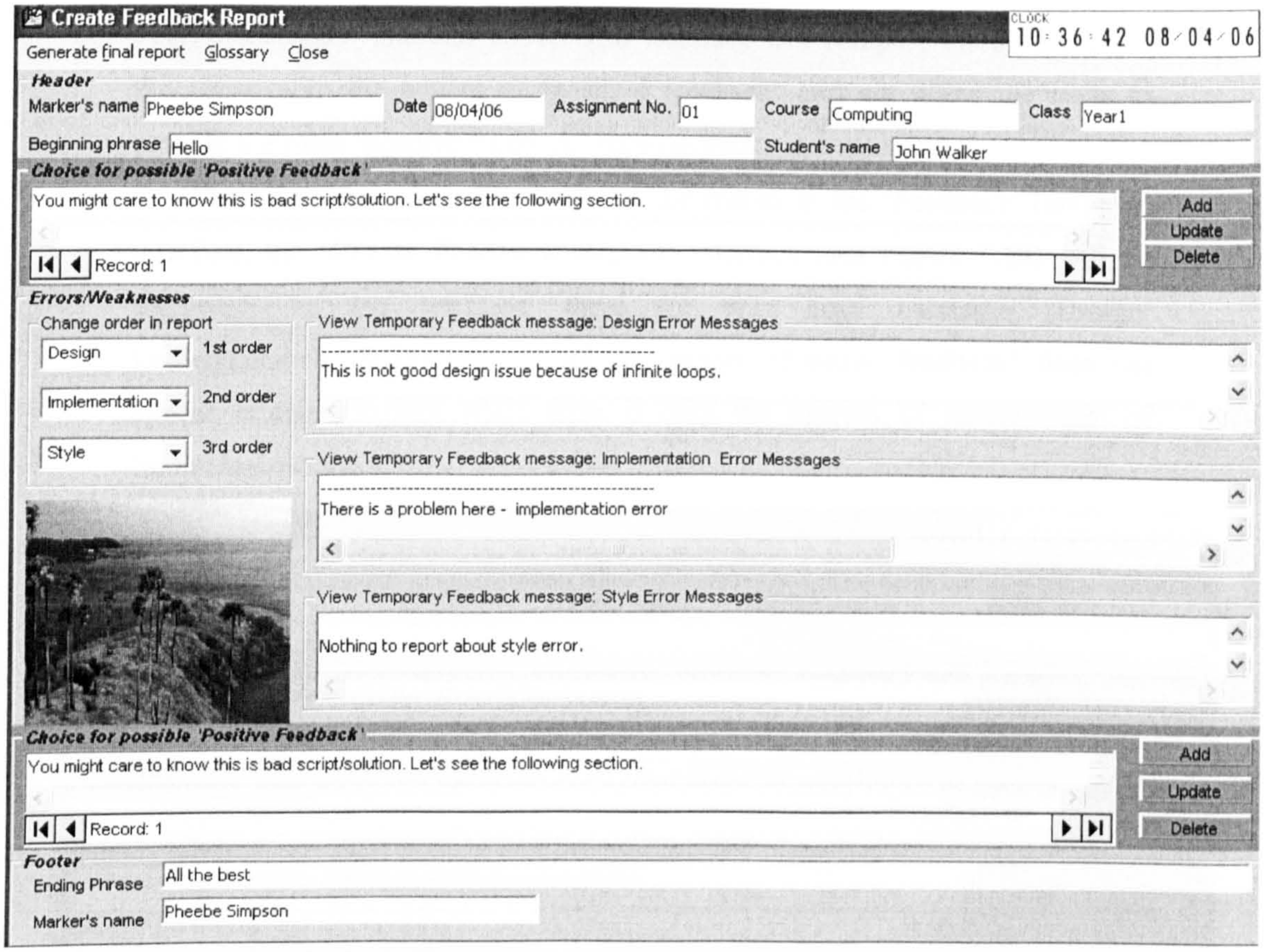


Figure 7.27 'Create Feedback Report' interface ('Positive feedback'/ Design error//Implementation error/Style error/ 'Positive feedback')

When the users select the upper most right buttons in the 'Feedback Template' interface, the **'Create Feedback Report'** interface (see Figure 7.27) will be displayed. In this interface, the position of 'positive feedback' is on both the top and the bottom of all error messages. The other objects in this interface are similar to ones in the 'Create Feedback Report' interface above (in Figure 7.25, Figure 7.26). This interface will compare the difference of the top and the bottom of 'positive feedback' and will not allow the users to provide the same 'positive feedback'.

When the users select the lower most left button in the 'Feedback Template' interface, the **'Create Feedback Report'** interface (see Figure 7.28) will be displayed. In this interface, each error message (Design/ Implementation/ Style) is

between two positions of 'positive feedback'. The right answer of the bottom position of 'positive feedback' in this interface is different from the other position of 'positive feedback'. The object in this interface is similar to the 'Create Feedback Report' interface above. This interface will compare the difference of the above with the bottom of 'positive feedback' and not allow the users to provide the same 'positive feedback' between each error message.

When the users select the lower middle button in the 'Feedback Template' interface, the **'Create Feedback Report'** interface (see Figure 7.29) will be displayed. In this interface, there are only error messages (Design/ Implementation/ Style) in the feedback report. 'Positive feedback' does not appear in this interface while there is only the process of measurement of 'individual feedback'.

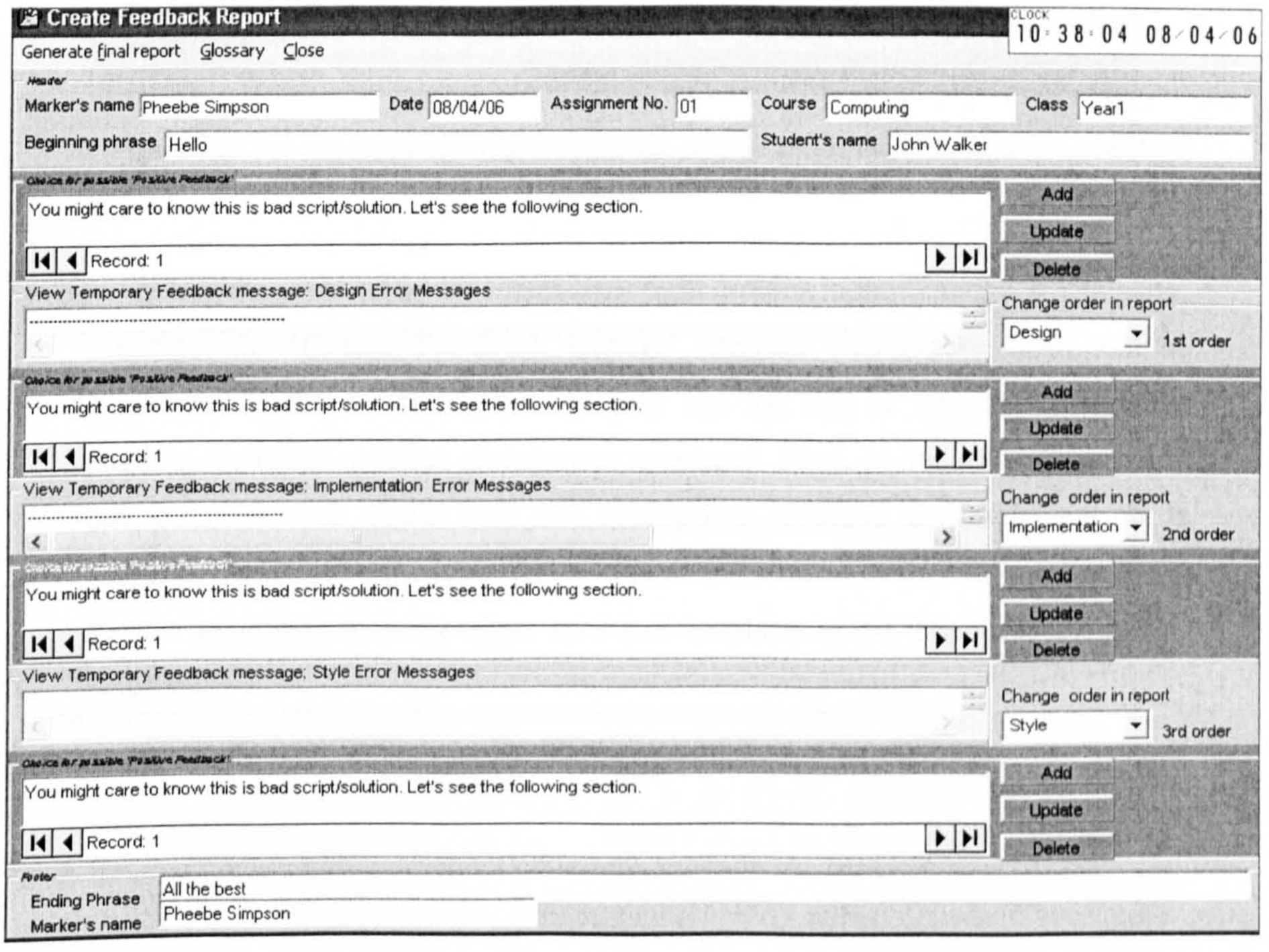


Figure 7.28 'Create Feedback Report' interface ('Positive feedback'/ Design error/ 'Positive feedback'/Implementation error/ 'Positive feedback'/Style error/'Positive feedback')

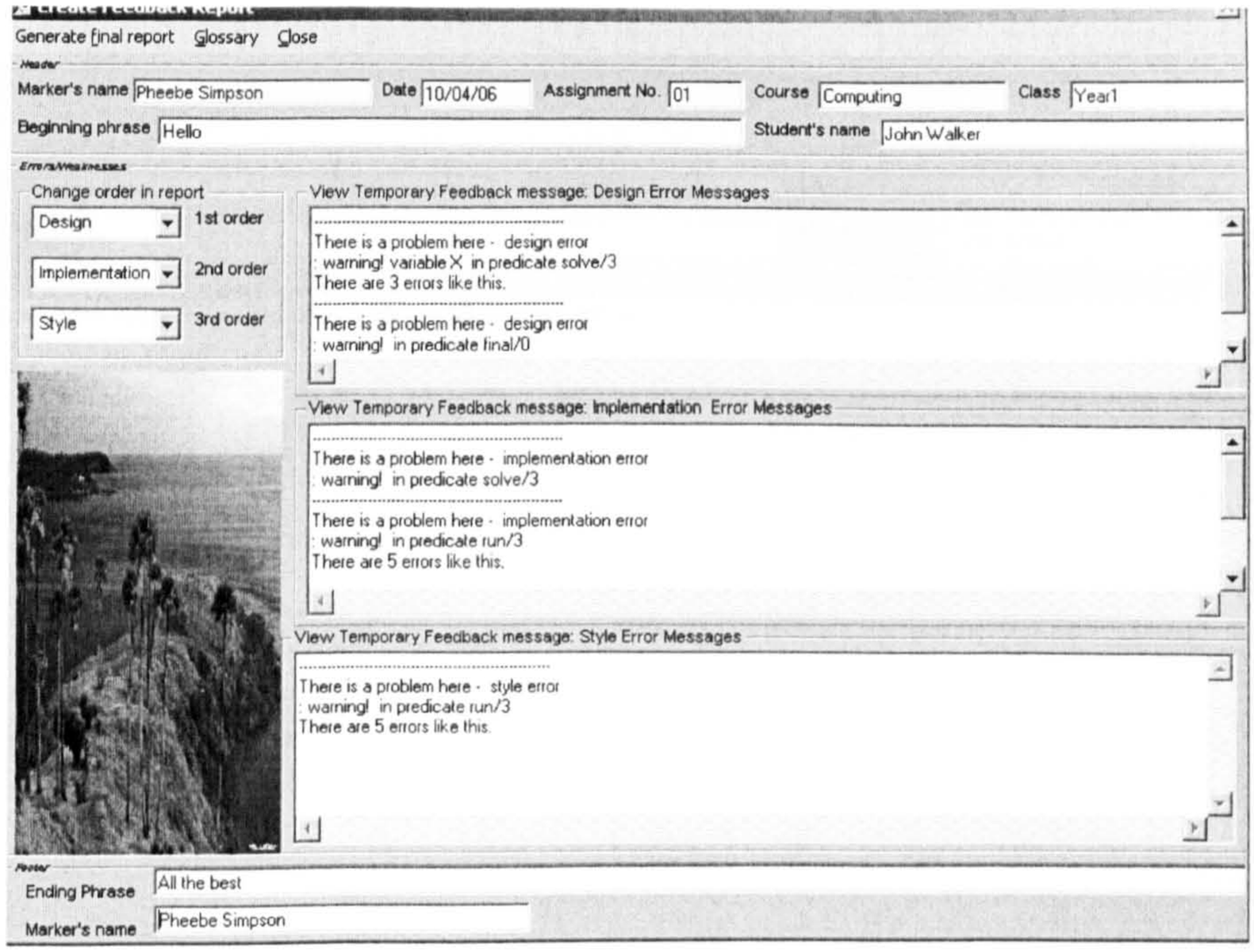


Figure 7.29 'Create Feedback Report' interface (Design error/ Implementation error/ Style error)

When the users select the lower most right buttons in the 'Feedback Template' interface, the **'Create Feedback Report'** interface (see Figure 7.30) will be displayed. In this interface, there is only one list box of 'choice for possible positive feedback' without giving error messages. Thus, in the scaffold-on mode this interface will check only the appropriate 'positive feedback' e.g. It is 'negative feedback' messages. Other objects are similar to the 'Create Feedback Report' generated from the upper most left templates in the 'Feedback Template' interface.

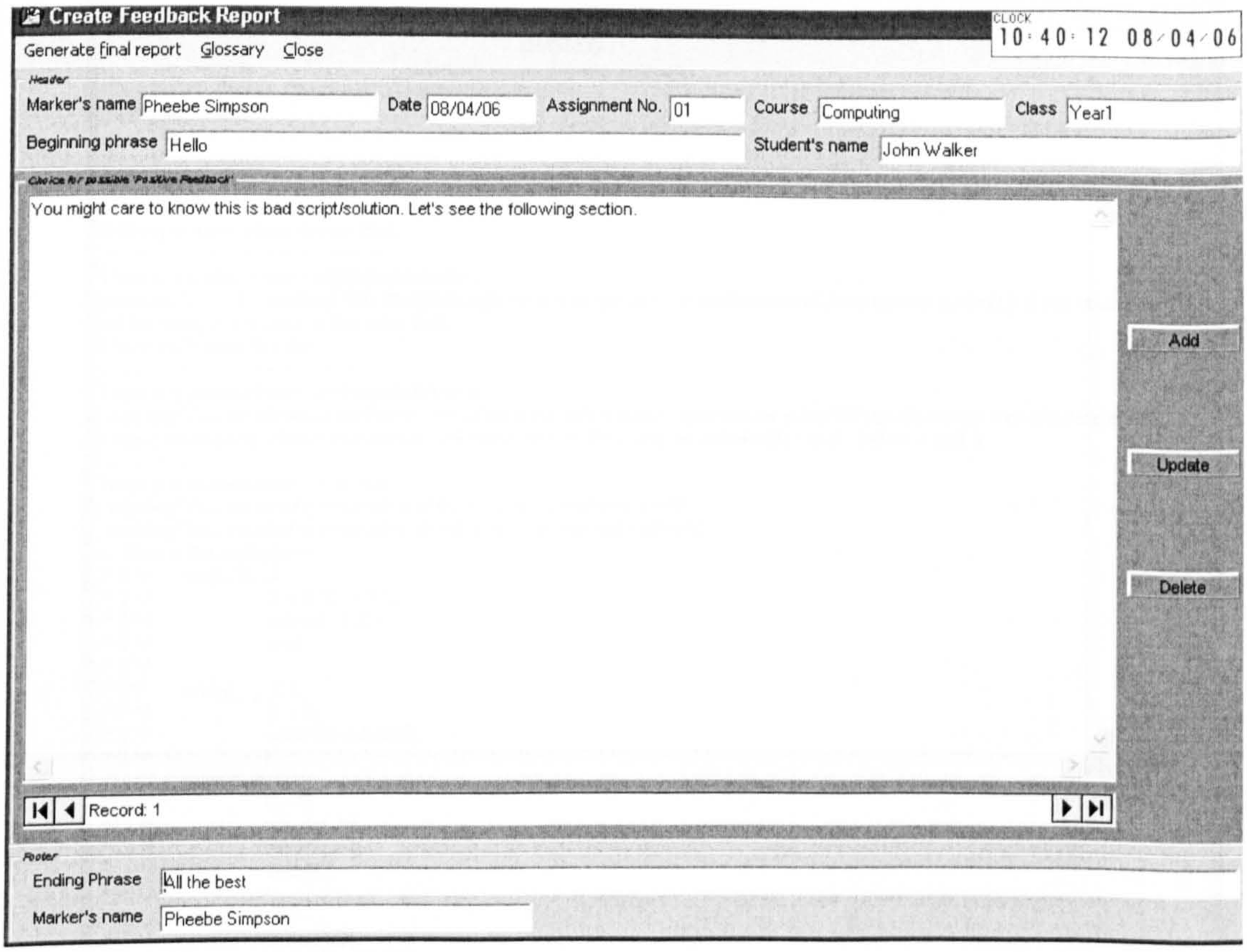


Figure 7.30 'Create Feedback Report' interface ('positive feedback')

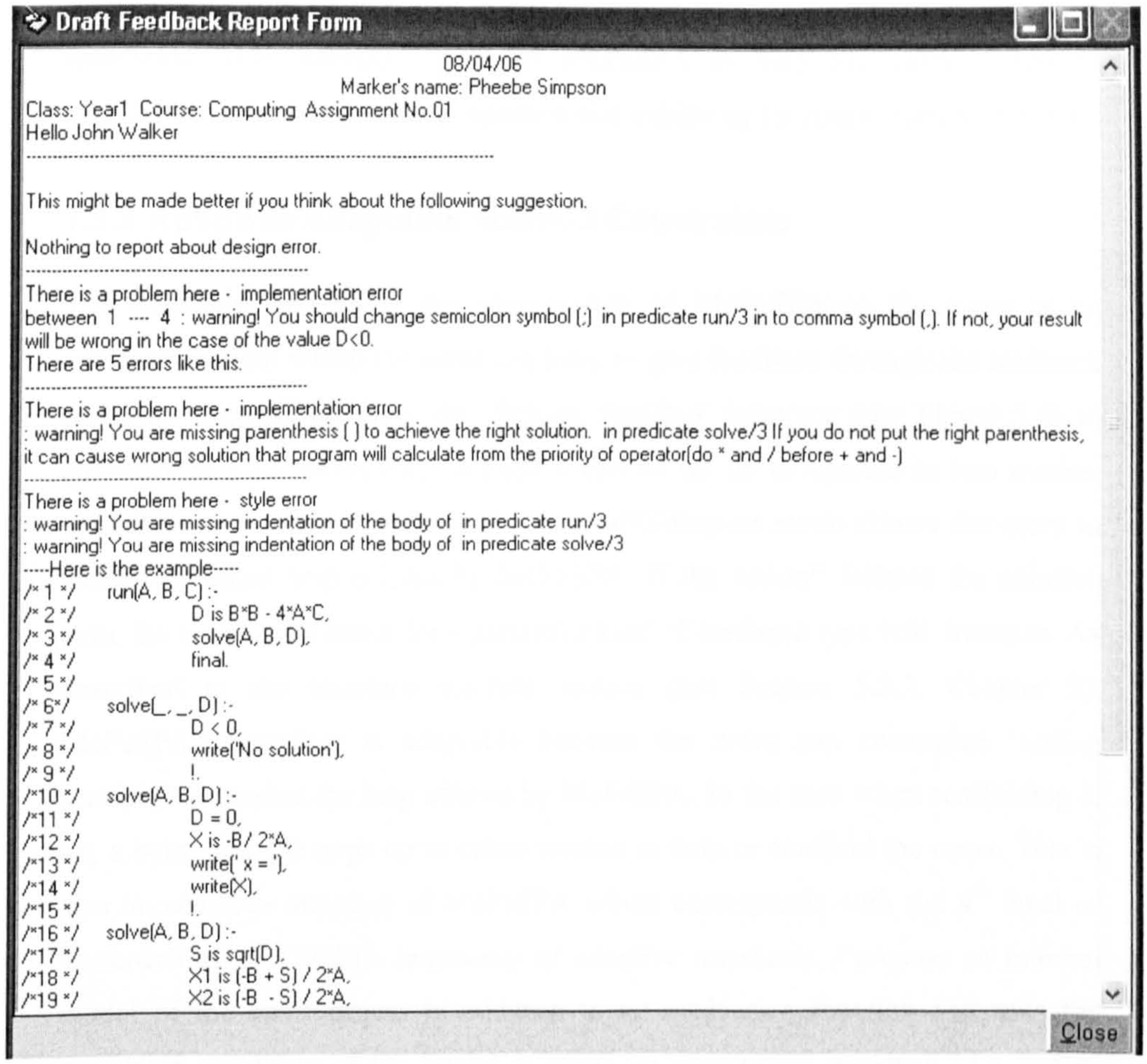


Figure 7.31 'Draft Feedback Report' interface

- The **'Draft Feedback Report'** interface (see Figure 7.31) assists the users to generate the final feedback report before sending the report to each student.

In scaffold-on mode, for every interface, the users are offered help support by McFeSPA. When users refuse help from McFeSPA, the refusing help counter will increase. While if the users accept help from McFeSPA, the counter of accepting help will increase.

7.3 Constraints

The interface, described in Section 7.2, was designed to embody McFeSPA in helping users learn to give quality feedback while marking student programming

assignments. This embodiment was achieved by means of identifying constrains associated with individual parts of McFeSPA to help the users customize McFeSPA and create feedback reports while achieving feedback from McFeSPA.

7.3.1 Adaptive/Adaptable Scaffold Constraints

The metaphor underlying the presentation of McFeSPA to the users is an environment into which the users can learn to give feedback through the feedback given by McFeSPA. Using the 'Setting Scaffold' interface (see Figure 7.8) in customization of McFeSPA, McFeSPA can be set up to operate in two modes: scaffolding-on and scaffolding-off. The scaffolding-on mode allows the users to accept or refuse help offered by McFeSPA. If the action[19] follows the solution path, the user's skill meter for a particular kind of feedback type will increase. As described in the interface scaffold section (see Section 5.3.2, Chapter 5), McFeSPA's interface is adaptable because the users can customize 'setting scaffold' to receive the help offered by McFeSPA. In the case when scaffolding is on, a help message pops up to either remind or help or scaffold the users. This is also an adaptive interface of McFeSPA which corresponds with the 4th level of Totterdell et al. (1987)'s taxonomy of adaptive interfaces -"possess an internal model of the environment in addition to an evaluation function and uses the predictive capability of the model to select an appropriate response." i.e. before providing any hint/help to the users, McFeSPA will check whether scaffold-on mode has been set. With regard to McFeSPA the fading of scaffolds halts immediately by the user's decision to choose adaptable fading (more details see Section 5.3.3 in Chapter 5).

Scaffold in McFeSPA is one way of providing adaptive help. It does not keep a sequence of nodes visited (previous TA's action). It records only the TA's customisation and the last level of help for each hint. These hints are recorded when the TA exits the system. If the TA logs in again, the system will be begun at the last level of help used. However, if the TA's record indicates that they are at the final level of help, the system will inform the previous level of the final level

[19] Action is a particular performance that the user does while using McFeSPA for example 'click menu', 'click menu item', 'click button', 'click item list', 'click checkbox', 'click option', 'click textbox', 'double click item list', 'type a message', 'scroll the list box'

and offer help to the TA. It is up to the TA whether to request or refuse help from the system.

7.3.2 Functional Scaffold Constraints

A functional scaffold in McFeSPA means that most of the screen objects will provide the users with information when the mouse pointer moves over it e.g. when moving over the 'Analysed Student' solution' pane in the main interface of McFeSPA, the text of "brief lists of analysed script --click each list item to create feedback message" will be shown. According to Randoll and Kali (2002) classification of types of scaffold, the tool text tip in McFeSPA is similar to functional scaffold that explain to the users what each object means.

7.3.3 Menu Constraints

The idea of implementing a menu system in McFeSPA derived from the principles of menu design in HCI. The organised items on a menu in McFeSPA are based on functional relationships between the menu items. McFeSPA's menu is not complicated so we designed it as a functional menu which includes shortcut keys.

7.3.4 Glossary Constraints

The idea of building the 'Glossary' interface in McFeSPA derived from PACT geometer (Aleven & Koedinger, 2000). While using/learning with McFeSPA, the user can select a particular quality feedback word from the list box of quality feedback in the 'Glossary' interface to see the explanation, the meaning in the skill meter, and an example.

7.3.5 Skill Meter Constraints

With regard to the main approach of McFeSPA's system design, McFeSPA allows the TAs to view their own progress in providing quality feedback to the students– this is called a metacognitive scaffold. We present this function as a skill meter which can be found in many systems (for instance, in Cognitive tutors e.g. ACT Programming Tutor: APT (Corbett & Trask, 2000); PACT Geometry

Tutor (Aleven & Koedinger, 2001)). In the current version, we offer 12 scripts of student's solution, 6 types of quality feedback and 20 scales[20] of the quality feedback skill meter in order to be sufficient in constructing a prototype for helping users to learn giving feedback. Currently, we provide 20 scales for each feedback type. If the meter of one becomes full and the number of this feedback type is more than this, these levels will stay full. The algorithm for increasing the skill meter can be seen in Appendix C. The approach for the representation of quality feedback given in the skill meter of McFeSPA can be derived from the following.

1) 'Important/Specific Feedback': The measurement of 'important/specific feedback' will be checked when there is more than one occurrence of the same error type *, found. If the users select **'Yes (just once)'** option, the skill's level of 'important/specific feedback' will be increased.

* (From the choices offered in the 'Choices for more errors' interface)

2) 'Positive Feedback': The measurement of 'positive feedback' will increase when users select the appropriate feedback template (feedback sandwiches –start and end with 'positive feedback'. When the users enter the 'Create feedback report' interface and select the appropriate 'positive feedback' from the list box then the measurement of 'positive feedback' will increase. This includes providing different header/footers of 'positive feedback' - feedback sandwiches (in the 'Create feedback report' interface).

3) 'Detailed/Elaborative Feedback': The measurement of 'detailed/elaborative feedback' will increase when the users select the 'Yes' option either there is a number of the same error type found more than once in the 'Choices for more errors' interface (Yes- just once, or Yes-always) or only one error type found –in the 'Choices for one error' interface. However, if the scaffold mode is on then McFeSPA will help the users to provide the appropriate one. That depends on more than one or only one of same error type found then McFeSPA will guide the users to provide detailed feedback only once.

[20] a scale with 20 points

4) 'Feedback Loop': When the users provide feedback to each student and it is associated with the history of the student's errors, e.g. in the case of the error type found being the same as the one in the student's history - *"This is the 2nd, ... same error found; hopefully, next time you could improve your script to avoid this type of error"*. The measurement of 'feedback loop' will increase and also the measurement of 'individual feedback' will increase because 'feedback loop' is associated with 'individual feedback'.

5) 'Individual Feedback': McFeSPA provides students' name by default, according to the student's script being read in the feedback report at the time. Thus, the users can always provide **'individual feedback'** to the students. McFeSPA will check the student's name again to check if there is a difference in the names between the name in temporary feedback report (in the 'Create feedback report' interface) and the student's script (in the main interface), for both forename and last name or either of them, which is not same as student's name that is retrieved from student's registration. If so, the measurement of 'individual feedback' will not increase. If the measurement of 'feedback loop' increases, the measurement of 'individual feedback' will also increase (according to the 'feedback loop' rule).

6) 'Asking Question': When the users provide feedback containing a'wh' question e.g. –"who", "when", "where", "why", "what", or "how", then the measurement of the 'asking question' will increase according to the number of 'wh' questions in the feedback report.

Note: Before providing any hint/help to the user, McFeSPA will check whether **scaffold-on** mode is true or false.

7.3.6 Add any Errors Constraints

In the case of adding errors (all error types: design/implementation/style errors), McFeSPA allows users to add any error while the student's script is being analysed. In addition, the users can add more design/implementation/style errors before generating the feedback report either by clicking the 'Add Extra Design/Implementation/Style button in the main interface (see Figure 7.1) or by clicking the 'Manage more errors/weaknesses' menu item in the customize menu.

These error messages will be checked before being added in the temporary report to ensure they are not duplicated. If error messages are duplicated, the system will not allow the error messages to be added to the temporary report.

7.3.7 Feedback Template Constraints

The current version of McFeSPA offers 6 feedback templates that are satisfactory choices in generating the feedback report (see Figure 7.2). These are the left most upper template (1st choice) – error messages (design error/ implementation error/ style error) and 'positive feedback'; the middle upper template (2nd choice) – 'positive feedback'/ error messages (design error/implementation error/style error); the right most upper template (3rd choice) - 'positive feedback'/ error messages (design error/implementation error/style error)/ 'positive feedback'; the left most lower template (4th choice) – 'positive feedback'/ design error/ 'positive feedback'/ implementation error/ 'positive feedback'/style error/ 'positive feedback'; the middle lower template (5th choice) – error messages (design error/ implementation error/ style error); and the right most lower template (6th choice)– 'positive feedback'. The user can select any choices of feedback template; however, there will only be the one that is appropriate for the feedback pattern. That is the 3rd choice – known as feedback sandwiches.

7.3.8 Feedback Report Constraints

The idea of generating feedback reports in McFeSPA derived from Denton's (2001) work. Each feedback report will be generated from a chosen feedback template. If the template contains error messages, McFeSPA will allow the users to change the order of the error messages according to the error types found (design/ implementation/ style). If the template contains 'positive feedback', McFeSPA will allow the users to manage the 'positive feedback' by adding/updating/deleting the current 'positive feedback'. McFeSPA also allows the users to change the header and footer of the feedback report before generating the final feedback report. Furthermore, McFeSPA checks for duplicates of given 'positive feedback' before generating the final feedback report.

7.3.9 Hint Constraints

McFeSPA has a number of solution paths. If the users depart from McFeSPA's solution path, McFeSPA will offer help to the users to provide quality feedback. In the case that the users select an inappropriate answer, there will be a hint – scaffold the users- as can be seen from the context of hint in Section 5.7, Chapter 5.

7.4 Usability Evaluation

We have selected to use a variation of the 'discount usability' approach (Nielsen, 1993) which requires a small number of test users. Nielsen (1993) reported that heuristic evaluation is a very efficient usability engineering method. The evaluation of McFeSPA appears to fit the requirements for the heuristic evaluation method. Different kinds of expertise are required for the evaluators' performance since heuristic evaluations are supported to by representative users such as- end users, product developers, and usability specialists. Usability specialist can be used as the evaluators even if they have little or no expert knowledge of the application domain. Heuristic evaluation is performed by examining an interface and then providing an opinion about the positive and negative aspects of the interface (Nielsen, 1993). This evaluation is useful for the redesign process because McFeSPA is developed as a novel application (Kochakornjarupong et al., 2005). We also regard it as useful if evaluators can be participants in the redesign process. The "discount usability engineering" method used here is based on the use of the following three techniques: scenarios, a form of thinking aloud, and heuristic evaluation.

McFeSPA has already had several version updates (see Appendix B). According to the preliminary usability evaluation in November, 2004, the evaluation was run by handing in screen-shots of the system in paper form to 11 members of the LTRG (Learning Technology Research Group at the University of Northumbria) and a group discussion was conducted to obtain opinions and suggestions regarding possible changes to the interface. The suggestions derived from the LTRG members who indicated that the color scheme used in the system should be consistent and not various different colours in the same interface;

another outcome was that the size and the position of the 'Student's solution' pane should be changed and the language used in the interface should be improved.

According to such suggestions, the later version was updated (see Appendix C). A later usability evaluation was held in August 2005, with two evaluators with experience in developing educational software. The evaluations were run individually using co-operative evaluation and talk aloud methods. The main suggestions which led to the system updates consisted of a) adding the initial message before 'do you need help?' dialogue message; b) in the form of choice for one/more error, if the user selects "say nothing" (in scaffolding-off mode), the system would not enter into the "Add extra sentence after error messages" window; c) in the "add any error" interface, if the user does not need to add any error, the system should not provide a 'No' button but should provide 'Nothing to report' in the list of choices instead; d) move all error messages and feedback messages from the text box of Design/Implementation/Style pane in the main interface to each feedback report according to the user's selection of the feedback template; e) in the 'Feedback Template' interface, in the scaffold-on mode when the user selects option 'No' (no need for help) from 'Do you need help?' dialogue message the system should generate a feedback report according to each selection; add more messages when the user selects the choice 'No' in 'Do you need help' dialogue message then 'You don't need help so I will let you go through your needs' dialogue message will pop up; add view to the user's performance form according to the result of the user's skill meter (e.g. 'You produce a lot of positive feedback).

The last usability evaluation was run in late January and February 2006 with three evaluators. We proposed the following hypothesis, specific questions, methods, participants, materials, procedure, results, and analysis and discussion of the results.

Hypothesis: Providing appropriate computer support could help the users deliver feedback to the learner using functional scaffolding that helps the users find out how to interpret/use McFeSPA;

7.4.1 Specific Questions

The questioning of the evaluators was based on a number of usability/learnability issues. The following questions were selected for this purpose.

1) How do the users add/update/delete data in 'Favourite Wording'/ 'Favourite Content'/ 'Report Template' menu from the customise menu?

2) How do the users provide a feedback message if there is no student error reported by McFeSPA?

3) How do the users select a choice for one error (of the same error type) found?

4) How do the users select a choice for more errors (of the same error type) found?

5) How do the users provide extra sentences after selecting choice for error messages?

6) How do the users select a feedback template?

7) How do the users generate a feedback report from the 'create feedback report' window?

8) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Favourite wording' interface.

9) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'ManageData' interface.

10) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Main' interface.

11) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Choices for More Errors' interface.

12) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Choices for One Error' interface.

13) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Add extra sentence after error messages' interface.

14) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Add Extra Design Error' interface.

15) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Feedback Template' interface.

16) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Create Feedback Report' interface.

17) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Draft Feedback Report' interface.

18) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the 'Skill Meter' interface.

19) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using McFeSPA for the 1st time to mark the 1st script.

20) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using McFeSPA for the 2nd time to mark the 2nd script.

21) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using McFeSPA for the 3rd time to mark the 3rd script.

7.4.2 Methods

We used multiple methods, which can reduce inappropriate certainty (Robson, 2002), to collect data in the usability evaluation. Our methods consisted of a form of cognitive walk through[21] evaluation (Rieman et al., 1995), co-operative evaluation[22] (Marsh & Wright, 1999) with talking aloud, thinking aloud method (Ericsson & Simon, 1993), semi-structured interview, and questionnaire. The procedure of evaluation starts with a form of cognitive walk through evaluation. We provided a task description with several screen captured shots of the interface

[21] The cognitive walkthrough is a technique for evaluating the design of a users interface, with special attention to how well the interface supports "exploratory learning," i.e., first-time use without formal training.

[22] Co-operative evaluation: the Experimenter works with the expert. This method relates to the Cognitive walkthroughs when the experts do the task and anticipate problems. It's different from Cognitive walkthroughs because the experimenter anticipates the interface and provides knowledge and prompts to the experts to help them do it.

(see Appendix E), covering the correct action sequence in marking a programming prolog programming assignment through to generating a feedback report. The evaluators stepped through the sequence of actions required to accomplish each task, their actions on the screen were video-recorded as an avi file which provided not only an indication of some of the errors made by the users but also allowed for a quantitative measure of the user's performance to be obtained e.g. which mechanisms were used and when they were used. The audio tape recording was used to identify some of the possible intentional changes made by the evaluator, based on reactions such as surprise, agreement, or suggestions to improve McFeSPA while the actions were taken. During the use of McFeSPA, the evaluator can think aloud and ask any questions to the researcher while the researcher explains and gives the reason for the particular problem found. This process uses a form of cooperative evaluation and thinking aloud. Cognitive walk through is adapted by taking the questions asked by the experts and using them as the basic material to interpret the user's behaviour. Thus, we use a combination of three methods in usability evaluation: cognitive walkthrough, thinking aloud, and cooperative evaluation.

7.4.3 Participants

In order to perform a proper evaluation of McFeSPA interface, some expertise in using McFeSPA was required from the evaluators. We used three evaluators for usability testing as there is evidence from 36 published usability studies that the benefit to cost ratio for running a medium-size usability study is a maximum of three users (Nielsen, 1993). The selection of a certain usability evaluation approach also depends on the end evaluators' availability according to Nielsen (1993). Their backgrounds vary as described below:

- Evaluator 1 (EV_1) is a senior lecturer (Ph.D. in Intelligent Computer-based Education) from the Department of Computing Science, University of Glasgow. EV_1 is currently teaching Information Management, User Centered Software Design (MScIT), and Professional Software Development. EV_1's research interest involves usability of software engineering diagrams and presentation. EV_1 has several years background in usability testing including

searching usability testing process, and participation in testing both commercial and educational products.

- Evaluator 2 (EV_2) is a researcher (Ph.D. in Computer Science) from the SCRE centre. EV_2's research interests include Intelligent Tutoring Systems, and User Modeling. EV_2 has a background in usability with various systems that EV_2 implemented over the last five years on his own.

- Evaluator 3 (EV_3) is a researcher (Ph.D. in Artificial Intelligence) from the SCRE centre. EV_3's research interests include Intelligent Tutoring Systems and User Modeling. EV_3 does not have a background in usability testing; however, EV_3's background involves designing and implementing Intelligent Tutoring Systems.

7.4.4 Materials

We used a variety of materials to evaluate the usability of McFeSPA with the help of the evaluators such as a questionnaire, semi-structured interview sheet, observation sheet, software for screen capture and producing a video file and audio recording apparatus. The aim was to collect the evaluators' impression of McFeSPA interface. The materials used in the study are:

- Prototype of McFeSPA interface version 1.2

- Handout (see Appendix E) that included a description of how to perform the task and a handout used as a brief introduction for using McFeSPA e.g. operating the interface, and the kind of tasks involved.

- Camstudio is software for producing the screen capture video file, all screen activity from the Windows Desktop into AVI movie files, generated by McFeSPA while the evaluator was performing the required tasks. The software stored user interaction with the interface as well as the time of recording.

- Observation sheet (see Appendix E) generated by the researcher that noted while the evaluator was interacting with McFeSPA and performing the tasks. The checklist contains the specific question from Section 7.4.1.

- System checklist questionnaire (see Appendix E) which contains lists relating to the satisfaction of using McFeSPA's interface. This was completed by the evaluator relating to their opinion about using McFeSPA's interface based on their experience.

- Consent forms were signed by the evaluator to agree to do the experiment.

- Semi-structured interview sheet (see Appendix E) consisted of the evaluators' background information, information about the satisfaction of using McFeSPA's interface to be completed by the researcher to gauge their opinion in using McFeSPA interface based on their experience and comments with regard to assorted interfaces of McFeSPA and their effect on the evaluators' performance.

- Audio recorder used to complement the information provided by the previous materials to help the researcher identify evaluator's intentions while performing particular sequences of actions.

- Researcher Script used for helping the researcher to introduce the process of the experiment to the evaluator properly.

7.4.5 Procedure

The study was run individually and separately with each evaluator who followed the actions from the handout for usability evaluation as a form of Cognitive walkthrough method. This study allows the evaluators to comment while they are testing McFeSPA i.e. a using thinking aloud method. In addition the evaluators are encouraged to ask any questions about the evaluation relating to the tasks that they are required to perform during the evaluation while the researcher may ask questions to the evaluators at any time during the evaluation – a form of co-operative evaluation. The researcher set up the physical environment to collect the data from the evaluators via a semi-structured interview with co-operative evaluation and thinking aloud methods to produce direct observation and to supply the clarification of the interface's technique during the interaction when the evaluator needed to do so. After that the evaluators were asked to complete the questionnaire regarding satisfaction of McFeSPA. The typical sequence of events is described below:

- Preliminary set up: The researcher performs the following process before the start of the interaction.

- Check printed material (consent form, handout, researcher script, researcher checklist, observation sheet, system checklist as satisfaction questionnaire, semi-structured interview sheet).

- Informed Consent Form: The researcher handed out consent forms to the participants for them read and sign to ensure they were willing to be volunteers.

- Training using McFeSPA: Demonstration and practice using McFeSPA before starting work with McFeSPA to ensure that they did not have any usability problems during the formal evaluation process.

- Handout (section 1): The researcher asked the evaluator to read the first section of the handout (description and about McFeSPA).

- Handout (section 2): The researcher asked the evaluator to comment on the second section of the handout that consisted of describing the evaluator's background information.

- Start audio recorder, screen capture video file (Camstudio program), and McFeSPA's prototype: Camstudio program catches every single mouse event (together with a time stamp); thus, the researcher can monitor the evaluator's action and the duration of the activities after the experiment. Whilst the evaluators carry out the tasks, they can talk and think aloud by using co-operative evaluation.

-Semi-structured interview: The researcher interviewed the evaluator with the semi-structured interview sheet (using the interface) alongside the verbal description of the technique to be used during the interaction.

- Handout (section 3: Task 1): The researcher asked the evaluator to perform task 1 of section 3, which relates to using McFesPA's customization.

- Handout (section 3: Task 2): The researcher asked the evaluator to perform task 2 of section 3, which relates to using McFesPA to generate a feedback report with scaffolding by McFeSPA.

- System checklist as a questionnaire on satisfaction: The researcher asked the evaluator to complete McFeSPA checklist.

7.4.6 Results

After iteratively improving McFeSPA design through pilot evaluation, we performed a study to test McFeSPA's usability. The evaluation consisted of two tasks that took about one and half hours on average for each evaluator. Though the researcher suggested that the maximum time in the study would be no more than two hours, EV_2 took approximately 2.38 hrs while EV_1 took 1.07 hrs and EV_3 took 1.18 hrs. Those times performed by the evaluators do not include filling in the questionnaire and exploring McFeSPA with the researcher and training in using McFeSPA. The researcher also gave out McFeSPA's usability testing handout a week in advance. The activities that the researcher required the evaluators to do were: read the McFeSPA description and provide the evaluator's background in usability testing, comment on McFeSPA's interface via semi-structured interview, comment on tasks, and complete the check list questionnaire. In order to roughly equate time spent on a task, the evaluators were asked to mark three student's scripts. During the study, the researcher explored McFeSPA briefly (roughly) on one occasion to guide the evaluator in using McFeSPA and record log files (as avi file) of the evaluations' interaction with McFeSPA. The researcher then analysed the log data to understand how the evaluator used the interface of McFeSPA and how they reacted to McFeSPA together with the audio recording of the evaluator's comments and thinking aloud whilst using the interface. Finally the evaluators were asked to complete the questionnaire about their level of satisfaction in using McFeSPA. The results were examined based on an observation sheet, semi-structured interview, log file (avi file) and questionnaire.

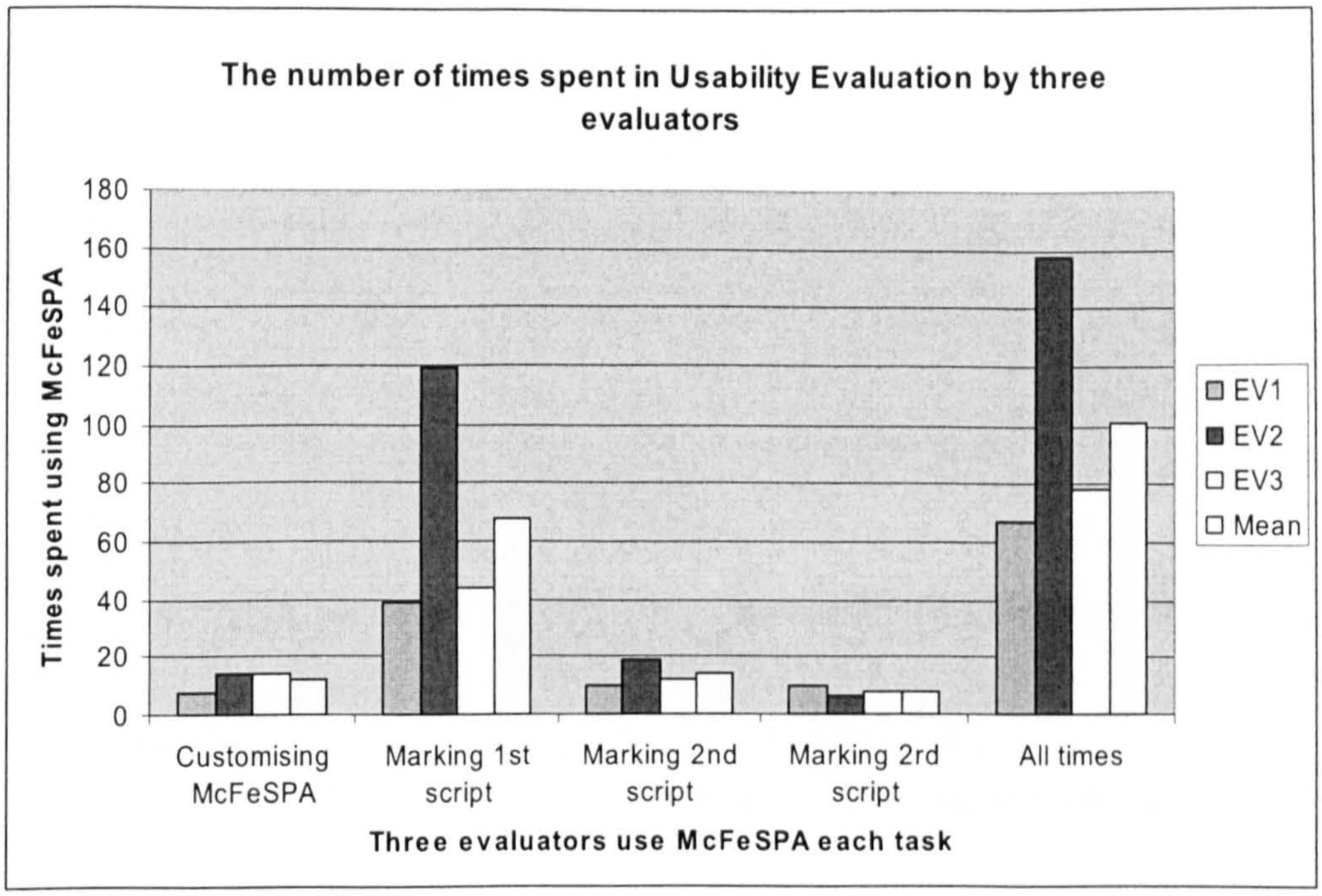


Figure 7.32 The number of times spent in usability evaluation of each evaluator

According to Figure 7.32, all evaluators spent time mostly marking 1st script. After marking the 1st script it appeared that they became familiar with the system. Overall, EV_2 is one who spent the most number of times in using the system compared with the others. The time EV_2 spent marking the 1st student's script was 6 times more than the 2nd scripts and 20 times more than the 3rd script. With regard to the actions of each evaluator that strayed from the solution path, McFeSPA offered help to get the evaluator back onto the correct solution path. Figure 7.33 reports the 'accept help'/ 'refuse help' of all contingent hints by three evaluators while Table 7.1 describes contingent 'accept help'/ 'refuse help' by time from EV_1, Table 7.2 from EV_2, and Table 7.3 from EV_3.

Table 7.1 Accept help (A) / Refuse help(R) of each contingent hint (in the parenthesis) by time of EV_1 e.g. A(5) means accepting help of the hint number 5

Time	3.47	3.49	3.50	3.51	3.52	3.52	3.52	3.53	3.57	3.58	3.58	3.58	3.58	4.00
A/R	A(2)	A(5)	R(5)	R(5)	A(5)	A(5)	R(5)	R(4)	R(1)	A(1)	A(1)	A(1)	A(1)	R(5)

Table 7.1 Accept help (A)/ Refuse help (R) of each contingent hint (in the parenthesis) by time of EV_1 e.g. A(5) means accepting help of the hint number 5 (contd.)

Time	4.01	4.07	4.28	4.28	4.29	4.30	4.30	4.31	4.31	4.32	4.32	4.35	4.37	4.39
A/R	R(5)	A(5)	A(6)	A(6)	R(4)	A(7)	A(7)	R(7)	A(12)	A(12)	A(12)	A(3)	A(3)	R(7)

Table 7.2 Accept help (A) / Refuse help(R) of each contingent hint (in the parenthesis) by time of EV_2 e.g. A(5) means accepting help of the hint number 5

Time	11.36	11.37	11.40	11.41	11.42	11.43	11.45	11.48	11.49	11.49	12.03	12.05	12.05
A/R	R(5)	R(5)	A(6)	A(6)	A(6)	A(6)	R(6)	A(1)	A(1)	A(1)	R(5)	R(5)	R(5)

Table 7.2 Accept help (A)/ Refuse help (R) of each contingent hint (in the parenthesis) by time of EV_2 e.g. A(5) means accepting help of the hint number 5 (1st contd.)

Time	12.13	12.14	12.17	12.18	12.19	12.29	12.34	13.41	13.41	13.42	13.42	13.42	13.43
A/R	R(7)	A(12)	R(12)	R(12)	R(12)	R(12)	R(5)	A(5)	R(5)	A(7)	A(7)	R(7)	A(12)

Table 7.2 Accept help (A) / Refuse help (R) of each contingent hint (in the parenthesis) by time of EV_2 e.g. A(5) means accepting help of the hint number 5 (2nd contd.)

Time	13.43	13.44	13.48	13.48	13.48	13.51
A/R	A(12)	A(12)	A(4)	R(5)	R(5)	R(7)

Table 7.3 Accept help (A) / Refuse help (R) of each contingent hint (in the parenthesis) by time of EV_3 e.g. A(5) means accepting help of hint number 5

Time	11.19	11.19	11.24	11.26	11.26	11.26	11.50	11.51	12.00
A/R	A(1)	A(1)	A(7)	A(7)	A(7)	A(7)	R(11)	R(8)	R(11)

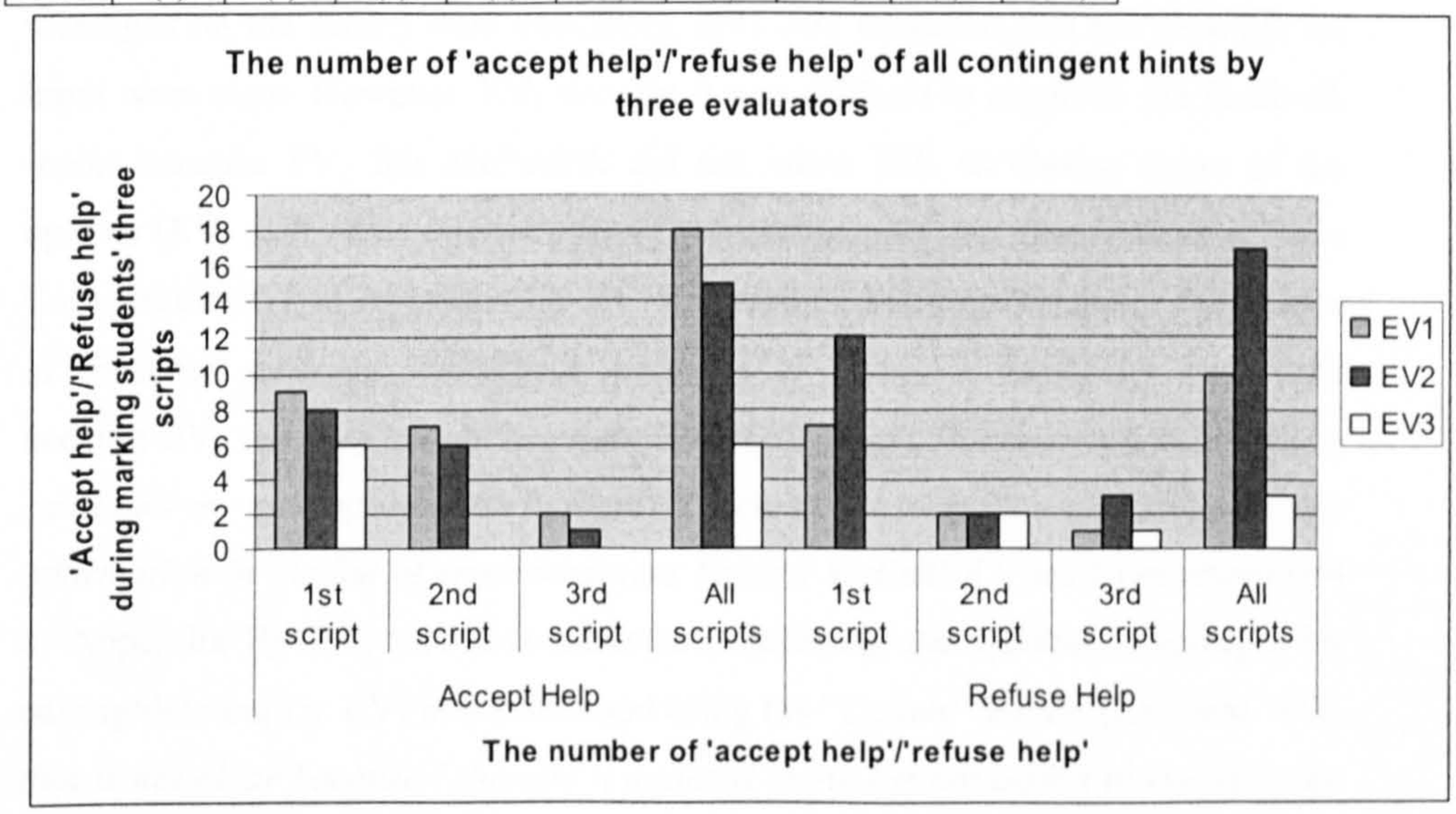


Figure 7.33 The number of 'accept help'/'refuse help' of all contingent hints by three evaluators

According to Figure 7.33, the number of EV_1's 'accept help' and 'refuse help' reduced respectively when EV_1 mark 1st, 2nd, and 3rd script respectively. The number of EV_2's 'accept help' reduced respectively when EV_2 mark 1st, 2nd, and 3rd script respectively. Although EV_2's 'refuse help' in marking the 2nd script reduced from the 1st script 6 times, it was small increased in marking the 3rd script. EV_3 is the one who 'accept help' in marking the 1st script. In marking all scripts, EV_1 is the one who is the most 'accept help' while EV_2 is the one who is the most 'refuse help'.

7.4.6.1 Evaluator I (EV_1)

EV_1 has several years experience in usability testing e.g. researching on usability testing processes, and participation in testing both commercial and educational products. EV_1 described 'Usability Testing' as "*process of determining whether a system satisfies the user in terms of ease of use and provision of function.*"

The following paragraph concerns EV1's results obtained by analysing the questionnaire, observation sheet, semi-structured interview, and the results of log data, which can be seen in Table F.1 – F.26 in Appendix F.

EV_1 found that it was very easy to edit the feedback report, to customize McFeSPA, and to read characters on the screen (see Conversation#1 in Appendix F). EV_1 agreed that the use of terms throughout McFeSPA and the position of messages on the screen were consistent. EV_1 also accorded that the prompts for input were clear. However, EV_1 thought it was difficult to organize the feedback report because EV_1 felt McFeSPA did not allow EV_1 to choose some of the options (EV_1 said "*This interface is very confusing so I can give it three ...*" see Conversation#11 in Appendix F). EV_1 commented that select/manage the choice of feedback messages, weakness messages or 'positive feedback' were easy because EV_1 felt they had different methods (EV_1 said "*The interface is not clear, can I add my own information in there?*"; Researcher said "*You can add/save this information in the list of error messages beyond McFeSPA*", see Conversation#4 in Appendix F). EV_1 also misunderstood managing the feedback messages by editing/deleting i.e. EV_1 misunderstood using the "Update" button (EV_1 said "*OK, that is not clear because I thought it updated to my list not added to my list*", see Conversation#2 in Appendix F). EV_1 also took issue that the language of the error messages was helpful because EV_1 thought a help offer from McFeSPA was an error message made by EV_1. EV_1 pointed out that the error messages were just frustrating. Finally, EV_1 found overall it was troublesome to use McFeSPA; EV_1's overall summary comment was "*Mainly, it was good, and a very good idea – it was, as always, the small things that gave me a negative impression. Overall, I think it is a good, worthwhile system- the small usability interface problems would be easily fixed.*"

We adopted the questions[23] of the Cognitive Walkthrough method (Lewis & Rieman, 1994) to ask the same questions respectively of the interaction of the evaluator and researcher working with the interface so that we can find some problems with the interface and to critique EV_1's action by using the results from the log data and audio recorded from the combination of think aloud method and semi-structure interview as below.

EV_1 did not try to produce whatever effect the action had if the interface was not responding to the effect according to EV_1's need. According to EV_1's log-file, there were some actions suggested by McFeSPA that EV_1 refused e.g. EV_1 did not accept help offered by McFeSPA, and did not consider the history of the student's profile account. EV_1 tried to carry out other actions to further the sequence of action – handout by the researcher i.e. EV_1 tried to customize some information in the 'Report Template' interface, by adding a further sentence in the implementation's pane of the main interface.

EV_1 could see the control (e.g. button/menu/text box) for the action but was confused about how the control works (e.g. add & update button in ManageData interface, (see Conversation#2 in Appendix F)); didn't understand the interface 'Exit' button is to exit McFeSPA in which it did not close the interface.

Once EV_1 followed the action, McFeSPA produced an effect that he did not expect i.e. EV_1 expected the 'student's profile' interface to be opened after the dialogue of "Do you want to take into account history of student's errors" was accepted. (EV_1 said "*When I click 'Yes' and what I am expecting to see, it is the 'student profile' and ...*", see Conversation# 5, in Appendix F)

After the action was taken, EV_1 understood the feedback message from McFeSPA and could go on to the next action with confidence even if the feedback message made EV_1 feel unclear about its meaning beforehand e.g. the help level 1 of the 'Choice for More errors' interface (see Conversation# 3, in Appendix F)

After the usability testing EV_1 also commented on McFeSPA as "*Well, the*

---

[23] 1) Will the users be trying to produce whatever effect the action has?
2) Will the users see the control (button, menu, switch, etc) for the action?
3) Once the user finds the control, will they recognise that it produces the effect they want?
4) After the action is taken, will users understand the feedback they get so they can go on to the next action with confidence?

important thing about this is that it is a very good and very useful system, a lot of work to program it, which is really good. The problems are small things that give a negative impression when using the system, they are not big things..." see Conversation#18 in Appendix F). EV_1 suggested possible improvements to the interface according to Table I.1 in Appendix I.

7.4.6.2 Evaluator II (EV_2)

EV_2 is experienced in usability testing by testing systems that he has implemented over the last five years. EV_2 described 'Usability Testing' as "*going though a scenario with a given piece of software in order to identify problems, issues, side-effects, etc. with the interface or the scenario itself.*"

The following paragraphs are the results obtained by analysing the questionnaire, observation sheet, semi-structured interview, and the result of the log data that can be seen from tables F.1 – F.26 in Appendix F.

EV_2 found that it was very easy to manage the feedback messages by editing/deleting and to generate the feedback report. EV_2 agreed that it was easy to edit the feedback report and to use McFeSPA's interface. EV_2 concurred that the use of terms throughout McFeSPA is consistent, the prompts for input were clear, and the terminology always related to the task. EV_2 also accepted that McFeSPA always provided a progress report but EV_2 argued that it was *"in a very annoying way*"). Nevertheless, EV_2 agreed that it was easy to organize the feedback report, and that the error messages were helpful because of mostly language problems (see Conversation#38 in Appendix F). EV_2 thought a help offer from McFeSPA was an error message he had made. Finally, EV_2 found overall it was simple to use McFeSPA but asserted that the problem was the initial explanation of tasks. This is not a primary problem because the handout was sent to EV_2 about a week before the testing date and EV_2 was trained to use McFeSPA.

We adopted 4 questions of the cognitive walkthrough method (Lewis & Rieman, 1994) to critique EV_2's action by using the results from the log data and audio recorded from the combination of think aloud method and semi-structure interview as below.

EV_2 tried to produce whatever effect the action has. If the interface was not responding the right effect according to EV_2's need, EV_2 did not produce the

required effect. According to EV_2's log-file, EV_2 refused some actions e.g. not accepting the help offered by McFeSPA, not taking into account the history of the student's profile. EV_2 tried to do other actions beyond the description of the action – handout by the researcher i.e. EV_2 tried to view 'Favourite content' interface, add further sentence in the design/implementation's pane in the 'Create Feedback Report' interface which McFeSPA does not allow to do see Conversation#37 in Appendix F).

EV_2 could see the control (e.g. button/menu/text box) for the action but some buttons' positions made EV_2 confused (e.g. a belief that the 'Add', 'Update' and 'Clear' button should be in the bottom left of the interface and the 'Delete' button should be under the list box but above the text box in the 'ManageData' interface, see Conversation#19 in Appendix F) and he understood that the 'Exit' button is to exit McFeSPA not to close the interface (see Conversation#36 in Appendix F).

Once EV_2 followed the action, EV_2 expected that the 'student's profile' menu item in the 'Choice for more errors' interface would be displayed for the student whose name appeared on the student's script being marked.

After the action was taken, EV_2 understood the feedback message from McFeSPA and could go on to the next action with confidence even when the feedback message made EV_2 feel its meaning is negative i.e. the help level 1 of the 'Choice for More errors' interface (see Conversation#28 in Appendix F). EV_2 suggested possible improvements to the interface according to Table I.1 in Appendix I.

7.4.6.3 Evaluator III (EV_3)

EV_3 has experience in designing and implementing Intelligent Tutoring Systems rather than usability testing. EV_3 described 'Usability Testing' as "*It is to do with estimating how easy it is to use a system, whether it does what it is designed for and what are the problems it poses to its users.*"

The following paragraph are the results obtained by analysing the questionnaire, observation sheet, semi-structured interview, and the result of the log data that can be seen in table F.1 – F.26 in Appendix F.

EV_3 found that it was very easy to use the interface, to customize McFeSPA (EV_3 said "*Oh! that's very easy.*"), to read characters on the screen, to manage the feedback message by editing/deleting, to edit the feedback report, and to generate the feedback report. EV_3 agreed that the organization of information and the sequence of screens were very clear. EV_3 concurred that the use of terms throughout McFeSPA and the position of messages on the screen are consistent, the terminology always related to the task, McFeSPA was always informative regarding his progress, and the prompts for input were clear. EV_3 was undecided about whether the error messages were helpful or not (*"because the feedback on progress was a little annoying."*). EV_3 thought a help offer from McFeSPA was an error message made by EV_3. Finally, EV_3 concluded that overall it was simple to use McFeSPA.

We adopted 4 questions to critique the story (Lewis & Rieman, 1994) by using the results from the log data and audio recorded from a combination of think aloud method and semi-structure interview as below.

EV_3 did not try to produce whatever effect the action has if the interface is not responding to the effect according to EV_3's need. According to EV_3's log-file, there were some actions that EV_3 refused the effect e.g. not accepting the help offered by McFeSPA, not taking into account the history of the student's profile. EV_3 tried to do other actions beyond the described actions – handout by the researcher i.e. EV_3 tried to edit the feedback message in the implementation's pane in the 'Create Feedback Report' interface in which McFeSPA does not allow this action.

EV_3 can mostly see the control (e.g. button/menu/text box) for the action but EV_3 could not file the 'Student Profile' menu item in the 'Add extra sentence after error messages' interface; thus EV_3 suggested the researcher improve the interface (see Conversation#46 in Appendix F).

Once EV_3 followed the action, EV_3 expected that the 'student's profile' menu item in the 'Add extra sentence after error messages' interface would display the student whose name appeared on the student's script being marked.

After the action was taken, EV_3 understood the feedback message from McFeSPA and could go on to the next action with confidence, even when the feedback message made EV_3 a little annoyed.

EV_3's suggested improvements to the interface are record in Table I.1 in Appendix I.

7.5 Analysis and Discussion of the Results

According to the results of the usability evaluation, we used multiple units of analysis from multiple sources of evidence based on log data from the avi file, semi-structured interview, McFeSPA satisfaction questionnaire, and the think aloud method. The component "efficiency of use" can be quantified as the average time it takes users to perform a certain number of specified tasks (Nielsen, 1993). The average time of usability testing from all evaluators is 101 minutes. EV_2 spent the longest time customising McFeSPA, marking the 1st script, and the 2nd script while all evaluators spent similar time in marking the 3rd script. EV_2 made more comments on using McFeSPA than EV_1 or EV_3 and also made more actions than the other evaluators. (EV_1 performed 415 actions, EV_2 performed 438 = actions, EV_1 performed = 326 actions).

We have adopted a subset of quantifiable usability measurement to analyse the data we collected from the usability evaluation (Nielsen, 1993).

EV_1 and EV_2 performed a similar number of actions, more than EV_3. This could be because EV_3 is the only one who never double selected the list item of analysed student's solution.

EV_1 spent slightly more time usability testing than EV_3 and made more comments than EV_3 (see Table I.1 in Appendix I). This could be because EV_1 has a strong background in usability testing. EV_3 accepted help rather than refused help, especially in the 1st marking when EV_3 knew he was making errors[24], EV_3 did not repeat the errors he made. EV_3 had the most success in trying to achieve his goal[25], and made fewer errors than EV_1, and EV_2. EV_1 and EV_2 sometimes knew the answers but they pretended not to know them in order to test McFeSPA (e.g. EV_1 said "*I know when I go click on 'None' I'm going to get trouble as well*"

[24] Error is the result of an action that the user is out of the step of the right solution path and was offered help by McFeSPA. There are two types of help response: refuse help and accept help.
[25] Success is the result of the action that the user reaches a step of the right solution path of McFeSPA

see Conversation#7, in Appendix F; EV_2 said *"I select this one while I know it doesn't work but I'm pretending. I want to see why."* see Conversation#30, in Appendix F). Unlike the other two EV_1 did not view the skill meter after finishing each marking but viewed after finishing marking all three scripts. EV_2 is had the most use of the glossary while EV_3 never used it because EV_3 had the most success without trying to find the answer by viewing the glossary. EV_3 examined the student's profile the most. Even though EV_1 does agree to receive feedback, it continuously affects the flow of the tasks.

We analysed the evaluators' actions based on the log data (avi file), semi-structured interview, and cooperation evaluation method as reported in the following paragraph according to the specific questions in Section 7.4.1.

1) How does the user add/update/delete data in 'Favourite Wording'/ 'Favourite Content'/ 'Report Template' menu from customise menu?

EV_1 managed data (Add/ Update/ Delete) in 'Favourite wording' interface better than EV_2 and EV_3. This could be because EV_1 is a senior lecturer and thus as more experience in giving feedback. EV_1 misunderstood how to use the 'Update' button in the 'ManageData' interface (see Conversation#2 in Appendix F). EV_1 tried to manage data in 'Report Template' interface but had no success in doing so because EV_1 did not select a word from the list box after managing the data which results in some changes not being changed according to EV_1's log-file. All evaluators tried to test 'Add', 'Update', 'Delete' button. EV_3 understood the most in using the 'ManageData' interface while EV_1 understood the most in using the 'Favourite wording' interface. Nevertheless, all evaluators understood using 'Favourite wording' interface quite well (see Table F.27, satisfied scales, Appendix F).

2) How does the user select a choice for one error (of the same error type) found?

EV_1 and EV_2 had more success than EV2 in selecting the choice from 'Choice for One Errors' interface. EV_1 accepted and refused help offers approximately the same number of times while EV_3 succeeded in selecting the option in this interface in every step. EV_2 mostly tried not to select the right option by accepting the defaulted option by McFeSPA. Both EV_1 and EV_2 tried to select the 'Say

nothing' option and 'None (just brief message)'. EV_1 and EV_3 preferred to indicate the error line number into the feedback report more than EV_2 while EV_1 preferred to include an example into the feedback report more than EV_2 and EV_3 although not every time. EV_2 seldom indicated the error line number and did not include an example into the feedback report because EV_2 suggested to the researcher that this should be improved (see Appendix I). Nevertheless, all evaluators did agree that the interface is easy to use.

3) How does the user select a choice for more errors (of the same error type) found?

Mostly EV_1 and EV_3 succeeded in selecting the 'Choice for More Errors' interface. EV_3 succeeded in every step of in the 'Choice for More Errors' interface, similar to 'Choice for One Errors' interface. EV_1 and EV_2 knew the right option but selected another option in order to test McFeSPA's usability (see Conversation#7, #30 in Appendix F). EV_3 did not refuse any help offered by McFeSPA in this interface, accepting help on two occasions which is similar to EV_1. EV_2 made the most errors in this interface by both accepting and refusing the help offered more often than EV_1 and EV_3. EV_1 performed the most number of actions in giving the example into the feedback report. All evaluators performed approximately the same number of actions in providing the 'indicate the error line number' option. All evaluators made an error by selecting the 'Yes (Always)' option to generate the error messages only once. This could be because they knew this is not the right answer. They then did not try to select it again. EV_1 and EV_2 did not select any option by accepting the defaulted choice ('None (just brief message)') i.e. EV_1 performed this two times while EV_2 performed this only once. EV_3 is the only one who never selected the 'Say nothing' option while EV_2 selected it because EV_2 thought some errors were not important and did not need adding into the feedback report (see Conversation#34 in Appendix F). Nevertheless, all evaluators agreed to some extent that this interface is easy to use. EV_2 and EV_3 gave the same level of satisfaction for the 'Choices for more errors' and 'Choices for one error' interface. EV_2 thought these interfaces easier to use than EV_3 while EV_1 thought the 'Choices for more errors' interface is more difficult to than the 'Choices for one error' interface because 'Choices for more errors' has more options than another.

4) How does the user provide extra sentences after selecting their choice of error messages?

EV_1 thought that the Student's profile' interface should appear before the 'Add extra sentence after error message' interface (see Conversation#5 in Appendix F). All evaluators suggested improvements to the 'Student's profile' interface (see the new interface in Figure 7.7) so that it only displays the history of the student currently being marked. (see Conversation#29, Conversation#54 in Appendix F). EV_1 and EV_2 mostly refused to add an extra sentence after the error message by taking into account the history of the student's profile. EV_1 was asked to add a sentence 16 times but only accepted to add a sentence on 5 occasions. From these five times, EV_1 succeeded in adding the sentence 3 times and also received help from McFeSPA. Even though EV_1 received help from McFeSPA, EV_1 refused to follow the help recommendations. Similarly, EV_2 was asked to add extra sentences 14 times but choose to add the sentences only twice. EV_1 and EV_2 both viewed the student's profile twice. Even though EV_2 viewed the 'Student's profile' interface before selecting a choice from McFeSPA, he still made errors and received the help offer by McFeSPA until succeeding. Unlike the others EV_3 accepted to add the extra sentence in the majority of the first 12 times (accepted to add the extra sentence for 9 out of 12 times). EV_3 tended to view the 'Student's profile' interface before selecting the choice from McFeSPA and EV_3 had the most success in selecting the right choice. Regarding their scale of satisfaction EV_2 and EV_3 thought the interface was easy to use whilst EV_1 did not think it easy to use.

5) How does the user provide a feedback message if there is no student's error analysed by McFeSPA?

While marking the 2nd script, McFeSPA could not find any design error. EV_1 did not know what was different between the left text box and the right box (see Conversation#15 in Appendix F). However, the other evaluators did not comment on this because the explanation of each pane in the interface is clear. EV_1 and EV_3 added design errors beyond the design error offered by McFeSPA twice and they refused to add the style errors while they were marking the 3rd script. EV_1 added his own error messages into McFeSPA and added them into the feedback report as "This program has an unreferenced variable" and an error message offered by

McFeSPA. EV_3 added design error messages as "Do xxxx is a design error" into McFeSPA then he deleted it. Thereafter EV_3 typed a message as "You are doing well with designing the program" and added them into the feedback report. However this is not an error message and it will be not added into McFeSPA EV_2 performed add extra error in a different way from EV_1 and EV_3. EV_2 refused all add design error and implementation error dialogues offered by McFeSPA but clicked the 'Add Extra Implementation' button next to the 'implementation' pane of the main interface (see Figure 7.1) and added the implementation error by using the default error messages from McFeSPA, not add any new error message, as "Be careful, this is a problem of implementation –passing the wrong type of arguments" into the feedback report. According to the scale of satisfaction, EV_2 and EV_3 thought this interface was easy to use while EV_1 regarded it has moderate.

6) How does the user select the feedback template?

According to the solution path of McFeSPA EV_1 succeeded in selecting the right feedback template at the first attempt. This could be because EV_1 is a senior lecturer and knew how to provide the appropriate feedback template for giving a feedback report to the student. However EV_1 also tried to select another template to see how McFeSPA would respond to the action and then EV_1 tried to accept and refuse the help offer by McFeSPA, while EV_2 never succeeded in selecting the right template. Mostly EV_2 refused the help offered from McFeSPA even though EV_2 tried to accept the help offered from McFeSPA sometimes. EV_3, succeeded in selecting the right feedback template by accepting the help offered from McFeSPA for 4 times, at which point he knew the right one, after which EV_3 achieved success. All evaluators understood this interface and provided the same scale of satisfaction.

7) How does the user generate a feedback report from 'create feedback report' window to be a final report?

EV_1 repeated the process of generating the final feedback report several times, and almost, succeeded in generating the final feedback report. EV_2 made the most errors in 'Create Feedback Report' interface because EV_2 intended to test the usability of McFeSPA. Even though EV_3 made some errors in this interface, EV_3

succeeded in generating the final feedback report. Thus, EV_3 had the best understanding of how to use this interface, which is also reflected by the satisfaction scale from Table F.27, Appendix F.

To sum up, all evaluators tested McFeSPA with a different perception. EV_1 and EV_3 commented on the usability problems while EV_3 tried to achieve the goal of the test.

Table I.1 in Appendix I justifies the evaluators' suggestions and some participants' suggestions (PT1 – PT3, PT5, and PT6 from Chapter 8) to improve McFeSPA. We changed most interfaces according to the evaluators' suggestions (see Table I.1 in Appendix I). Most of the changes were designed to adopt a different interface but there were some issues that addressed the design criteria. A small number of them are fundamental and affect the design of the system i.e. according to suggestion 7, reporting the evaluator's performance from the skill meter comes from the new principle of informing changes in the skill meter according to the number of times/words (a word to count a question in 'Asking question') shown of each evaluator's feedback skill as described below:

0 time/word to be informed as "Your skill meter hasn't increased, try to check your performance in giving [feedback type] feedback"

1-5 times/words to be informed as "Good, you progressed in giving [feedback type] feedback"

6-10 times/words to be informed as "Well done, you produced a lot of [feedback type] feedback"

11-15 times/words to be informed as "Very good, you performed very well in giving [feedback type] feedback"

More than 15 times/words to be informed as "Excellent, you are using the [feedback type] feedback frequently"

In order to facilitate the developer in re-implementing the system in the next version, we have sequenced the suggestions of the evaluators and the participants from high to low importance which relate to improving the next version of McFeSPA (see Appendix I). Furthermore, some usability issues where related to the pedagogical approach (some not important, some important) e.g. pop up encouraging messages (see suggestion#5 in Appendix F) is an important issue in

the pedagogical approach but we found from the results of the usability study that they did not encourage all evaluators. It is true that they are adults who do not need a pedagogical approach, which is the science and art of helping children to learn, and they felt annoyed when the messages were displayed and suggested that they are removed, or displayed somewhere other than the centre of the interface; As a result there was no pop up encouragement message in the evaluation of McFeSPA in the next chapter.

7.6 Summary

This chapter presents McFeSPA's interfaces and constrains. The evaluation study described in this chapter has shown that all evaluators, including an expert, were able to understand and operate the mechanisms incorporated into the prototype Human-Computer interface described in Section 6.2, which was presented in the design of the Scenario-Based Scaffolding System. Nevertheless, "breakdowns" (Winograd & Flores, 1986) can happen in situations that the developer could not expect them to happen.

Among the changes to our interfaces suggested by the evaluators and participants (in the evaluation of McFeSPA from Chapter 8), most were aimed at improving the interface. Some issues, however, do impact on the pedagogical approach e.g. a message pops up when the users give the right feedback for each action, but all evaluators suggested that these messages should be removed (see suggestion#5 in Appendix I) because they felt they were distracting them.

In this chapter we have called our participants "users" or evaluators. In the next chapter our participants will be TAs. In order to help the TAs improve the quality of their feedback, we provide them with pre-constructed student scripts that will provide them with suitable opportunities to give feedback and also give a starting point to the analysis of the scripts through simulating the effect of having a run time analyser.

In McFeSPA, the user can add new student scripts but since there is no run time analysis of the scripts, the users will have to provide their own analysis of the errors in the new scripts. The run time analyser is part of McFeSPA's

architecture, (see Chapter 5) but it was decided there was no need to implement the run time analyser for this research - it can be done later.

In sum, the point of this chapter is interface design through a usability study to try to eliminate any usability problems. We now expect that the revised version will be adequate for the purpose of demonstrating how McFeSPA can contribute to helping the TAs to learn how to give feedback. The next chapter presents the evaluation study that was carried out to evaluate the prototype of McFeSPA in terms of how well users might be able to learn to give feedback.

Evaluation of McFeSPA's Learning Environment

8.1 Introduction

In previous chapters, we described how we designed and implemented McFeSPA's approach based on our literature review of how to give quality feedback (in Chapter 2); feedback design (in Chapter 3); analysis of student programming weaknesses relating to Prolog programming (in Chapter 4); scaffolding system design (in Chapter 5); and scenario-based scaffolding system design (in Chapter 6). In addition, we interviewed experts on feedback, on programming, and on Prolog programming in particular (see Appendix A). We then built the initial system and performed several studies to test McFeSPA's usability after iteratively improving the McFeSPA design through pilot evaluation. The last usability evaluation of McFeSPA was presented in Chapter 7. The empirical study in this chapter was designed to examine whether the system helped TAs to give feedback with the help of McFeSPA, and usability issues indirectly. According to Nielsen (1993; 2000) we could argue that the numbers of participants involved in our study is sufficient to examine the majority of usability problems and to measure the learnability of the system. To follow we detail the hypotheses, specific questions, methodology, participants, materials, procedure, results, analysis of the results, summary of the analysis of the results, discussion, and summary of this chapter.

8.2 Hypotheses

We proposed several hypotheses in Chapter 1. In order to aid the reader, we repeat them again in relation to whether McFeSPA meets the aim of the research.

1) Providing content scaffolding (i.e. detailed feedback using contingent hints) in McFeSPA could help TAs increase their knowledge/ understanding of issues about learning to give feedback.

2) Providing metacognitive scaffolding (i.e. each level of detailed feedback in contingent hints) in McFeSPA could help the TAs reflect/rethink his/her skills in giving feedback.

3) When the TAs obtains knowledge of giving quality feedback, providing adaptable fading of McFeSPA could allow the TAs to learn alone without any further support.

8.3 Specific Questions

The questioning of the evaluators was based on a number of learnability aspects of feedback giving using McFeSPA, and usability issues indirectly. The following questions were selected for this purpose. We use a triangulation method (Yin, 1994) to ensure the answers to these questions are consistent with all the evidence obtained.

1) Is the basic idea of McFeSPA helpful?

2) What did the participants learn by using McFeSPA?

3) How do the participants learn to provide each type of feedback according to McFeSPA?

3.1) How do the participants learn to provide the 'Feedback loop' and 'Individual feedback' messages?

3.2) How do the participants learn to provide 'Detailed/Elaborative feedback' messages?

3.3) How do the participants learn to provide 'Important/ Specific feedback' messages?

3.4) How do the participants learn to provide 'Positive feedback' messages?

4) How does McFeSPA help the participants increase their knowledge on the issue of learning to give feedback?

5) Can participants learn more about giving quality feedback by using McFeSPA again?

6) What are the participants' perspectives in the experimental group of receiving help messages?

6.1) In the experimental group, were the help messages easy to understand?

6.2) In the experimental group, were the help messages useful?

6.3) Did the help messages help the participants in the experimental group improve their skills in giving feedback?

7) Is it possible to learn how to use McFeSPA without any assistance from the system?

8) How does McFeSPA help the participants to reflect/rethink on their skills in giving feedback?

9) What are the participants' perspectives about the representation of their skill in giving feedback?

9.1) Did the representation of each participant's skill in giving feedback after using McFeSPA make each participant realise that he/she needs to improve his/her skills at giving feedback?

9.2) Did representation of each participant's skill at giving feedback (skill meter) help each participant think more about his/her skills?

10) What are the participants' perspectives about effectiveness of using McFeSPA?

10.1) Do the participants agree that McFeSPA can help them to give quality feedback to their students?

10.2) Did McFeSPA help participants to finish work quickly, effectively and improve their productivity?

10.3) Would the participants like to use McFeSPA in giving feedback to their students?

11) Can the participants apply the knowledge they obtained by using the system to mark any programming assignment (not only Prolog)?

12) What is the participants' level of satisfaction using McFeSPA?

12.1) Did the participants enjoy learning with McFeSPA?

12.2) Can McFeSPA be frustrating?

12.3) What did the participants like in particular about McFeSPA?

12.4) What did the participants dislike in particular about McFeSPA?

8.4 Methodology

We use multiple methods to reduce inappropriate certainty (Robson, 2002) and to clarify ambiguous[26] data (Molka-Danielsen, 2000) in the evaluation of McFeSPA. Our methods can be described as a combination of a form of cognitive walkthrough (Rieman et al., 1995), thinking aloud (Ericsson & Simon, 1993), structured interview, questionnaire, and comparison of pre-test and post-test. The evaluation begins with a pre-test, then a form of cognitive walkthrough. There are two TA groups to test the system. One is an experimental group which tested the system with help offered by the system and then used the system without any help from the system. Another is a comparison group, who tested without any help being offered by the system. We have provided a task description with several screen captures of the interface (see Appendix E, G), and a correct action sequence for marking a Prolog programming assignment by analysis of student's weakness (see Chapter 4) through to generating a feedback report. As each TA stepped through the sequence of actions required to accomplish each task, his/her actions were screen-recorded to an avi file which provides not only an indication of some of the errors made by the TA but also the opportunity to provide a quantitative measure of the TA's performance e.g. which mechanisms were used

[26] Ambiguous: having or expressing more than one possible meaning, sometimes intentionally (Cambridge advanced learner's dictionary, 2003)

and when they were used. The audio tape recording was used to identify some of the possible intentional changes made by the TA, based on reactions such as surprise and agreement when the actions were taken. During the use of the system, the TA was encouraged to think aloud and ask questions to the experimenter once the TA had finished using the system. Thereafter the TA was asked to complete a questionnaire and was interviewed to find out their perspective on their improvement through learning with the system.

8.5 Participants

In order to perform an evaluation of McFeSPA that met our requirements, we sought six relatively inexperienced TAs who had some familiarity with Prolog Programming (or have previously used Prolog as a learning tool; or understand simple Prolog code) and had marked some programming assignments (these did not need to be Prolog ones) and who were willing to take part in a study on how to improve feedback. In our plan we wanted three inexperienced TAs to use McFeSPA with scaffolding (the experimental group) and another three to use the system without scaffolding (the comparison group). It proved difficult to find TAs with very little experience, so the plan was adapted.

8.5.1 Participant 1 (PT1)

PT1 is a researcher (Ph.D. in communication and collaborative systems specialising in cognitive science and artificial intelligence) at the School of Informatics, University of Edinburgh. PT1's work in academia falls under two categories: teaching (as tutor and teaching assistant for 4 years) and research. PT1 has lectured on Artificial Intelligence Programming in Prolog and assisted with the teaching of courses on Prolog, Cognitive modeling, Advanced Interactive Learning Environments, and Statistics and Experimental Methodologies. PT1 has also tutored on numerous AI and Cognitive Science courses.

8.5.2 Participant 2 (PT2)

PT2 is a Ph.D. student at the School of Informatics, University of Edinburgh. PT2 completed the MSc. in Informatics, at the University of Edinburgh and was a

lecturer at the University Tenaga Nasional, Malaysia for two years. PT2's teaching includes Prolog programming. PT2 also worked as a software engineer with Altion Ltd in Dublin, Ireland for three years. Currently, PT2 is a teaching assistant which requires being a demonstrator/marker for Functional Programming, Computation and Logic, Object-Oriented Programming, and Data and Analysis.

8.5.3 Participant 3 (PT3)

PT3 is an M.Sc. student at the Department of Computing Science, University of Glasgow. PT3 has familiarity with Prolog (as part of PT3's undergraduate studies) and has marking experience since PT3 was a teaching assistant in the department for two years.

8.5.4 Participant 4 (PT4)

PT4 is a Ph.D. student in the SCRE centre, University of Glasgow. PT4 learnt Prolog over ten years ago. PT4 has only slight familiarity with Prolog, but has several years of experience in marking C programming assignments.

8.5.5 Participant 5 (PT5)

PT5 is a researcher (Ph.D. in the centre for intelligent systems and their applications) at the School of Informatics, University of Edinburgh. PT5 has three years experience in teaching and marking assignments, especially in teaching Prolog programming.

8.5.6 Participant 6 (PT6)

PT6 is a Ph.D. student in the centre for intelligent systems and their applications at the School of Informatics, University of Edinburgh. PT6 had teacher training with PGCE programmes in education. PT6 also has experience in teaching in secondary schools, sixth form colleges, tutoring at the introductory level at university, and some experience of teaching about Prolog programming

8.6 Materials

We used a variety of materials to evaluate the McFeSPA learning environment and learning gained through using McFeSPA. These materials included a questionnaire, structured interview sheet; observation checklist; screen capture video file, audio recording; and a pre-test and a post-test. These resources were chosen to collect the TA's perspective in learning with McFeSPA. In more detail, the materials used in the study were:

- Prototype of McFeSPA interface version 1.3

- Demonstration file generated by the experimenter to train the participants to use McFeSPA.

- Handouts which included the directions needed to perform the task and a brief user manual handout describing how to use the system e.g. using the interface, and tasks.

- Camstudio, a tool that records all screen activity from the Windows Desktop into AVI movie files generated by the system while the participants are performing the task required. It stores participants' action with the interface as well as the time of the recording.

- Observation sheet generated by the experimenter which was completed whilst the participants were interacting with the system and performing the tasks. The checklist contains the specific questions listed above.

- System checklist, used as a satisfaction questionnaire, that contained a checklist about the usability of the system, and a checklist about feedback regarding the participant's experience in using McFeSPA. Participants were required to complete these checklists detailing their opinions in using the system based on their experience.

- Consent form to be signed by the participants to record their agreement to take part in the experiment.

- Structured interview sheet which was used to gather the participants' background data, data about the satisfaction of using the McFeSPA's interface,

data about the participants' perspective in learning by using the system to be completed by the experimenter on their opinion based on their experience and comments with regard to assorted interfaces of the system and their effect on the participants' performance.

- Audio recording intended to complement the information provided by the previous materials to help the experimenter identify the participants' intention while performing particular sequences of actions.

- Experimenter Script intended to help the experimenter introduce the process of the experiment to the participants properly.

- Pre/post test paper intended to help the experimenter examine the participants' previous knowledge in giving feedback and the knowledge they gained after using the system. The Pre/Post test was used in order to measure the participants' understanding of giving good feedback before and after using McFeSPA. The pre/post test paper was designed to measure giving 'detailed/elaborative feedback', 'important/specific feedback', 'feedback loop' and 'Individual feedback', and 'feedback pattern' as feedback form. The choices of giving feedback presented in the pre/post test are quite similar to the choices in the system. However, the participants were asked to write the reasons for each selected answer.

8.7 Procedure

The study required each participant to take part individually and separately, with each participant following the actions from the McFeSPA user guide handout. The participants were encouraged to comment on the system after they finished using the system. The participants were divided into two groups. The evaluation of the first group consisted of two tasks (task one, using McFeSPA with scaffolding, task two using the system without scaffolding) while the evaluation of the second group involved using McFeSPA without scaffolding only. The participants were asked to do a pre-test before using the system and a post-test after using the system. The experimenter was responsible for setting up the physical environment to collect the data from the participants via Camstudio.

After that the participants were asked to complete a questionnaire regarding satisfaction with the system. Finally, the participants were interviewed using the structured interview sheet. The typical sequences of events are described below:

- Preliminary set up: The experimenter checked printed material (consent form, pre-test paper, handout, experimenter script, experimenter checklist, system checklist (satisfaction questionnaire), structured interview sheet, post-test paper) before the interaction started.

- Informed Consent Form: The experimenter handed out an informed consent form to the participants to read and signed to ensure the participants were willing to be volunteers.

- Pre-Test Process: Before starting the pre-test, the experimenter had to be sure that the participants understood all the instructions in the paper test. To do this, before starting the process some practice questions were asked to check the participants' understanding and also to give them a chance to ask any questions until they were satisfied that they understood what to do.

- Training using the system: A pre-recorded demonstration of using the system was provided and the participant practiced using the real system before starting work with the system to ensure that they did not have any usability problems during the evaluation process.

- Started audio recorder, Camstudio, and the McFeSPA prototype: Camstudio catches every single mouse event (together with a time stamp); thus, the experimenter can monitor the action of participants and their activity durations after the experiment. During performing the task the participants were encouraged to think aloud. Meanwhile using the system each participant's action was recorded as log-files to understand how each participant used McFeSPA, how they reacted to McFeSPA together with the audio recording with the participant's comments and thinking aloud whilst using the interface.

- Handout (section 1) Process: The experimenter asked the participant to read the first section of the handout (description about McFeSPA).

- Handout (section 2: Task 1) Process: The experimenter asked the participant to perform section 2 task 1, which related to using McFesPA to generate a feedback

report with scaffolding by McFeSPA (this task was not tested by members of the comparison group)

- Handout (section 2: Task 2) Process: The experimenter asked the participant to perform section 2 task 2, which related to using McFesPA to generate a feedback report without scaffolding by McFeSPA

- Post-Test Process: Immediately after finishing the learning session with McFeSPA, the experimenter handed out the Post-Test which was similar to the Pre-Test, but contained a different order sequence to measure the knowledge improvement of participants.

- System checklist as a satisfaction questionnaire: The experimenter asked the participant to complete the questionnaire in order to collect data regarding to the participants' level of satisfaction when using McFeSPA.

-Structured interview: The experimenter interviewed the participants with the structured interview sheet in order to collect the participants' background information, judge the efficiency of the process when using McFeSPA, and the participants' improvement of giving better feedback.

8.8 Results

The results were examined based on the triangulation approach (Yin, 1994) which contains many sources of data collection i.e. observation checklist, log-files, questionnaire, interview, pre-test, post-test. Overall activities took about two hours on average for each participant.

By using McFeSPA with scaffolding, all participants in experimental group (PT1, PT2, and PT3) marked three scripts in 38, 26, and 14 minutes respectively. Later they use McFeSPA without scaffolding and mark the same three scripts in less time than previously i.e. 20, 7, and 7 minutes respectively. PT1 always viewed the skill meter after marking each script but later PT1 marked the same three scripts in a different order. PT2 viewed the skill meter after marking each script with scaffolding by McFeSPA. Later, PT2 viewed it after re-marking the 2nd, and the 3rd script. PT3 viewed the skill meter only once after

marking all scripts. PT1 and PT2 needed between 5 and 10% of overall time spent on training to use McFeSPA while PT3 needed 10%. With regard to their actions in giving each type of feedback to the student's feedback report, each participant's log-file results, pre- and post-test results are shown in Table 8.1 for PT1, Table 8.2 for PT2, and Table 8.3 for PT3 respectively. In these tables, the 'successes' means that appropriate answers were given. It means each decision made by the participant is judged as appropriate or not using the principles followed by McFeSPA. If not appropriate, this is scored as an 'error'. If appropriate, this is scored as a 'success' so each decision is categorized in one of two ways. The percentage of 'refuse help' in this table means the ratio of the number of 'refuse help' by the combination of the number of 'refuse help' and 'accept help' in particular for each type of feedback.

Figure 8.1 presents all participants' percentages in the experimental group of successes for each type of feedback, 'refuse help', and 'accept help' for each contingent hint. In using the system marking student's script the results in the graph show that, PT1 and PT2 improved giving 'Detailed/Elaborative feedback' 13.33% and 22.22% respectively while in giving 'Feedback loop' and 'Individual feedback'. PT2 and PT3 improved 30.77% and 5.55% respectively. In giving 'Important/Specific feedback', PT1 and PT2 improved 25% and 28.57% respectively while PT3 always gave such feedback. In giving 'Positive feedback', PT1 and PT2 improved 44.44% and 17.60% respectively while PT3 always gave such feedback. PT2 is the one who accepted the most help of all while PT1 accepted help only for 'Positive feedback' and PT3 only for 'Feedback loop' and 'Individual feedback'. PT3 is the one who refused help more than accepted help in giving Feedback loop' and 'Individual feedback' 50% while PT1 refused help in giving 'Positive feedback' but overall 'accepted help' 75% of possible time. PT2 is the one who most refused help from giving feedback except for 'Important/Specific feedback'. In the paper test, PT2, and PT3 improved giving 'Feedback loop' and 'Individual feedback' 16.67%. PT1 and PT2 always gave 'Detailed/Elaborative feedback'. They also improved giving 'Important/Specific feedback' 33.33% while PT3 always gave this feedback. PT1 improved giving 'Positive feedback' 100% while PT2 and PT3 always gave such feedback.

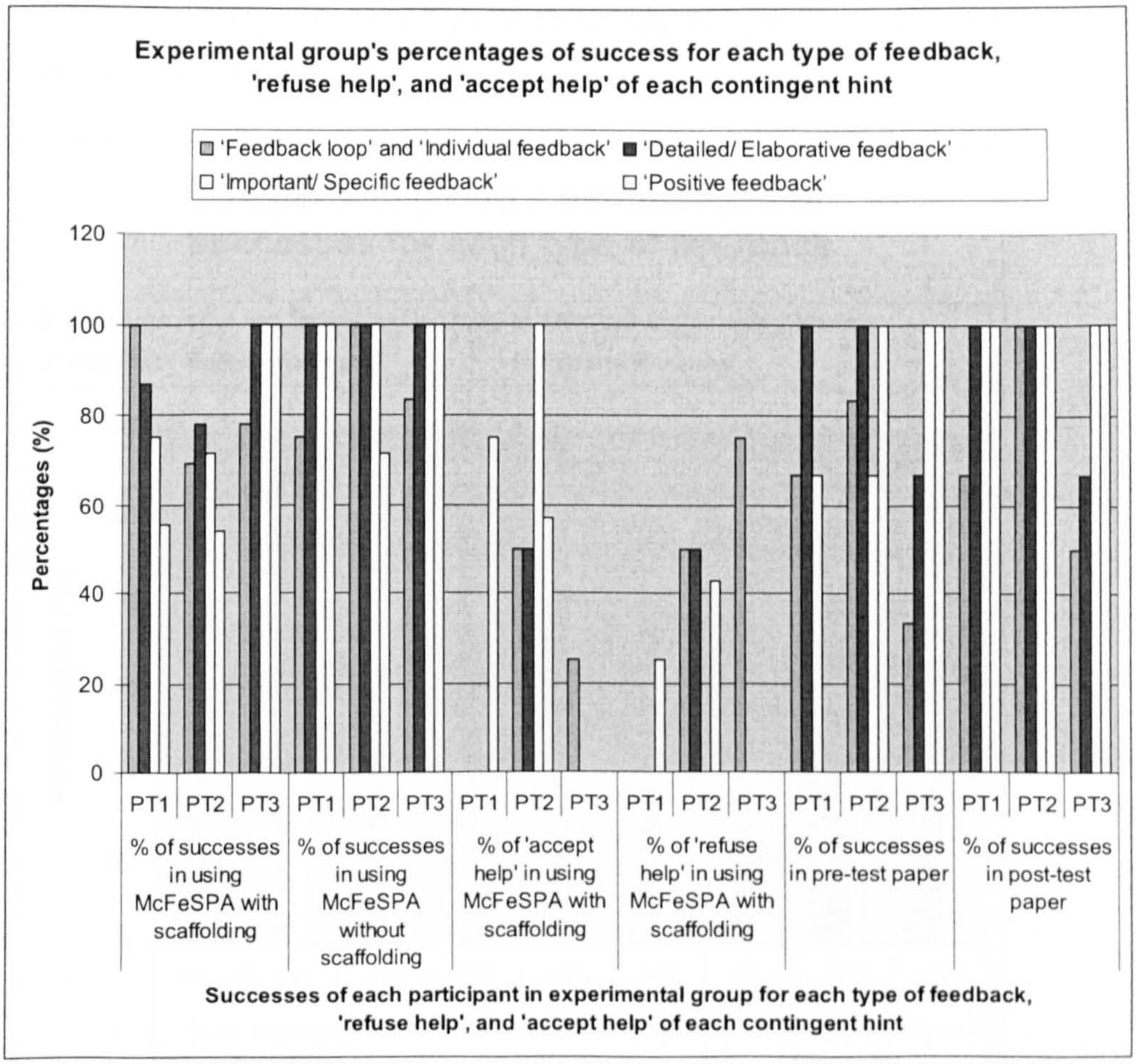


Figure 8.1 Experimental group' percentages of successes for each type of feedback, 'refuse help', and 'accept help' of each contingent hint

In using McFeSPA without scaffolding all participants in comparison group (PT4, PT5, and PT6) marked the three scripts in 11, 24, and 17 minutes respectively. They viewed the skill meter once after marking all scripts. PT4 and PT5 needed 5-10% of time spent on training to use McFeSPA while PT5 needed 2% of time spent on training to use McFeSPA ("The system is very easy to use and I think I don't need any training at all...It is pretty obvious what I have to do."). With regard to their actions in giving each type of feedback to the student's feedback report, the results of their log-files, and pre- and post-test is shown in Table 8.4 for PT4, Table 8.5 for PT5, and Table 8.6 for PT6 respectively. In these tables, the meaning of 'successes' is similar to that in Table 8.1. Figure 8.2

presents all participants' percentages in the comparison group of successes for each type of feedback.

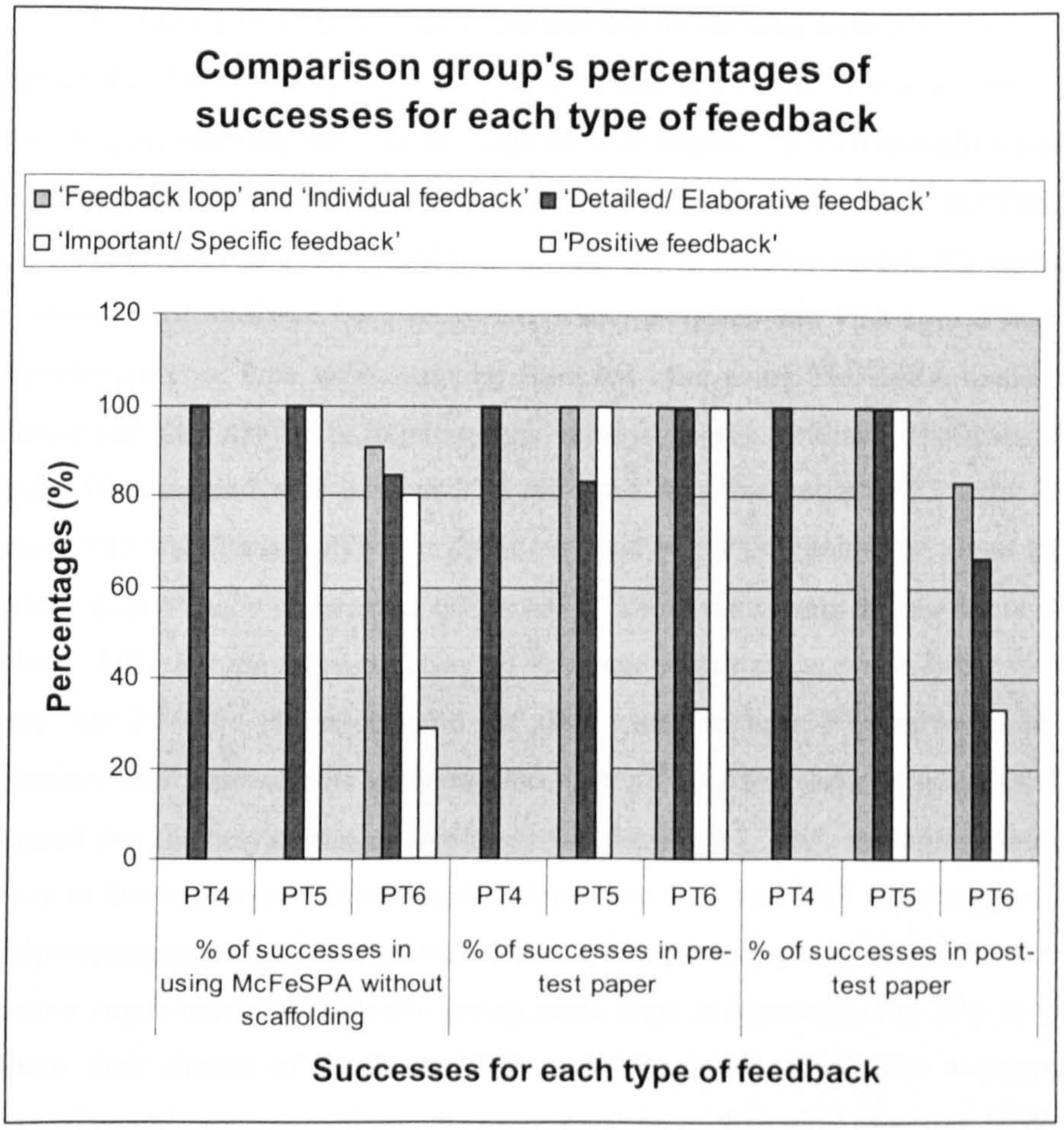


Figure 8.2 Comparison group' percentages of successes for each type of feedback

According to Figure 8.2, PT4 always gave 'Detailed/Elaborative feedback' for both using the system and both pre- and post-test. In marking students' scripts with the system, PT6 is the one who gave 'Feedback loop' and 'Individual feedback' while the other did not provide this feedback at all. However, PT5 always gave 'Feedback loop' and 'Individual feedback'. PT6's giving 'Feedback loop' and 'Individual feedback' reduced in the post-test 16.67% when compare with the pre-test while PT5's giving 'Detailed/Elaborative feedback' increased 16.67%. For both using the system and paper test, PT5 is the one who always

gave 'Important feedback'; however PT4 and PT5 never gave 'Positive feedback' at all.

According to all participants' perspective in learning with McFeSPA, they agreed that the basic idea of McFeSPA is helpful (PT1 said "*It is a good tool for speeding the marking task and the scaffolding is useful....*"). PT6 thought it helped evaluate what feedback PT6 gave (*"It would greatly help when marking an assignment. Even if it isn't used (e.g. assignment in another topic), it's useful to evaluate what feedback I'd give."*). Experimental group and PT4 agreed that the representation of their skill in giving feedback after using McFeSPA made they realise that they needed to improve their skills at giving feedback. However, PT5 and PT6 disagreed with this but PT5 did think that this helped PT5 think more about PT5's skill while PT6 thought this did not help PT6 think more about PT6's skill (*"Didn't really understand this point"; "Using the system helped me to think about skills, but the representation on this page didn't mean much to me."*). It is true that PT6 did not understand the skill meter because PT6 never used the glossary that provided the meaning and example of each skill. In addition, PT2 agreed that the help messages were easy to understand (*"They are quite standard, easy to learn after several attempts"*). However, PT1 and PT3 did not agree that help messages are easy to understand because help messages in McFeSPA started with a vague message gradually giving more hints to encourage the TAs to think about their giving of feedback (PT3 said "*Not understand help messages*"). Experimental group agreed that the representation of their skill at giving feedback (skill meter) helped them think more about their skill. PT1 and PT2 agreed that the help messages facilitated them improve their skills in giving feedback (PT2 said *"They are quite standard, easy to learn after several attempts"*) while PT3 disagreed with this because PT3 mostly refused help from the system and received only the top level of help which is vague ("*Never used them*"). All participants enjoyed learning with McFeSPA (PT6 said *"Very nice system and session. Good to have paper exercised too."*). Most participants would also like to use McFeSPA in giving feedback to their students (PT6 said *"If I were marking Prolog, this could greatly help."*) except for PT5 because PT5 does not teach Prolog anymore. Experimental group and PT4 thought McFeSPA could help them finish their work quickly, effectively and productively; however, PT5 disagreed

with this because PT5 thought McFeSPA should be more automated particularly in giving 'Feedback loop' and 'Individual feedback' with regard to history of student's errors while PT6 thought McFeSPA could not help PT6 finish quickly ("*I've only used in experiment, so it hasn't helped as yet. If I was using it to mark, it would be useful.*"). However, PT6 felt it could help PT6 finish work effectively. Nevertheless, PT1 and PT3 felt frustrated with McFeSPA when PT1 made a mistake according to McFeSPA and did not know what the mistake was while PT3 thought it might be complex program.

The following are each participant's results obtained by analyzing the pre-post test, log-files, observation checklist, questionnaire, and structured interview.

8.8.1 Participant 1 (PT1)'s results

PT1 had quite a lot of experience in giving feedback because PT1 had previously had some skills training in this area. PT1 had been a teaching assistant and felt that the school he had been working in had provided courses to train people to teach students in tutorials and how to provide feedback.

Table 8.1 PT1's percentages of successes for each type of feedback (between using McFeSPA with and without scaffolding and between pre- and post-test), refuse help and accept help of each contingent hint for each type of feedback.

Issue	Type of feedback			
	'Feedback loop' and 'individual feedback'	'Detailed/ Elaborative feedback'	'Important/ Specific feedback'	'Positive feedback'
% of successes in using McFeSPA with scaffolding	100	86.67	75	55.56
% of successes in using McFeSPA without scaffolding	75	100	100	100
% of 'accept help' in using McFeSPA with scaffolding	0	0	0	75
% of 'refuse help' in using McFeSPA with scaffolding	0	0	0	25
% of successes in pre-test paper	66.67	100	66.67	0
% of successes in post-test paper	66.67	100	100	100

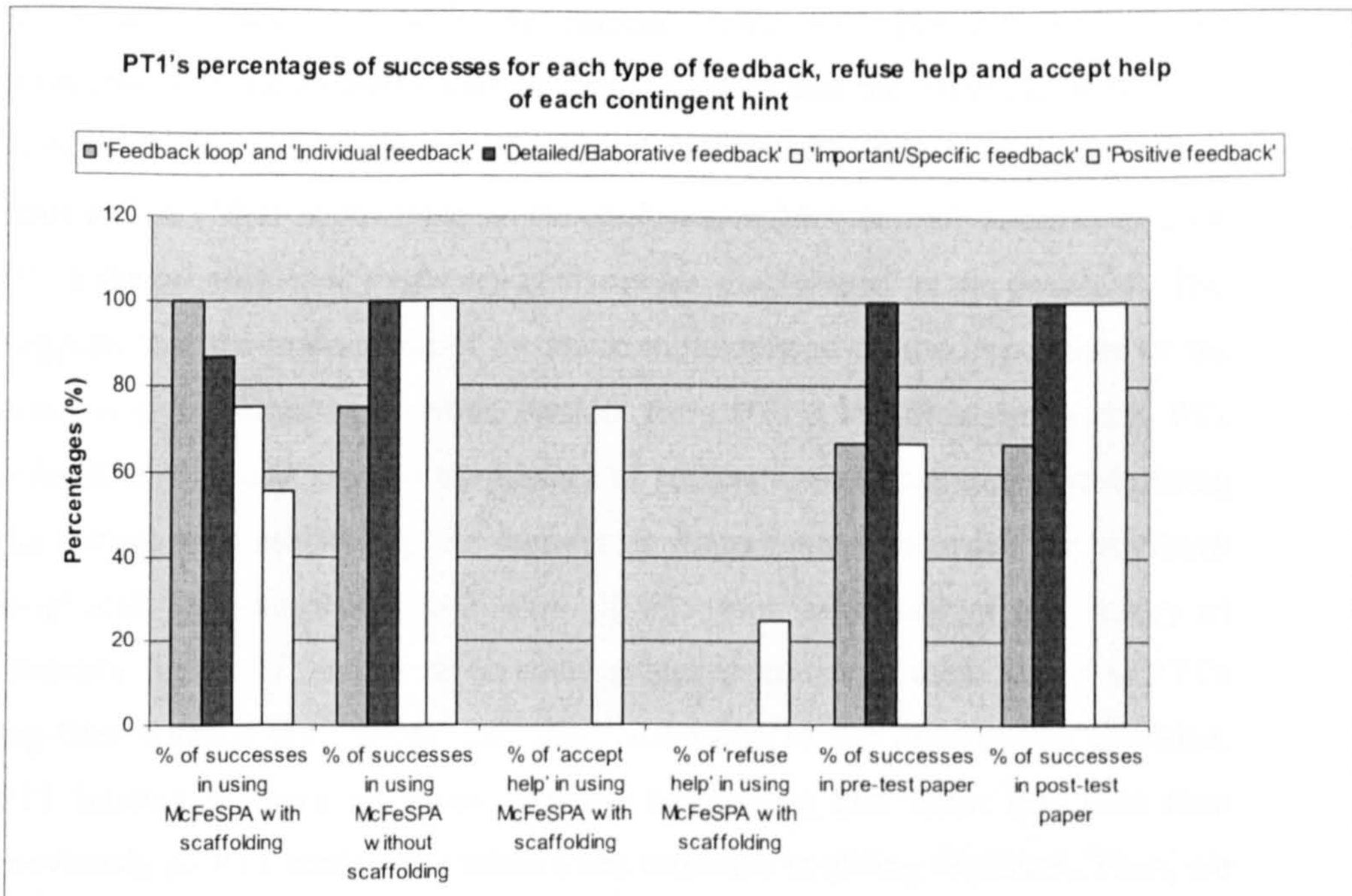


Figure 8.3 PT1' percentages of successes for each type of feedback, refuse help, and accept help of each contingent hint

According to table 8.1 and Figure 8.3, in using the system for marking students' script, PT1 improved giving 'Detailed/Elaborative feedback, 'Important/Specific feedback', and 'Positive feedback' 13.33%, 25%, and 44% respectively. PT1 always gave 'Detailed feedback' in the paper test and the result of PT1's giving 'Important/Specific feedback', and 'Positive feedback' increased 33.33% and 100% respectively. PT1 accepted help from the system more than refused help 50%.

According to table 8.1, PT1's giving of the 'feedback loop' and 'Individual feedback' for both pre-test and post-test did not improve because PT1 did not select the appropriate answers. Regarding PT1's perspective in giving such feedback from the pre-test, PT1 thought it was good to point out that students have made the error again but PT1 did not like the negative aspect of "*next time you could avoid this type of error*". In McFeSPA, the implementation of giving the 'feedback loop' and 'Individual feedback' considers the history of student's

errors and compares it with the current errors. PT1 pointed out that the comparison of the student's current type of errors and the previous ones is not important, it was just necessary to inform the student that they have made such an error before ("*It is stylist thing so the student shouldn't be only reprimanded for it.*" in the pre-test, and "*frequency of errors isn't important*" in the post-test). This suggests that the explanation of feedback could depend on the importance of the error, not the number of errors. Results from PT1's log-files shows that PT1 refused to take into account the history of student's error four times while using the system with scaffolding. So there is no improvement recorded for 'feedback loop' and 'Individual feedback' skill. If PT1 took into account the history of student's errors, PT1 might make some mistakes similar to those found in PT1's log-files without scaffolding. Likewise, according to the observation checklist, PT1 hurried to mark all scripts without scaffolding and spent less time than previously so PT1 could have made some mistakes in giving feedback. Thus, we could not say that PT1 did not improve giving such feedback. In addition, PT1 preferred to give motivational comments rather than giving more explanations based on the student's history of simple errors. Thus, this corresponds with the interview results.

Results from the paper test shows that PT1 always gave 'detailed/elaborative feedback' to all types of errors for both pre- and post-test but did not always give such feedback while using McFeSPA. PT1's log-files and observation, with scaffolding, showed that PT1 pretended to make a mistake in giving such feedback so results from PT1's log-files contrast to results from pre- and post-test.

According to Table 8.1, PT1 did not always give 'important/ specific feedback' in the pre-test; however, PT1 always gave such feedback in the post-test. PT1 slightly improved in giving 'important/specific feedback'. This result corresponds with PT1's log-file results between with and without scaffolding.

In the pretest, PT1's pattern for giving 'positive feedback' started from giving the 'positive feedback' followed by the error messages but in the post-test PT1's giving of such feedback was changed so that the error messages were in between two instances of 'positive feedback' according to McFeSPA's principle. PT1 preferred not to use "well done" if the student made a long list of mistakes.

According to the pre- and post-test results, PT1 improved in giving appropriate 'positive feedback'. This corresponds with PT1's log-file results both with and without scaffolding. i.e. PT1 obtained the benefit for the scaffolding from the system. ("*...I don't know I learned so much and did what it told me to do.*")

Overall, PT1's greatest improvements were in giving good feedback and reducing the number of mistakes made while using the system without scaffolding. This is true because PT1's log-file results for both with and without scaffolding generally correspond with PT1's pre- and post-test results. Results from PT1's questionnaire shows that overall, PT1 agreed that it was easy to learn how to use McFeSPA and was satisfied with McFeSPA but PT1 felt frustrated - especially when PT1 was told what mistakes PT1 had made.

In learning to give feedback with McFeSPA, PT1 thought that the citation in the Glossary interface of McFeSPA was quite useful and helped PT1 access information about feedback skill. PT1 thought this could help PT1 increase knowledge/understanding of the issue of learning to give feedback. In addition, PT1 considered the system would encourage further thought about the effect of different forms of feedback and exploration of the different skills that the system tried to teach ("*...I don't know I learned so much and did what it told me to do.*"). PT1 felt it strange that the system reported PT1 was doing the wrong thing, but PT1 agreed and saw the benefits of the system's suggestion. PT1 thought McFeSPA could help PT1 reflect/rethink PT1's skill in giving feedback because PT1 also felt the system explored different approaches and gained benefit from the different approaches. PT1 agreed that McFeSPA could help PT1 to give quality feedback to the students ("*probably more than writing on a piece of paper. I can structure and give some more details*"). PT1 thought using McFeSPA helped PT1 learn ("*I learn the way that the errors are enclosed by positive feedback motivating feedback is very important, "sandwiched"*"). PT1 pointed out that it might be complicated to apply the knowledge PT1 obtained by using the system to mark any programming assignment (not only Prolog) but it could be done in theory. In McFeSPA, PT1 liked the automated analysis tool that automated find the error and give the location point ("*That is very good*").

8.8.2 Participant 2(PT2)'s results

PT2 had never been trained to give feedback but had some experience in giving feedback on programming assignments through learning by oneself while PT2 taught Prolog programming in Malaysia. Currently, PT2 is a TA in the School of Informatics at Edinburgh University and marks Java programming assignments according to the tutor's format guide.

Table 8.2 PT2's percentages of successes for each type of feedback (between using McFeSPA with and without scaffolding and between pre- and post-test), refuse help and accept help of each contingent hint for each type of feedback.

Issue	Type of feedback			
	'Feedback loop' and 'Individual feedback'	**'Detailed/ Elaborative feedback'**	**'Important/ Specific feedback'**	**'Positive feedback'**
% of **successes** in using McFeSPA **with** scaffolding	69.23	77.78	71.43	53.83
% of **successes** in using McFeSPA **without** scaffolding	100	100	100	71.43
% of **'accept help'** in using McFeSPA with scaffolding	50	50	100	57.14
% of **'refuse help'** in using McFeSPA with scaffolding	50	50	0	42.86
% of **successes** in **pre**-test paper	83.33	100	66.67	100
% of **successes** in **post**-test paper	100	100	100	100

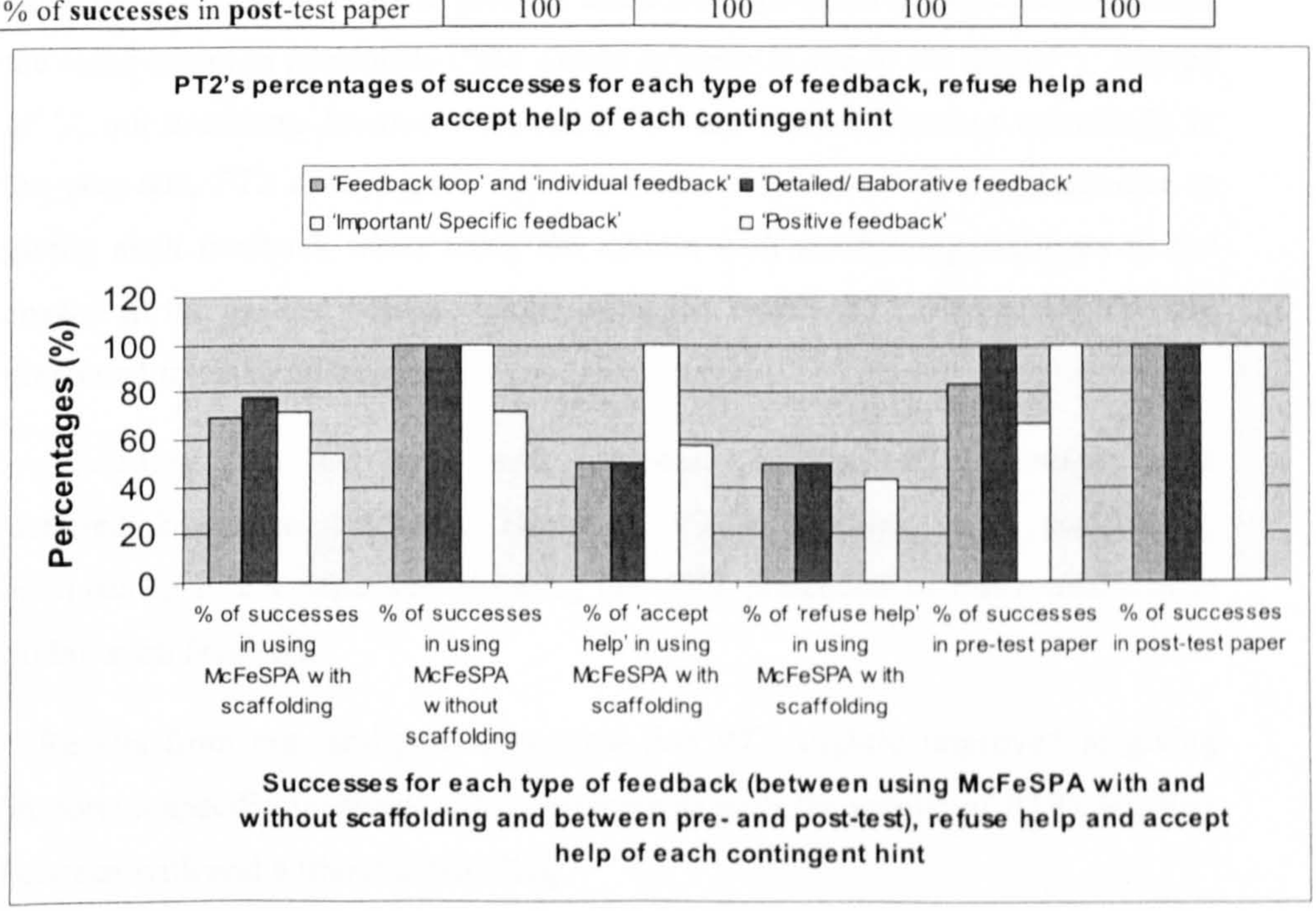


Figure 8.4 PT2' percentages of successes for each type of feedback, refuse help, and accept help of each contingent hint

According to table 8.2 and Figure 8.4, in using the system for marking students' script, PT2 improved giving 'Feedback loop and Individual feedback', 'Detailed/Elaborative feedback, 'Important/Specific feedback', and 'Positive feedback' 30.77%, and 28.57%, 17.60% respectively. The number of PT2's 'accept help' and 'refuse help' is similar in both 'Feedback loop' and 'Individual feedback', and 'Detailed/Elaborative feedback' with 50%. PT2 always accepted help for giving 'Important/Specific feedback' while in giving 'Positive feedback', PT2 did both accept and refuse help but accepted more than refused 14.28%. In the paper test, PT2 always gave 'Detailed/Elaborative feedback' and 'Positive feedback' and improved giving 'Feedback loop and Individual feedback', and 'Important/Specific feedback' with 16.67% and 33.33% respectively.

PT2's log-file results show that PT2 slightly improved giving a 'Feedback loop' and 'Individual feedback'. This result is interesting in comparison with PT2's pre- and post-test results. In the pre-test, PT2 mostly gave 'Feedback loop' and 'Individual feedback' and pointed out that it is an encouraging message to students. PT2 did not agree to give the same message again if the student repeated the same errors as previously ("*the source of error is due to the use of ';' instead of ',', not necessary because the student has learnt from previous mistake.*"). In the post-test, PT2 always gave such feedback. However, PT2's performance in giving such feedback while using the system with scaffolding contrasts to the answer in the pre-test because whilst using the system PT2 tried to explore and pretended to make mistakes.

According to the pre- and post-test results, PT2 always gave 'detailed/elaborative feedback. However, PT2's log-files, with scaffolding, contrasts to PT2's paper-test showing that PT2 pretended to make mistakes in giving such feedback.

Results from pre- and post-tests show that PT2 slightly improved in giving 'important/specific feedback'. This corresponds with the results of PT2's log-files between with and without scaffolding.

According to the pre- and post-test results, PT2 always demonstrated an appropriate pattern for giving 'positive feedback' i.e. the error messages placed between two 'positive feedback' messages. PT2's log-files, with scaffolding, shows that PT2 pretended to make a mistake in giving such feedback. In using McFeSPA without scaffolding, PT2 made a small mistake regarding the position of 'positive feedback'.

According to Table 8.2, overall, comparing between with and without scaffolding, PT2 improved giving feedback. This is likely because the results of PT2's log-files for both with and without scaffolding correspond with PT2's pre- and post-test results. According to PT2's log-files, PT2 tried to explore and pretended to make mistakes while using the system with scaffolding because PT2's performance contrasts with the answer in the pre-test. Results from PT2's questionnaire show that overall, PT2 agreed that it was easy to learn how to use McFeSPA and PT2 was satisfied with McFeSPA.

In learning to give feedback with McFeSPA, PT2 thought McFeSPA can help PT2 increase knowledge/understanding in the issue of learning to give feedback because it was systematic and provided more structure ("*...I realise that maybe I have been thinking that I give you more structure and organise the way of marking; ... it is a good way of presenting of the feedback because it structure and organise...; ... it taught me how to be simple and straight forward about my feedback...and not analyse too much sometime; you had like design/implement/style...Yeah, it's systematic. I think*"). In addition, PT2 thought McFeSPA helped PT2 reflect/rethink PT2's skill in giving feedback in terms of its automatics. PT2 felt it is very good and thought it could help PT2 mark quickly ("*...because it is so much quicker, and I think it is very good*"). PT2 said that the help from the system assisted PT2 to reflect/rethink PT2's performance in giving feedback ("*... I think it would help me more clear and precise about my feedback like, yeah, and break down every error, you know*"). PT2 thought McFeSPA could help PT2 give quality feedback to students but the quality in terms of time ("*...it does the analyse I think it's great... I don't do it myself then quality because of time. I think I save a lot of times...Yeah, I think the system is good because I never use anything like this. I always just use hand marking... I'm using which is good so it's fantastic*"). PT2 thought PT2 learnt using McFeSPA in an automated way

and it was useful for marking and giving feedback to students' in small assignments ("*I think they are fantastic...I think this system is excellence for marking...*"). In addition, PT2 thought that the fact that the system can summarise the student's errors was very good. PT2 said that PT2 could learn McFeSPA without any assistance from the system because PT2 has some familiarity with it. PT2 thought that PT2 could apply the knowledge PT2 obtained by using the system to mark any programming assignment (not only Prolog) because most programming language has some structure, automation and organisation. PT2 said that PT2 liked McFeSPA in that it worked quickly and it was easy to use ("*It's very quick and also it's easy to use in general...You don't read different messages all the times and I think the uniformity, it's easy to use*"). PT2 did not like the feature while using McFeSPA with scaffolding when it popped up a message and informed PT2 to give the appropriate pattern for giving 'positive feedback' because PT2 intended to use PT2's own style. Regarding taking into account history of student's error, PT2 also suggested that if the students submitted different assignments, comparison of the numbers of previous and current student's errors did not reflect anything. PT2 suggested that the number of student's mistakes should be presented in terms of percentages. However, PT2 pointed out that this comparison of student's errors might be good with the 1st simple assignment for students.

8.8.3 Participant 3 (PT3)'s results

Before becoming a teaching assistant PT3 received about half a days training in giving feedback so he does have some previous knowledge in this area. PT3 pointed out that he was taught to give feedback, about how to be supportive, and how to criticise students' work constructively and the general guidelines of giving feedback. PT3 has some experience in marking C programming assignments.

Table 8.3 PT3's percentages of successes for each type of feedback (between using McFeSPA with and without scaffolding and between pre- and post-test), refuse help and accept help of each contingent hint for each type of feedback.

Issue	Type of feedback			
	'Feedback loop' and 'Individual feedback'	**'Detailed/ Elaborative feedback'**	**'Important/ Specific feedback'**	**'Positive feedback'**
% of **successes** in using McFeSPA **with** scaffolding	77.78	100	100	100
% of **successes** in using McFeSPA **without** scaffolding	83.33	100	100	100
% of **'accept help'** in using McFeSPA with scaffolding	25	0	0	0
% of **'refuse help'** in using McFeSPA with scaffolding	75	0	0	0
% of **successes** in **pre**-test paper	33.33	66.67	100	100
% of **successes** in **post**-test paper	50	66.67	100	100

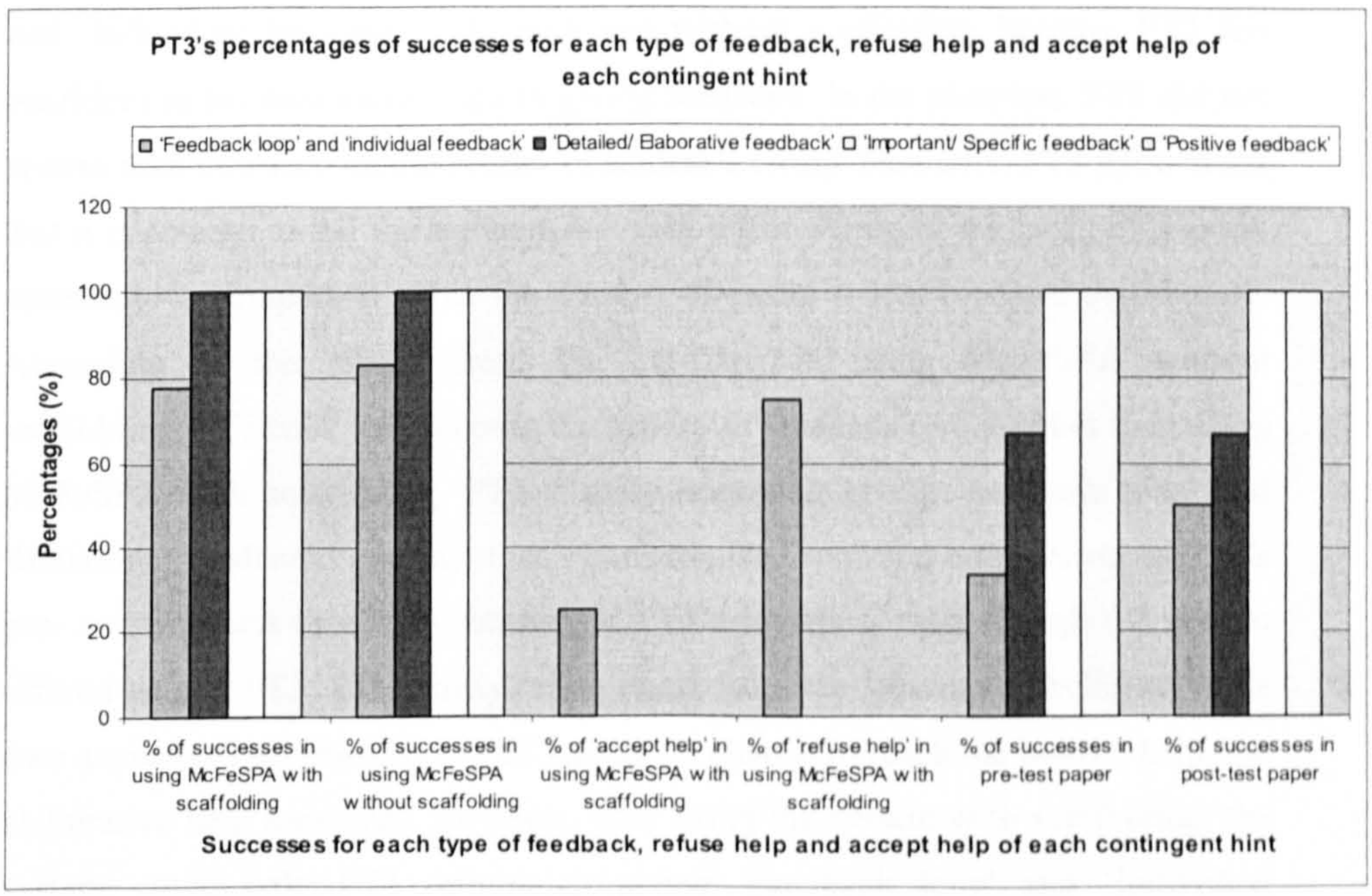


Figure 8.5 PT2' percentages of successes for each type of feedback, refuse help, and accept help of each contingent hint

According to Table 8.3 and Figure 8.5, in marking students with McFeSPA, PT3 always gave 'Detailed/Elaborative feedback', 'Important/Specific feedback', and 'Positive feedback'. In giving Feedback loop and Individual feedback', PT3 improved 5.55% and refused help more than accepted 50%. In the paper test, PT3 always gave 'Important/Specific feedback', and 'Positive

feedback' and increased giving 'Feedback loop' and 'Individual feedback' 16.67%.

According to PT3's perspective in giving 'feedback loop' and 'Individual feedback' from the pre- and post-test, PT3 mostly took into account the history of student's errors for both pre- and post-test except for the common error (e.g. missing parenthesis in PT3's perspective). PT3 did not take into account the history of the student's simple errors because PT3 believed in PT3's answer and commented that (*"It is enough to point out the error. Not compare if there are less (or) more times than before."*). This demonstrates that PT3's explanation of feedback depends on the importance of the error, not the number of errors. Results from PT3's log-files show that PT3 made errors in giving 'feedback loop' and 'Individual feedback' both with and without scaffolding because PT3 felt confident in his own knowledge in giving feedback. In the post-test, PT3 did not always take into account the history of student's errors. However, PT3 pointed out that it is enough to tell the students that they did it wrong in the past but it is not necessary to compare whether the number of errors is less or more than before. According to the results from the log-files, in using McFeSPA without scaffolding, PT3 took into account the history of student's errors rather than using McFeSPA with scaffolding. PT3 slightly improved giving 'feedback loop' and 'Individual feedback' and this result corresponded with the comparison of PT3's pre- and post-test results. According to PT3's log-files, even though the system offered help to PT3, PT3 mostly refused help from the system and believed in his own answers. This resulted in PT3 having no chance to reach the bottom help, the elaborative help messages. However, after using the system with scaffolding, the system could help PT3 rethink in giving 'Feedback loop' and 'Individual feedback'. This resulted in PT3 always taking into account the history of student's answer while using the system without scaffolding i.e. PT3 slightly improved PT3's skill in giving such feedback. In sum, PT3's giving 'Feedback loop' and 'Individual feedback' corresponds with PT3's results from both pre- and post-test.

According to the pre- and post-test results, PT3 did not always give 'detailed/elaborative feedback. PT3 provided explanation to the important errors according to PT3's perspective. However, PT3 did not give explanation for common errors. PT3 thought students often overlook parenthesis. Regarding the

error of unreachable goal, PT3 pointed out that giving the line number should suffice. According to the log-files these results did not correspond with the comparison of PT3's results in giving feedback with and without scaffolding. This could be because PT3 acknowledges learning to give feedback while using the system is required, but still believes that giving brief detailed errors to the common errors is better than an explanation of such errors.

Pre- and post-test results show that PT3 always gave 'important/specific feedback', and this result corresponds with the comparison of PT3's result of log-files between with and without scaffolding.

PT3's pattern for giving 'positive feedback' was the same in both pre- and post-test. That is the error messages in between two 'positive feedback' messages. According to the pre- and post-test results, PT3 always gave appropriate 'positive feedback' and this result corresponds with the comparison of PT3's result of log-files between with and without scaffolding.

Overall, to compare between with and without scaffolding, PT3 slightly improved giving feedback. This is true because the results of PT3's log-files for both with and without scaffolding correspond with PT3's pre- and post-test results. PT3 thought PT3 used McFeSPA in an hour so PT3 thought it is hard to change PT3's behavior in giving feedback. According to PT3's answers in the interview section, PT3 is still familiar with the previous style of giving feedback. Also, PT3 said that PT3's tutor who asked PT3 to mark student's assignment did not allow PT3 to write any comments, just find the mistake. Thus, PT3 thought learning to give feedback with McFeSPA in an hour didn't help PT3 change PT3's behavior at all. In addition, results from PT3's questionnaire show that overall, PT3 agreed that it was easy to learn how to use McFeSPA and was satisfied with McFeSPA.

In learning to give feedback with McFeSPA, PT3 thought McFeSPA helped PT3 increase PT3's knowledge/understanding in learning to give feedback a bit because PT3 was familiar with giving feedback in the normal way ("*because I am used to giving feedback in the normal way, in the standard way, so it's hard to change in an hour. It happens like this for two years. I am quite used to giving feedback in the same way*") PT3 did not learn much from McFeSPA because PT3

mostly refused the help offered by McFeSPA and did not choose the most appropriate answer (according to the system). PT3 was also confident about PT3's own knowledge about marking. PT3 thought McFeSPA helped PT3 reflect/rethink PT3's skills in giving feedback because McFeSPA was more structured and it helped PT3 to see the mistakes i.e. PT3 could see what feedback PT3 wanted to give (*"it's like an assembly change, when I was doing it manually it tended to be quite random, but this system provides more structure"*). PT3 agreed that McFeSPA can help PT3 give quality feedback to students (*"I think so because it allows you to add more errors something else. It can help, yeah"*). By using McFeSPA, PT3 thought PT3 learnt about using student's previous mistake in giving feedback. (*"I think I learnt different way of giving feedback ... in McFeSPA maybe the previous mistakes of student can also be used in giving feedback. I never used that kind of thing"*). PT3 could use McFeSPA without any assistance from the system but PT3 would need some form of training to use it properly. PT3 thought PT3 could apply the knowledge PT3 obtained by using the system to mark any programming assignment (not only Prolog). PT3 liked McFeSPA because it allowed PT3 to increase his own knowledge of feedback giving. (*"Yeah, I can extend my own knowledge. I can get the messages you can see"*).

8.8.4 Participant 4 (PT4)'s results

PT4 has never been trained to give feedback but is self taught. PT4 has some experience in giving feedback over a period of several years.

Table 8.4 PT4's percentages of successes for each type of feedback (by using McFeSPA without scaffolding and between pre- and post-test).

Issue	Type of feedback			
	'Feedback loop' and 'Individual feedback'	'Detailed/ Elaborative feedback'	'Important/ Specific feedback'	'Positive feedback'
% of successes in using McFeSPA without scaffolding	0	100	0	0
% of successes in pre-test paper	0	100	0	0
% of successes in post-test paper	0	100	0	0

According to PT4's perspective on giving 'feedback loop' from the pre- and post-test, PT4 did not take into account the history of the student's error because

PT4 pointed out that the given feedback messages had enough details to help students. These results correspond with PT4's log-files.

PT4 always gave 'detailed/elaborative feedback' according to both pre- and post-test and log-files.

According to the results of pre- and post-test, and log-files, PT4 always gave 'detailed/elaborative feedback' (but did not give 'important/specific feedback') even though there were several mistakes with the same error types in the student's script.

Regarding the use of an appropriate pattern for giving 'positive feedback' which is the error messages in between two messages with 'positive feedback', PT4's pattern for giving 'positive feedback' for both pre- and post-test and in log-files are the same - that is, giving feedback starting with 'positive feedback' and followed by the error messages.

Overall, PT4's log-file results correspond with PT4's pre- and post-test results i.e. PT4 always gave 'detailed/elaborative feedback' in the feedback report. It is true that PT4 did not improve learning in giving feedback from McFeSPA because PT4 did not use McFeSPA with scaffolding. In addition, results from PT4's questionnaire showed that overall, PT4 agreed that it was easy to use McFeSPA in giving feedback and was satisfied with McFeSPA.

In learning to give feedback with McFeSPA, PT4 thought McFeSPA helped PT4 increase PT4's knowledge/understanding about the issue of learning to give feedback because PT4 thought McFeSPA gave PT4 more examples of feedback. PT4 felt McFeSPA helped PT4 reflect/rethink PT4's skills in giving feedback because it gives PT4 the idea that the feedback should be of more than one kind. However, PT4 doesn't know how to give feedback in each situation because PT4 used the system without scaffolding. PT4 agreed that McFeSPA could help TAs give quality feedback to students. In using McFeSPA without scaffolding, PT4 thought PT4 learned to provide more kinds of feedback. PT4 thought PT4 can apply the knowledge PT4 obtained by using the system to mark any programming assignment (not only Prolog). PT4 liked McFeSPA in that it automatically generated a report and provided an adaptable feedback message. PT4 disliked

McFeSPA in that sometimes, it showed many dialogues. For example, when PT4 did a repeat action on McFeSPA, it goes through exactly the same process.

8.8.5 Participant 5 (PT5)'s results

PT5 had a lot of experience in teaching and marking both Prolog and Functional programming language assignments over a three year period. PT5 had some training in giving feedback because PT5 has been a teaching assistant in the School of Informatics at Edinburgh University.

PT5 thought PT5 gave oral feedback to students quite a lot while teaching but not much in written form.

Table 8.5 PT5's percentages of successes for each type of feedback (by using McFeSPA without scaffolding and between pre- and post-test).

Issue	Type of feedback			
	'Feedback loop' and 'Individual feedback'	'Detailed/ Elaborative feedback'	'Important/ Specific feedback'	'Positive feedback'
% of successes in using McFeSPA without scaffolding	0	100	100	0
% of successes in pre-test paper	100	83.33	100	0
% of successes in post-test paper	100	100	100	0

According to PT5's perspective on giving 'feedback loop' and 'Individual feedback' from the pre- and post-test, PT5 always considered the previous student's error; however, giving such feedback according to PT5's log-files contrasted between pre- and post-test because PT5 needed to know the previous student's code or previous version of the student's solution. In addition, PT5 argued that giving a comparison with the student's previous and current errors should be generated automatically or provided by the system, not manually. Nevertheless, we required the user of McFeSPA to know the student's errors for both previous and current error because we needed he/she to consider the history of student's errors before generating feedback messages.

According to the pre- and post-test results, PT5 always gave 'detailed/elaborative feedback'. This result corresponds with PT5's log-files in giving such feedback.

Pre- and post-test results showed that PT5 always gave 'important/specific feedback' if the same types of errors were repeated in a different line number/position. PT5 pointed out that one could explain the error without being too verbose. This result corresponds with PT5's log-files in giving such feedback.

PT5's pattern for giving 'positive feedback' for both pre- and post-test is the same – that is, start with 'positive feedback' and follow with error messages. According to PT5's perspective, PT5 gave 'positive feedback' according to student's errors. If the students made no or few errors, PT5 might just give 'positive feedback'.

Overall, PT5's log-file results about giving 'detailed/elaborative feedback' and 'important feedback' correspond with PT5's pre- and post-test results i.e. PT5 always gave 'detailed/elaborative feedback' and 'important/specific feedback' to the student. PT5 did not take into account the history of the student's error because PT5 required this function to be done automatically. In addition, results from PT5's questionnaire showed that overall, PT5 agreed that it was easy to use McFeSPA in giving feedback and was satisfied with McFeSPA.

In learning to give feedback with McFeSPA, PT5 thought PT5 did not realise that McFeSPA helped PT5 increase knowledge/understanding in the issue of learning to give feedback but PT5 was interested in the process of considering what feedback to give ("*I didn't realise it helped me to learn to give feedback. It's kind of interesting to just think about what the process is and for those reasons so the task of considering what feedback to give is kind of interesting.*"). PT5 thought McFeSPA helped PT5 reflect/rethink PT5's skill in giving feedback ("*...in this sense, measurement has done a very good job, but it was nonetheless interesting to see that and good to see that*"). PT5 did not agree that McFeSPA can help PT5 to give quality feedback to students because PT5 thought the marker needed to know and understand the errors before writing the feedback and because PT5 thought McFeSPA did not help this take place and felt that McFeSPA did not give all kinds of error types or analyse all errors because it was developed for experimental purposes and as such only represented some kinds of errors. PT5 thought PT5 learnt giving separation of the different section of error types by using McFeSPA ("*I like the idea of the separating of the different sections of*

design, implementation, style... they give a summary of that. That's quite nice. I like that"). PT5 pointed out that McFeSPA made PT5 realise that 'positive feedback' isn't about writing things ("*It sounds nice but it is about when the students had done well in one of those sections, design... Implementation but they had done nicely... make me realise that notion of positive feedback with McFeSPA is clearly not what I want but I didn't really think about that before, so that sounds good*"). Even though PT5 did not know what PT5 learnt from using McFeSPA, PT5 thought PT5 could apply the idea obtained by using McFeSPA to mark other programming language (not only Prolog). PT5 used McFeSPA without scaffolding so McFeSPA did not help PT5 learn anything. PT5 saw some ideas while using the system. PT5 liked McFeSPA in that it provided ideas to help student and tutor marking ("*I like the idea. The idea of having a tutor that helps student marking and the system that helps tutor marking. I think that's great. This is nice to be compared between the students...*"). PT5 is experienced at teaching and marking Prolog programming assignments. Thus, it is true that PT5 wanted to mark an assignment in a realistic way (in terms of usability). PT5 didn't need to learn to give feedback (in terms of learnability) but PT5 needed the system to produce a feedback report quickly or automatically ("*I seems it should automatically generate the report, no clicking and then you can add it yourself. That would be much simpler and much better I think*"). PT5 didn't think that learning to give feedback with McFeSPA could help people learn but PT5 thought it would be good if learning with experienced tutor in giving feedback ("*I think UMM having some experience and working with someone else usually or see other people's answer. That will make it help to learn... having UMM someone else to generate the report and then say I think is a good report, and then compare your report with that report. That'll help you to improve your ability to give feedback on a student.*"). Nevertheless, PT5 felt frustrated when the system asked many questions ("*it asks me a lot of questions, makes me frustrated with it and I don't want that. I want to go through as quick as possible, so I would learn to click the button very quickly without reading any thing*").

To sum up, PT5 mostly agreed with the idea of giving feedback by McFeSPA but disagreed with the implementation of McFeSPA in giving 'Feedback loop' and 'Individual feedback'.

8.8.6 Participant 6 (PT6)'s results

PT6 had some training in the area of giving feedback when studying for the Post Graduate Certificate in Education (PGCE), and has also had quite a lot of further practical experience in giving feedback. PT6 also has teaching experience in secondary schools, sixth form colleges and tutoring at the introductory university level. The latter includes some, but not much in Prolog teaching. PT6 pointed out knowing the students personally helps the feedback giver in giving feedback to them because students respond differently to the feedback they receive. Also giving feedback depends on the errors, what the student was taught and the timing in giving detailed feedback.

Table 8.6 PT6's percentages of successes for each type of feedback (by using McFeSPA without scaffolding and between pre- and post-test).

Issue	Type of feedback			
	'Feedback loop' and 'Individual feedback'	'Detailed/ Elaborative feedback'	'Important/ Specific feedback'	'Positive feedback'
% of successes in using McFeSPA without scaffolding	90.91	84.62	80	28.57
% of successes in pre-test paper	100	100	33.33	100
% of successes in post-test paper	83.33	66.67	33.33	0

According to PT6's perspective in giving 'Feedback loop' and 'Individual feedback' from the pre-test, PT6 always considered the history of the student's errors and almost always gave such feedback in the post-test. These results correspond with PT6's log-files. PT6 thought that students will remember feedback if they spotted the errors themselves. PT6 also pointed out that feedback depends on the student's level of performance

According to the pre-test results, PT6 always gave 'detailed/elaborative feedback'. However, after using McFeSPA, PT6 did not always give such feedback in the post-test because PT6 felt giving 'detailed feedback' depends on the history of students' errors. According to PT6's log-files PT6 will not give 'detailed feedback' to students if they have made the same kinds of errors before. However, PT6 sometimes gave 'detailed feedback' to students and thought that if

the students had made the same errors before then they should be given 'detailed feedback' to help them learn more about the errors.

Pre- and post-test results showed that PT6 rarely gave 'important/specific feedback' i.e. PT6 sometimes gave 'detailed feedback' if the same types of errors are repeated in a different line number/position. This result corresponded with PT6's log-files in giving such feedback.

PT6's pattern for giving 'positive feedback' in the pre-test shows the error messages inserted in between two 'positive feedback' messages. After using McFeSPA, PT6's giving of such feedback in the post-test started with 'positive feedback' and is then followed by the error messages. According to PT6's perspective, PT6 gave 'positive feedback' according to student's errors i.e. if the students made no or few errors, PT6 might give the error messages in between two 'positive feedback' and if the students made more errors, then give less 'positive feedback'.

Overall, PT6's log-file results in giving feedback mostly correspond with the post-test result i.e. after using McFeSPA, PT6 changed PT6's perspective in giving feedback. Mostly, PT6 gave all the kinds of feedback provided by McFeSPA depending on the history of student's errors. In addition, results from PT6's questionnaire showed that overall, PT6 agreed that it was easy to use McFeSPA in giving feedback and was satisfied with McFeSPA.

In learning to give feedback with McFeSPA, PT6 thought McFeSPA helped PT6 increase knowledge/understanding in the issue of learning to give feedback because it made PT6 consider giving 'detailed feedback' depending on the student's previous errors and the current errors the student made. (*"...You want more 'detailed feedback' or assuming you have more 'detailed feedback' at first time and then give less 'detailed feedback'...if they made same errors you may give them 'detailed feedback'"*). PT6 thought McFeSPA helped PT6 reflect/rethink on the various kinds of feedback and which type of errors that the students made and that this encouraged PT6 to give more 'detailed feedback' even though the student made common errors (e.g. missing indentation). PT6 also agreed that McFeSPA could help PT6 to give quality feedback to students. By using McFeSPA, PT6 thought PT6 learnt about what to write in the feedback and

how much detail should be provide. PT6 thought PT6 could apply the knowledge PT6 obtained by using the system to mark any programming assignment (not only Prolog). PT6 liked McFeSPA as it was very simple to use, easy to learn how to use and it helped the markers to think about the way in which they mark. PT6 disliked McFeSPA in relation to the skill meter because PT6 did not know the meaning of each feedback in each line. However, PT6 didn't know that the 'glossary' button in the 'skill meter' interface provided the explanation of each kind of feedback skill in the 'skill meter' interface.

8.9 Analysis of the results

In the previous section, we provided the results for each participant from evaluation of the McFeSPA learning environment. In this section we present the analysis of the results obtained by analyzing the pre- and post-test, log-files, observation checklist, questionnaire, and structured interviews.

1) Is the basic idea of McFeSPA helpful?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Strongly agree	Agree	Strongly agree	Strongly agree	Strongly agree	Strongly agree

All participants agreed that the basic idea of McFeSPA is helpful. PT1 stated that McFeSPA was a good tool to speed the marking task and agreed that the scaffolding was useful. PT6 also agreed that McFeSPA was a useful tool to help PT6 evaluate what feedback PT6 would like to give.

2) What did the participants learn by using McFeSPA?

PT1 thought PT1 learnt how to surrounded errors by 'positive feedback' and felt that motivating the student is very important i.e. PT1 learnt to give appropriate 'positive feedback'. PT2 thought PT2 learned about using McFeSPA in marking programming assignments automatically and giving feedback for small assignments. PT2 also pointed out that McFeSPA is a good tool because it can summarise student's errors. PT2 learnt to give feedback in general and the way to summarise feedback to students i.e. PT2 learnt to give important feedback to students. PT3 thought PT3 learnt about using student's previous mistakes in giving feedback i.e. PT3 learnt to give 'Feedback loop' and 'Individual feedback'

PT4 thought PT4 gave several kinds of feedback. PT5 thought PT5 learnt to separate different sections of feedback (design/implementation/style errors), to summarise these errors, and to give 'positive feedback' that related to the errors that the student made. PT6 thought PT6 learnt about what to write in the feedback report and how much details were required.

3) How do the participants learn to provide each type of feedback according to McFeSPA?

3.1) How do the participants learn to provide 'Feedback loop' and 'Individual feedback' messages?

Table 8.7 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refuse help and accept help of each contingent hint in giving 'Feedback Loop' and 'Individual feedback'

Giving 'feedback loop' and 'Individual feedback'	Experiment group (EG)			Comparison group (CG)		
	PT1	PT2	PT3	PT4	PT5	PT6
% of all successes in giving 'Feedback Loop' and 'Individual feedback' (with scaffolding)	100	69.23	77.78	-	-	-
% of all successes in giving 'Feedback Loop' and 'Individual feedback' (without scaffolding)	75	100	83.33	0	0	90.91
% of all 'refuse help' in giving 'Feedback Loop' and 'Individual feedback' (with scaffolding)	0	50	75	-	-	-
% of all 'accept help' in giving 'Feedback Loop' and 'Individual feedback' (with scaffolding)	0	50	25	-	-	-
% of all successes in giving 'Feedback Loop' and 'Individual feedback' (Pre-test)	66.67	83.33	33.33	0	100	100
% of all successes in giving 'Feedback Loop' and 'Individual feedback' (Post-test)	66.67	100	50	0	100	83.33

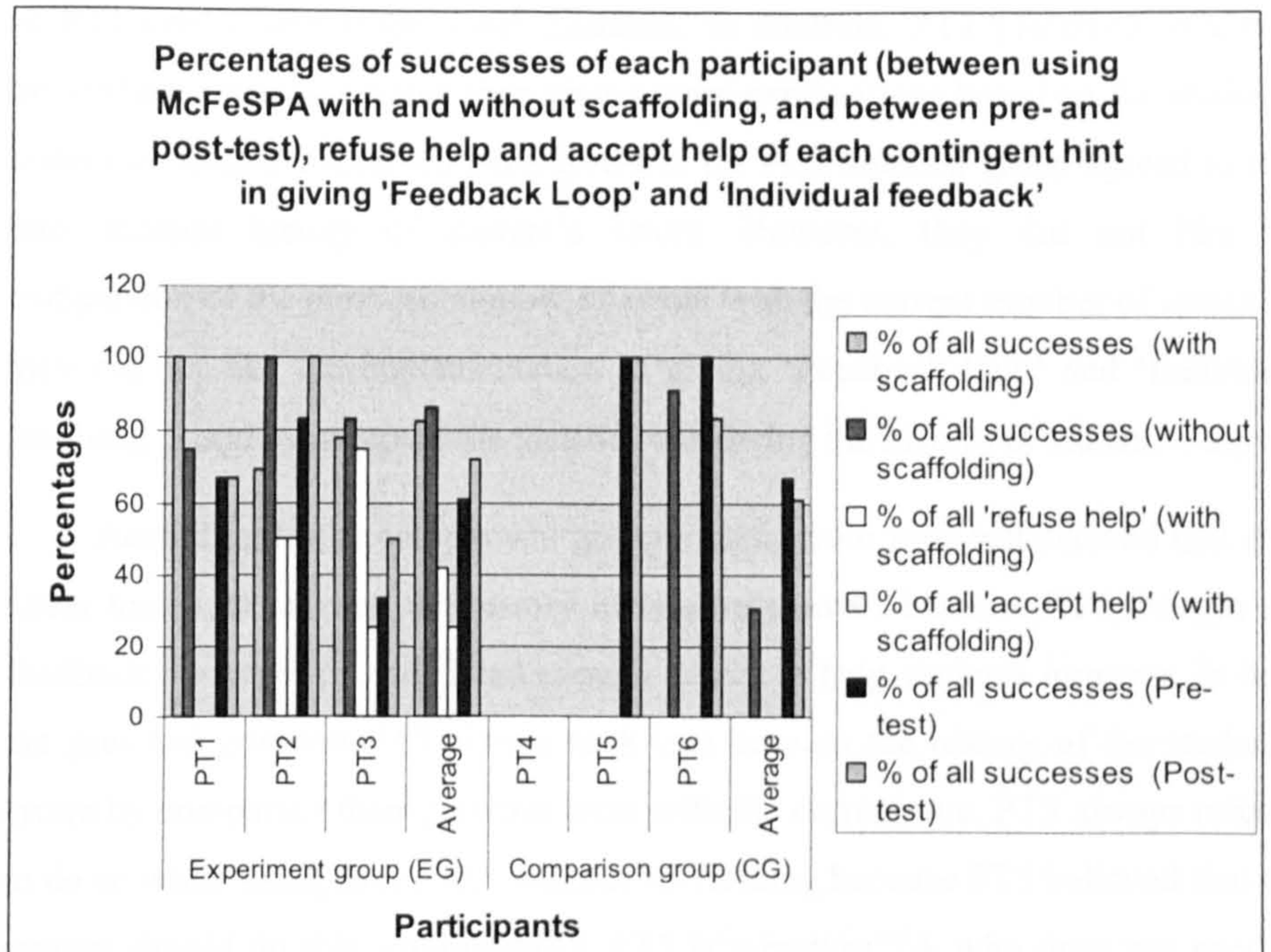


Figure 8.6 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refuse help and accept help of each contingent hint in giving 'Feedback Loop' and 'Individual feedback'

According to the pre- and post-test results in Table 8.7 and Figure 8.6, most participants in the experimental group improved on giving 'Feedback loop' and 'Individual feedback' except for PT1. It is probable that the experimental group improved their giving 'Feedback loop' and 'Individual feedback' because they are mature TAs and had experience in giving feedback. These results correspond with the comparison of giving such feedback between using McFeSPA with scaffolding and without scaffolding. It could be the case that PT1 did not like phrasing such feedback using McFeSPA. PT1's log-files showed that PT1 refused to take into account history of student's errors four times. This resulted in no count for giving 'feedback loop' and 'Individual feedback' skill. If PT1 took into account the history of student's errors, PT1 might make some mistakes while using the system with scaffolding. We deduce this from the result of PT1's log-files without scaffolding. Likewise, according to the observation checklist, PT1 hurried to mark all scripts without scaffolding and spent less time than previously

so PT1 could have made some mistakes. In addition, PT1 preferred to give a motivating comments rather than giving more explanations based on the student's history of simple errors. All participants in the experimental group agreed to take into account history of student's errors. However, they did not like the comparison of the previous number of errors with the current number of errors i.e. they did not like the implementation of giving 'Feedback loop' and 'Individual feedback' but they accepted the idea of considering the history of student's errors.

According to the comparison group, results from post-test showed that PT4 never took into account the history of student's errors because PT4 thought the feedback messages provided had enough details to help students improve. In both the pre- and post-test PT5 always took into account the history of the student's errors by comparing their previous error with the current one. PT5 always refused to do so while using McFeSPA without scaffolding because PT5 believed that the system should do this automatically. PT5 is a mature TA who does not need to learn to give feedback from the system but needs to finish marking quickly. PT6 took the history of student's errors into account most often, and used such information to consider the detail of feedback for each error type. Thus, most participants in the comparison group accepted the idea of giving 'Feedback loop' and 'Individual feedback' by taking into account the history of student's errors, except for PT5 who did not agree with the implementation of giving such feedback.

Although PT6's post-test in giving 'Feedback loop' and 'Individual feedback' reduced from the pretest, evidence from the log-files showed that PT6 did not intend to make mistakes in the post-test. In addition, all participants in comparison group are experienced in giving feedback. They mostly took into account the history of student's errors for both pre-and post-test. Most participants slightly improved giving such feedback because they had experience in giving feedback so they agreed that taking into account the history of student's errors is a good idea in giving feedback.

3.2) How do the participants learn to provide 'Detailed/Elaborative feedback' messages?

Table 8.8 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refusing help and accepting help of each contingent hint in giving 'Detailed/Elaborative feedback'

Giving 'Detailed/Elaborative feedback'	Experiment group (EG)			Comparison group (CG)		
	PT1	PT2	PT3	PT4	PT5	PT6
% of all successes in giving 'Detailed/Elaborative feedback' (with scaffolding)	86.67	77.78	100	-	-	-
% of all successes in giving 'Detailed/Elaborative feedback' (without scaffolding)	100	100	100	100	100	84.62
% of all 'refuse help' in giving 'Detailed/Elaborative feedback' (with scaffolding)	0	50	0	-	-	-
% of all 'accept help' in giving 'Detailed/Elaborative feedback' (with scaffolding)	0	50	0	-	-	-
% of all successes in giving 'Detailed/Elaborative Feedback' (Pre-test)	100	100	66.67	100	83.33	100
% of all successes in giving 'Detailed/Elaborative Feedback' (Post-test)	100	100	66.67	100	100	66.67

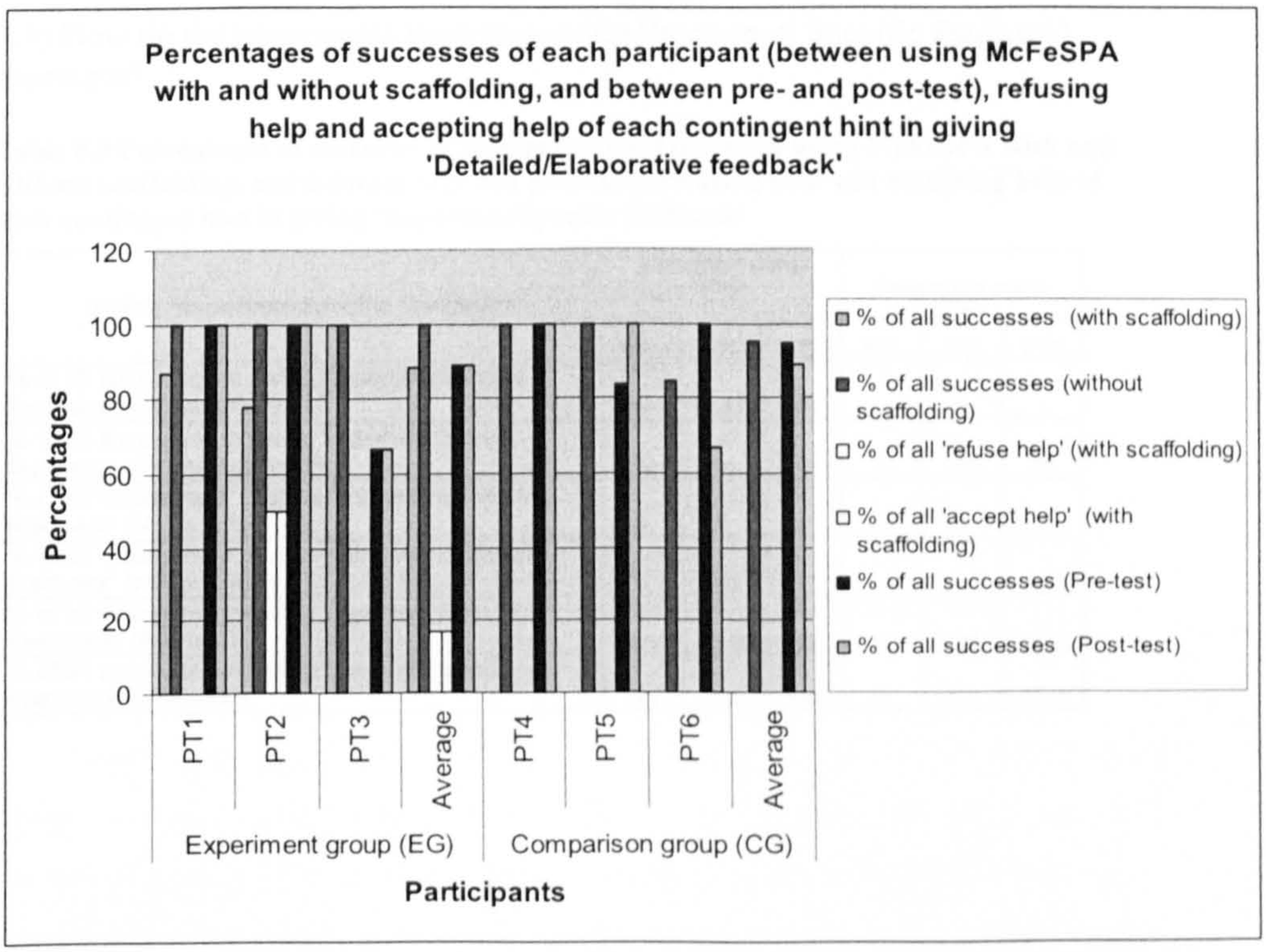


Figure 8.7 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refuse help and accept help of each contingent hint in giving 'Detailed/Elaborative feedback'

According to paper-test results in Table 8.8 and Figure 8.7, all participants gave mostly 'detailed/elaborative feedback' for both pre- and post-test. Even though PT2 occasionally accepted and occasionally refused the help offer in giving such feedback from the system, there is evidence that showed that PT2

tried to evaluate the system. This is believed since PT2 always gave such feedback for both paper-tests. In addition, PT3 did not always give such feedback because PT3 thought that it is not necessary to give 'detailed feedback' on the common errors, according to PT3's perspective. After using McFeSPA without scaffolding PT6 understood that it was not necessary to provide detailed feedback if the student has made the same errors as they made in the previous submission of the same assignment. PT6 was concerned with the history of student's errors. This could lead to reducing giving such feedback in the post-test paper in average of the comparison group.

3.3) How do the participants learn to provide 'Important/ Specific feedback' messages?

Table 8.9 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refusing help and accepting help of each contingent hint in giving 'Important/Specific feedback'

Giving 'Important/Specific feedback'	Experiment group (EG)			Comparison group (CG)		
	PT1	PT2	PT3	PT4	PT5	PT6
% of all successes in giving 'Important/Specific feedback' (with scaffolding)	75	71 43	100	-	-	-
% of all successes in giving 'Important/Specific feedback' (without scaffolding)	100	100	100	0	100	80
% of all 'refuse help' in giving 'Important/Specific feedback' (with scaffolding)	0	0	0	-	-	-
% of all 'accept help' in giving 'Important/Specific feedback' (with scaffolding)	0	100	0	-	-	-
% of all successes in giving 'Important/Specific Feedback' (Pre-test)	66 67	66 67	100	0	100	33.33
% of all successes in giving 'Important/Specific Feedback' (Post-test)	100	100	100	0	100	33.33

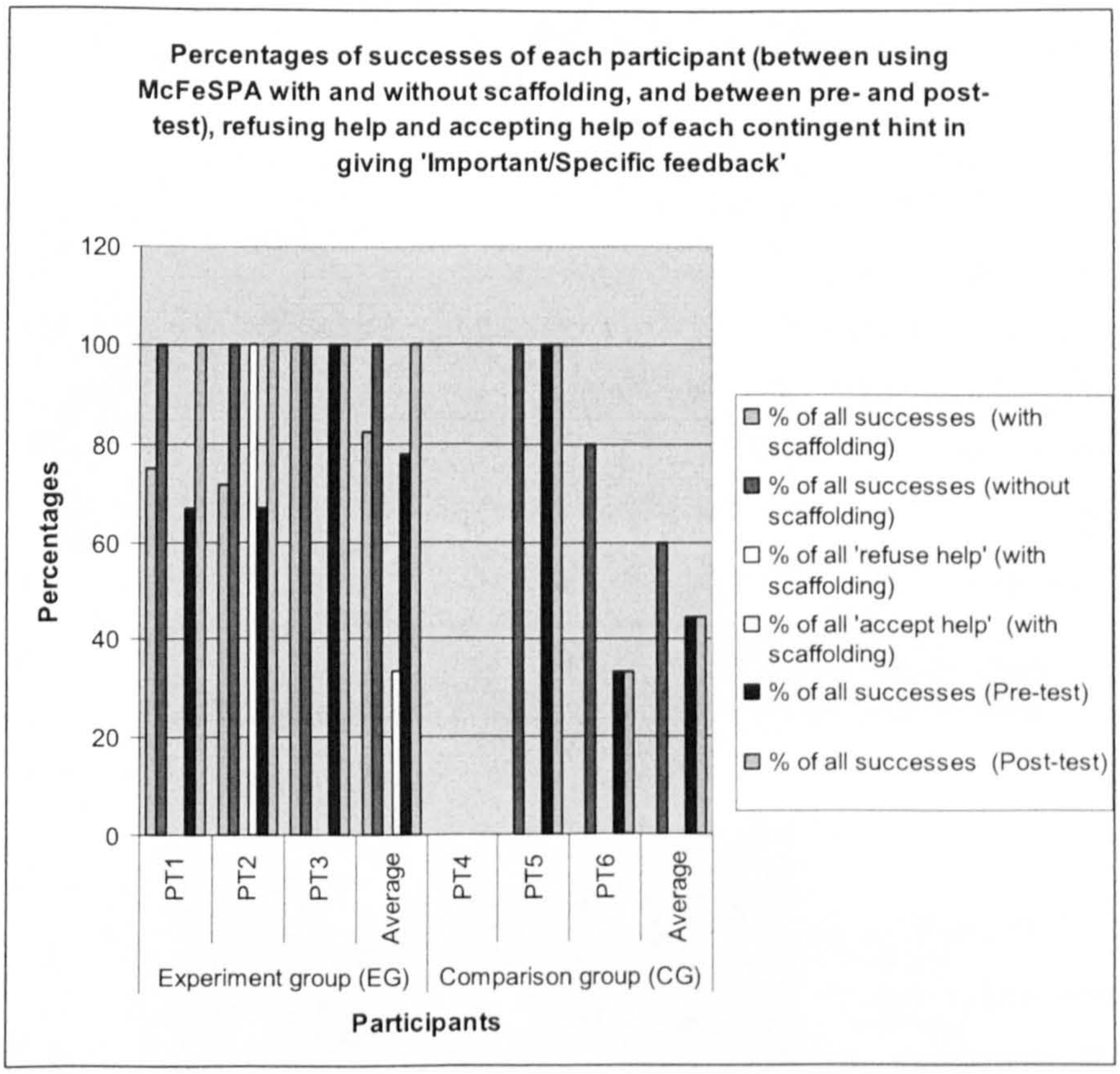


Figure 8.8 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refuse help and accept help of each contingent hint in giving 'Important/Specific feedback'

According to Table 8.9 and Figure 8.8, all participants in the experimental group provided 'important/specific feedback' in the post-test. PT1 and PT2 improved giving such feedback after using the system with scaffolding. These results correspond with post-test results. PT3, from the experimental group, always gave such feedback. In the comparison group, PT5 always gave such feedback while PT6 rarely gave such feedback because PT6 thought if the students have not made the errors before he/she should receive detailed feedback for all errors that appear even though they are the same types.

3.4) How do the participants learn to provide 'Positive feedback' messages?

Table 8.10 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refusing help and accepting help of each contingent hint in giving 'Positive feedback'

Giving 'Positive feedback'	Experiment group (EG)			Comparison group (CG)		
	PT1	PT2	PT3	PT4	PT5	PT6
% of all successes in giving 'Positive feedback' (with scaffolding)	55.56	53.83	100	-	-	-
% of all successes in giving 'Positive feedback' (without scaffolding)	100	71.43	100	0	0	28.57
% of all 'refuse help' in giving 'Positive feedback' (with scaffolding)	25	42.86	0	-	-	-
% of all 'accept help' in giving 'Positive feedback' (with scaffolding)	75	57.14	0	-	-	-
% of all successes in giving 'Positive feedback' (Pre-test)	0	100	100	0	0	100
% of all successes in giving 'Positive feedback' (Post-test)	100	100	100	0	0	0

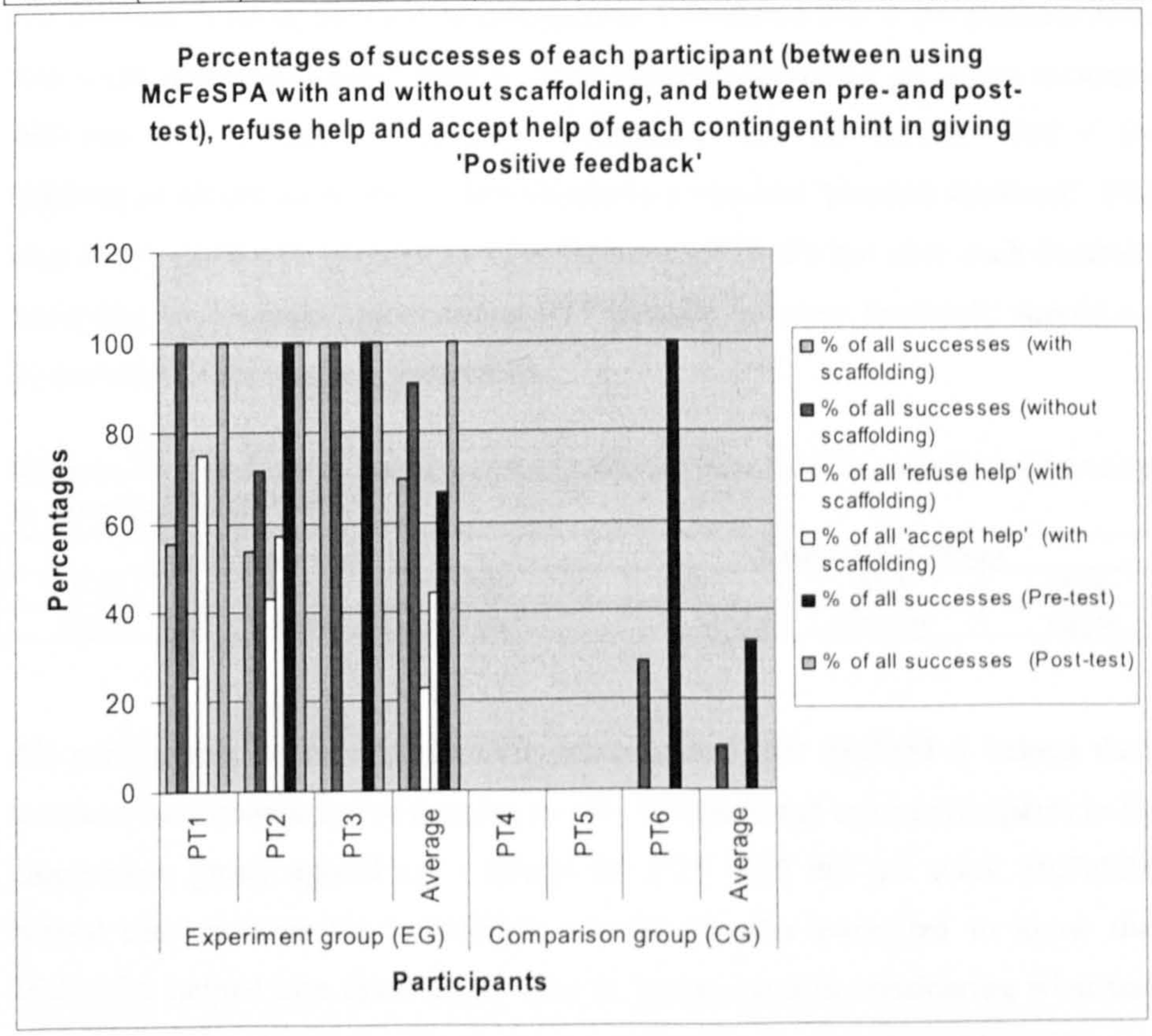


Figure 8.9 Percentages of successes of each participant (between using McFeSPA with and without scaffolding, and between pre- and post-test), refuse help and accept help of each contingent hint in giving 'Positive feedback'

According to Table 8.10 and Figure 8.9, results from the post-test showed that all participants in the experimental group gave an appropriate 'positive feedback' pattern according to the principle of McFeSPA i.e. error messages in between two instances of 'positive feedback' while PT1 is the one who improved

giving such feedback. PT3 gave such feedback for both paper tests. However, PT2's log-files did not correspond with the paper-test because PT2 tried to explore the system in giving feedback rather than learn to give such feedback i.e. PT2 always gave such feedback for both paper tests. Nevertheless, results from the post-test showed that all participants in the comparison group did not give an appropriate pattern for 'positive feedback' - we assume they did not learn to give such feedback from the system. Even though PT6 gave such feedback in the pre-test, once PT6 used the system then PT6 changed attitude in giving such feedback and thought that giving such feedback would depend on the number of errors that the students made in the current assignments. PT6 stated that if the students made less error, then they should receive more 'positive feedback' i.e. error messages between two instances of 'positive feedback.' PT5 also believed that if the students produced more errors, they should be given less 'positive feedback'. PT5 also had a similar perspective to PT6. However, PT5 did not give such feedback according to the paper tests because PT5 thought 'positive feedback' should not be provided if the student made errors.

4) How does McFeSPA help the participants increase their knowledge of learning to give feedback?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Agree	Agree	Agree	Disagree	Agree

All participants in the experimental group agreed that McFeSPA helped them increase their knowledge of learning to give feedback and most participants in the comparison group agreed too - except for PT5. PT5 did not think McFeSPA helped him increase his knowledge but felt he was interested to know that McFeSPA helped him think about what to process and in considering what task and what feedback to give. There are a number of ways in which McFeSPA can help the participants increase their knowledge about learning to give feedback. These are

- PT1 pointed out that citation in McFeSPA helped PT1 access information about feedback giving skills. PT1 felt that it is the result of different kinds of feedback and to explore the different skills that the system tried to teach i.e. PT1 achieved the benefit of scaffolding from McFeSPA by exploring

the theory of giving feedback from the glossary.

- PT2 thought being systematic in presenting the different kinds of errors (e.g. Design/Implementation/Style) provided a way of organizing the marking. The structure of giving feedback helped PT2 increase PT2's knowledge in learning to give feedback i.e. PT2 obtained the advantage in learning to give feedback from McFeSPA.
- PT3 agreed that McFeSPA helped PT3 increase PT3's knowledge/ understanding on the issue of learning to give feedback only a little because PT3 was familiar with giving feedback in a standard way ("*because I am used to giving feedback in the normal way, in the standard way so it's hard to change in an hour. It happens like this for two years. I am quite used to giving feedback in the same way*"). According to the triangulation data PT3 didn't learn much from McFeSPA because PT3 mostly refused help offered by McFeSPA and didn't try to provide the appropriate answer according to McFeSPA. Also PT3 believed in his own marking knowledge. This is probably the case because PT3, as an adult, has experience in giving feedback. PT3 mostly provided good feedback and rarely received any help offered by the system. Thus, PT3 felt that his knowledge of giving feedback increased by only a small amount.
- PT4 thought McFeSPA helped PT4 give more examples of feedback because McFeSPA provides some feedback messages that could help the TA in learning to give feedback.
- PT6 thought McFeSPA helped PT6 consider giving detailed feedback and PT6 thought that this could depend on what the student's previous errors were and what the current errors the student made were.

5) Can participants learn more about giving quality feedback by using McFeSPA again?

Experiment Group			Comparison Group		
PT1	**PT2**	**PT3**	**PT4**	**PT5**	**PT6**
Disagree	Agree	Disagree	Agree	Disagree	Agree

Some participants disagreed that they can learn more about giving good feedback with McFeSPA (e.g. PT1 pointed out that PT1 would not learn again because PT1 felt exhausted to learn any further teaching skills). However, PT3 felt PT3 could do this if the system provides more theory in giving feedback ("*Need more theory*

before this"). Results from log-files showed that PT3 never explored the glossary of McFeSPA in which this interface included theory about giving quality feedback - particularly the feedback that was measured in the experiment, therefore PT3 would have benefited by using this interface.

6) What are participants' perspectives in the experimental group of receiving help messages?

6.1) In the experimental group, were the help messages easy to understand?

Experiment Group		
PT1	PT2	PT3
Disagree	Agree	Disagree

PT1 and PT3 did not agree that the help messages in McFeSPA were helpful because PT3 did not understand the help messages and PT1 felt it was unclear as to what the messages were referring to. The application of contingent help may not have appealed to them. They are adults and it seems that they did not like the vague messages to encourage them to give good feedback while the system offered help. They might have preferred the system to relay the correct answer immediately to help them process the step quickly.

6.2) In the experimental group, were the help messages useful?

Experiment Group		
PT1	PT2	PT3
Disagree	Agree	Agree

Most participants in the experimental group agreed that help messages in McFeSPA were useful except for PT1. PT1 stated that PT1 had to take time to think about what the system meant by PT1's skill as a marker. It is likely that McFeSPA helped PT1 through a metacognitive scaffold in giving good feedback with providing different levels of help messages according to McFeSPA.

6.3) Did the help messages help the participants in the experimental group improve their skills in giving feedback?

Experiment Group		
PT1	PT2	PT3
Agree	Agree	Disagree

Most participants in the experiments group agreed that help messages of McFeSPA helped them improve their skill in giving feedback except for PT3 because PT3 felt that PT3 never used the help messages ("*Never used them*") and also PT3 mostly provided an appropriate answer according to McFeSPA. As a result PT3 received less help offers from the system.

7) Is it possible to learn how to use McFeSPA without any assistance from the system?

All participants in the experimental group agreed that it is possible to learn McFeSPA without any assistance from the system. PT1 said that it is possible to learn if the system provided a help file button. It would appear that PT1 did not know that the handout for using the system could help PT1 learn without any assistance from the system. PT2 said that PT2 could learn McFeSPA without any help from the system because PT2 has some familiarity with using it. PT3 said that PT3 can use McFeSPA but needed some form of training to use it properly.

8) How does McFeSPA help the participants to reflect/rethink on their skills in giving feedback?

All participants agreed that McFeSPA helped them reflect/rethink their skills in giving feedback. There are a number of areas that McFeSPA can help them achieve this, which are;

- PT1 thought providing different approaches in giving feedback is good when McFeSPA reported what PT1 was doing wrong. PT1 agreed that PT1 achieved the benefit of the system's suggestions.
- PT2 thought providing automated marking is very good because PT2 felt it helped PT2 mark quickly i.e. PT2 agreed that McFeSPA helped PT2 to reflect/rethink PT2's skills in giving feedback.
- PT3 thought McFeSPA gave more structure and helped PT3 see mistakes rather than PT3 just seeing what PT3 wanted to give.
- PT4 thought McFeSPA helped PT4 reflect/rethink about skills in giving feedback because PT4 felt it provided the idea that feedback should have more than one type.
- PT5 agreed that McFeSPA helped PT5 reflect/rethink the skills in giving feedback because PT5 felt that the measurement of McFeSPA worked well.
- PT6 agreed that McFeSPA helped PT6 reflect/rethink the skills in giving feedback because it referred to which feedback, which type of errors that the students made and encouraged PT6 give more detailed feedback even though the student made common errors (e.g. missing indentation)

To sum up, McFeSPA helped the participants to rethink/ reflect on their skill in giving feedback which included 1) report what was wrong (PT1, PT3), 2)

automatic way (PT2), 3) more structure (help to see mistakes) (PT1, PT2, PT3) 4) type of feedback is more than one (PT4) 5) measurement of McFeSPA worked well (PT5) 6) which feedback to give/which type of error the student made (PT6) 7) encouraging to give more detailed feedback (PT1, PT6).

9) What are the participants' perspectives about the representation of their skill in giving feedback?

9.1) Did the representation of each participant's skill in giving feedback after using McFeSPA make each participant realise that he/she needed to improve his/her skills at giving feedback?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Agree	Agree	Strongly agree	Strongly disagree	disagree

All participants in the experimental group agreed that the representation of their skills in giving feedback made them realise that they needed to improve their skills at giving feedback. However, in the comparison group PT5 and PT6 did not agree with this so it might be that scaffolding is an important factor. PT5 and PT6 did not realize that the glossary explained the meaning of each type of feedback, and therefore they did not understand what each feedback meant?

9.2) Did representation of each participant's skill at giving feedback (skill meter) help each participant think more about his/her skills?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Agree	Agree	Strongly agree	Agree	Disagree

Almost all participants agreed that the representation of their skill at giving feedback (skill meter) helped them think more about their skills except for PT6 because PT6 did not understand the representation used. However, PT6 thought the system helped PT6 think about PT6's skill. It should be noted that PT6 never used the glossary to explore each meaning of giving feedback in McFeSPA.

10) What were each participants' perspective about the effectiveness of using McFeSPA

10.1) Do the participants agree that McFeSPA can help them to give quality feedback to their students?

All participants in both groups, except PT5, agreed that McFeSPA could help them give quality feedback to their students. PT5 thought that the marker needed

to know and understand the error before writing the feedback. PT5 felt that McFeSPA did not help with this. It is true that McFeSPA doesn't give a complete analysis of all kinds of error types because McFeSPA was developed for experimental purposes and represented only some kinds of errors in order to test the hypotheses. It is also true that McFeSPA did not help because the aim of this research is to help people learn to give feedback. It did not focus on helping people understand the errors before giving feedback. We assume that the marker knows and understands the error well but maybe lacking knowledge and skills in giving good feedback. McFeSPA helped the participants in the following areas;

- PT1 thought McFeSPA helped PT1 structure and give more detailed feedback.
- PT2 thought McFeSPA helped PT2 mark quickly and improve the quality of feedback in terms of time (saving time in marking)
- PT3 thought that McFeSPA allowed the TA to add more errors.

10.2) Did McFeSPA help participants to finish work, quickly, effectively and improve their productivity?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Agree	Agree	Agree	Disagree	Disagree

Most participants thought McFeSPA could help them finish work quickly, effectively and productively except for PT5 and PT6. PT5 stated that PT5 would not use it for PT5's work and also felt more automated assistance was needed. While PT6 thought McFeSPA was developed for experimental purposes and was as yet unhelpful to PT6; however, PT6 felt that if PT6 was using it to mark, it would be useful.

10.3) Would the participants like to use McFeSPA in giving feedback to their students?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Agree	Agree	Agree	Disagree	Agree

Almost all participants would like to use McFeSPA in giving feedback to their students except for PT5 because PT5 does not teach Prolog anymore and thought that the hard part is to find the errors not writing the feedback.

11) Can the participants apply the knowledge they obtained by using the system to mark any programming assignment (not only Prolog)?

All participants agreed that they can apply the knowledge they obtained by using the system to mark any programming assignment (not only Prolog). However, PT1 thought whilst in theory it is possible it might be complicated to put into practice.

12) What is the participants' level of satisfaction with using McFeSPA?

12.1) Did the participants enjoy learning with McFeSPA?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Strongly agree	Agree	Agree	Strongly agree	Agree	Agree

All participants agreed that they enjoyed learning to give feedback with McFeSPA.

12.2) Can McFeSPA be frustrating?

Experiment Group			Comparison Group		
PT1	PT2	PT3	PT4	PT5	PT6
Agree	Disagree	Agree	Strongly Disagree	Agree	Disagree

Some participants felt McFeSPA to be frustrating in some respects. PT1 felt frustrated when PT1 was told that PT1 made a mistake but did not know what it was i.e. McFeSPA encouraged PT1 to think about giving feedback, and therefore does not give the answer directly. PT3 thought McFeSPA might be a complex program while PT5 still needed McFeSPA to be more automatic. All our participants were adults and according to Knowles adult learners have self directed goal/learning (Knowles, 1988). They need to achieve a goal properly especially if they are experienced TAs who may need the system to process their task quickly. They may have forgotten to play the role as a new TA while they were using the system in the experimental environment.

12.3) What did the participants like in particular about McFeSPA?

PT1 liked the automated analysis tool for finding errors and giving the location point. PT2 thought McFeSPA worked quickly and was easy to use. PT3 liked McFeSPA as it allowed the user to extend the errors reported. PT4 liked McFeSPA in that it automatically generated a report and provided adaptable

feedback messages. PT5 liked McFeSPA in that it provided ideas to help student and tutor marking, and also having a comparison of students. PT6 liked McFeSPA in that it is easy to use and helped the marker to think about the way they mark.

12.4) What did the participants dislike in particular about McFeSPA?
In terms of learnability issues, PT2 did not like using McFeSPA with scaffolding when it popped up a message and informed PT2 to give the appropriate feedback pattern according to McFeSPA.

In terms of usability issues, PT1 thought the range of feedback limited. However, there are a number of lists of comments that could be added in the future. If we achieve these files, we could plug in all comments to the system so that all TAs will have several examples of feedback messages to select and adapt for use. PT6 disliked McFeSPA because PT6 did not understand the representation of the skill meter. However, PT6 could have understood this if PT6 had explored the glossary button.

At some point of using McFeSPA, most participants disliked McFeSPA's implementation in terms of usability issues. As described earlier, McFeSPA was implemented for experimental purposes to test the hypotheses efficiently. The current version of McFeSPA is not ready for delivery into the real world. It needs further work (see discussion).

8.10 Summary of analysis of the results

In this chapter we presented the results of the evaluation study of McFeSPA as the approaches of McFeSPA and evaluation study of the feasibility of scaffolding TAs to help them learn to give feedback while engaged in an 'authentic' task as the implementation of McFeSPA. This can be seen in Table 8.11.

In general, according to Table 8.11, all participants in the experimental group accepted the approach of McFeSPA in giving 'Feedback loop' and 'Individual feedback', 'Detailed/Elaborative feedback', 'Important/Specific feedback', and pattern for giving appropriate 'positive feedback'. They also mostly accepted the implementation for giving feedback except for the implementation of giving 'Feedback loop' and 'Individual feedback' because they

did not like the comparison of the previous number of errors with the current number of errors.

Table 8.11 The acceptance of participants to McFeSPA's approaches and McFeSPA's implementation (by triangulation data) when '√' means agreement, 'X' means disagreement, and 'some' means mixed.

Feedback type	Acceptance of McFeSPA's approach						Acceptance of McFeSPA's implementation					
	Experiment Group			Comparison Group			Experiment Group			Comparison Group		
	PT1	PT2	PT3	PT4	PT5	PT6	PT1	PT2	PT3	PT4	PT5	PT6
'Feedback Loop' and 'Individual feedback'	√	√	√	X	√	√	X	X	X	N/A	X	√
'Detailed/ Elaborative feedback'	√	√	√	√	√	√	√	√	√	√	√	√
'Important/ Specific feedback'	√	√	√	X	√	some	√	√	√	N/A	√	√
Pattern for giving appropriate 'Positive feedback'	√	√	√	some	some	some	√	√	√	N/A	√	√

PT1 suggested that 'Feedback loop' and 'Individual feedback' could depend on important errors, not the number of errors. PT2 pointed out that an explanation of the history of a student's errors should not be provided again if the student repeated the same errors as before. PT3 suggested that such an explanation should not be provided for the simple errors because PT3 believe that it is enough to point out the errors. McFeSPA may present both important errors and simple errors to TAs in order to remind them to give better quality feedback to students. McFeSPA may give too much information to the TAs and not enough quality information so this could be improved in the later version. If the feedback giver does not need to inform common errors to students, how do we know they know which are common errors? Thus, they need to understand this before reporting or explaining the errors to students (for further details see question 1, 2, and 4 in Section 8.11).

Almost all participants in the comparison group accepted the approaches and the implementation of McFeSPA in giving 'Feedback loop' and 'Individual feedback', 'Detailed/Elaborative feedback', 'Important/Specific feedback', and

pattern for appropriate 'positive feedback'. PT5 did not agree with the implementation of 'Feedback loop' and 'Individual feedback' but PT5 accepted the approach of giving such feedback because PT5 needed the system to generate such feedback automatically while PT4 thought giving explanations of error messages to students according McFeSPA is enough without adding the history of students' errors (i.e. 'feedback loop' and 'individual feedback'). PT6 thought giving such feedback should depend on the student's level (for further detail see question 3 in Section 8.11).

Regarding giving 'Detailed/Elaborative feedback', all participants accepted the approach and the implementation of such feedback. However, PT3 suggested giving such feedback should depend on important errors while common errors need only give the error line number. PT6 also agreed that giving such feedback depended on the history of student's errors because PT6 did not give such feedback to some errors that the student made before (for further detail see question 5 in Section 8.11).

Specifically, almost all participants accepted the approaches and the implementation in giving appropriate 'positive feedback'. However, PT5 and PT6 believed that giving such feedback should depend on the number of errors that the student made i.e. if the students made less errors, they would give more 'positive feedback' while if the student made more errors, they would give less 'positive feedback'. Even though PT5 and PT6 are experienced TAs and they used McFeSPA without scaffolding, they did not achieve the benefit of scaffolding of giving 'positive feedback'. PT5 and PT6 could learn more in giving feedback with scaffolding because they may not understand the importance of motivating weak students (see question 6 in Section 8.11 for more details).

As can be seen from the analysis of the results (questions 1, 8, 12.1) presented in the previous section, all participants agreed that the basic idea of McFeSPA was helpful and they enjoyed learning with McFeSPA. All participants agreed that they can apply the knowledge they obtained by using the system to mark any programming assignment (not only Prolog). All of them agreed that McFeSPA helped them reflect/rethink their skills in giving feedback which include 1) report what was wrong, 2) automatic way, 3) more structure (help to

see the mistake) 4) type of feedback is more than one 5) measurement of McFeSPA worked well 6) which feedback to give/which type of error the student made 7) encouraging to give more detailed feedback.

According to Table 8.12, most participants in the experimental group improved their knowledge of giving 'Feedback loop' and 'Individual feedback', and 'Important/Specific feedback'. PT1 improved and benefited in giving feedback pattern for appropriate 'positive feedback' while the other participants always gave such feedback. PT5 in the comparison group improved giving 'Detailed/Elaborative feedback'.

Table 8.12 Participants' improved knowledge of giving feedback by comparison of their pre- and post-test results.

Improved knowledge of giving feedback	Experiment Group			Comparison Group		
	PT1	PT2	PT3	PT4	PT5	PT6
'Feedback Loop' and 'Individual feedback'	-	√	√	-	-	-
'Detailed/ Elaborative feedback'	-	-	-	-	√	-
'Important/ Specific feedback'	√	√	-	-	-	-
Pattern for giving appropriate 'Positive feedback'	√	-	-	-	-	-
Overall of participants' perspective	√	√	√	√	X	√

Almost all participants agreed that McFeSPA helped them increase their knowledge of learning to give feedback except for PT5. However PT5 was interested in how McFeSPA helped PT5 think about the process, what tasks to consider, and what feedback to give. PT5 considers giving feedback is to correct the error rather than motivating the students. There are some interesting features of McFeSPA that help the TAs. These are 1) Citation of theory of giving feedback that was presented in the glossary 2) Presenting different kinds of errors (design/ implementation/ style) and providing an organized way of marking with the structure of giving feedback 3) examples of feedback 4) previous student's errors.

Almost all participants agreed that McFeSPA can help them to give quality feedback and would like to use it in giving feedback to their students except for PT5 because PT5 thought that the marker needed to know and understand the error before writing the feedback. PT5 also said that PT5 did not teach Prolog any more and thought that the hard part is finding the error not writing the errors i.e. PT5 is concerned about correcting errors rather than motivating students.

Almost all participants agreed that representation of their skill at giving feedback (skill meter) helped them think more about their skills except for PT6 because PT6 did not understand the representation. PT6 also never explored the glossary in which the representation of the skill meter was explained.

Regarding participants in the experimental group, all of them agreed that the representation of their skills in giving feedback after using McFeSPA made them realize that they need to improve their skills in giving feedback. Some of them did not understand the help messages and required more details with the use of help messages. Most of them agreed that the help messages were useful except for PT1 because PT1 said that it took a while to realize that what the help messages referred to was PT1's skill as a marker i.e. McFeSPA helped PT1 to think about PT1's performance in giving feedback but PT1 may not prefer to learn with this method (contingent help). Most of them agreed that help messages help them improve their skills in giving feedback except for PT3 because PT3 rarely used the help messages so PT3 received less offers of help from the system. Particularly, each participant in the experimental group learnt different skills by using McFeSPA according to their previous experience, as can be seen from analysis of the results (question 2).

Nevertheless, during some stage of using McFeSPA, almost all participants felt frustrated; they suggested improvements for the implementation (see question 12.4 in analysis of the results, usability evaluation in Chapter 7, and discussion section in this chapter).

In summary, the analysis of the results of this study indicate that McFeSPA could help TAs who mark programming assignments think/reflect on their feedback giving to provide quality feedback to students in general. There is some promise to the approach of McFeSPA and some to the implementation according to the discussion in the following.

8.11 Discussion

The evaluation discussed here is very much a preliminary exploration of the system and underlying approach, given a particular domain and a particular group

of users. We discussed with people how to develop a model of giving good feedback. We implemented the model by bringing psychology and abstract ideas together in training people to give good feedback. We have provided evidence using triangulation that McFeSPA is based on a model that helps the feedback giver improve their feedback. We then analysed the triangulation data to be a model to help the feedback giver. We again provided evidence by triangulation data that all participants used the system effectively for some kinds of good feedback giving. There were a number of reasons[27] that make it difficult to design a system to help at an appropriate time to assess whether the student's help requests are appropriate (Aleven & Koedinger, 2000; Aleven & Koedinger, 2001).

In the current version we designed and implemented help in McFeSPA depending on the TA's actions without considering how to provide help at an appropriate time to sufficiently test our hypotheses. The TA used the system in a simulated situation with pre-provided student scripts. However, we have learnt from the previous section that most participants agreed with McFeSPA's approaches in giving feedback in general with a whole training program by McFeSPA and the feedback given by principles of McFeSPA (see Chapter 5).

Most results we obtained correspond with our model (see Chapter 5) even though some aspects of the implementation are not consistent with the users' wishes (i.e. implementation of 'Feedback loop' and 'Individual feedback'). In general, the systems help for giving feedback to the feedback giver has benefits. Almost all participants in the experimental group improved giving 'Feedback loop' and 'Individual feedback', 'Important/Specific feedback' and one improved their use of a pattern for giving appropriate 'Positive feedback' while the rest always provided feedback according to McFeSPA's approaches. This is consistent with the participants' background information. All participants who took part in our studies are experienced teaching assistants. Most of them have been trained in giving feedback to students. They are all adults who use self-directed learning (Knowles, 1988). Self-directed learning can be enhanced with facilitation (Conlan et al., 2003) so they need tools to help them to process tasks effectively. They

[27] For example, it is not true to determine when a particular step in a particular problem at a particular time is far beyond a particular student's ability.

need to achieve goals especially since they are experienced TAs who may need the system to process tasks quickly. McFeSPA may distract some participants in the experimental group because they are adults who felt frustrated when the system told them they made a mistake without reporting the reason for the mistake.

The evidence has shown that some participants in the experimental group did not understand the help messages and required more details with the use of help messages. There is also a participant who did not like the way feedback messages were phrased. However, we have mentioned in relation to the system design (see Chapter 5) that we were careful about phrasing feedback. It is true that a native English speaking person may provide better phrasing of statements e.g. semi-negative statements, and we may have overlooked this in our implementation, but it is a legitimate and complex point. To have more reliability, we address this issue later (see Chapter 9). They may not like the system's phrasing (which may not correspond with their experience in giving feedback). Even though we tried to eliminate this, implementation may disturb some participants but it does not mean our model is wrong. It could be true that some of them may receive the top level of 'contingent help' in which it is vague feedback messages in order to encourage the participants to think about their giving of feedback. By that means, at some point of using McFeSPA, most participants felt dissatisfied with some of the features so they suggested improvements for the implementation (see Appendix I).

Nevertheless, the results have shown that our model can be accepted. It might be argued that some evidence is relative to the accuracy of McFeSPA's judgments i.e. the evaluation study indicates all participants agreed that the basic idea of McFeSPA was helpful and helped them reflect/rethink their skill in giving feedback. However, some evidence may be inappropriate in terms of implementation. In addition, we could not obtain novice TAs to take part in our study. The TAs in our study are adults and experienced in giving feedback so we might not achieve the results that novice TA's would have given. Even though McFeSPA was designed carefully with respect to the andragogical model (of adult learning theory see Chapter 5), implementation of McFeSPA might not be robustly efficient according to the TAs' need. In order to meet the TAs'

requirement, we need to improve McFeSPA according to their suggestions (See Appendix I) so that tools could comfort adults in learning to give feedback properly with respect to the andragogical model. Mature TAs may need to be treated differently (see Chapter 7, 9). Previous to using McFeSPA, they may have had some general experience in giving feedback which did not focus in such a detailed manner on student's needs. Thus, learning to give feedback with McFeSPA could encourage them to think about how to give good feedback.

Analysis of the results has shown what we learnt according to the following questions.

1) Should the 'history of student's error' (giving 'Feedback loop' and 'Individual feedback') depend on the important errors, not the number of errors?

There is evidence that some people find some errors more complicated. It could be helpful to give a summary of common errors as 'history of student error' to help students learn not to make the mistake next time if they made a number of simple errors. We argue that giving a history of student's error by summarizing of each type of error (both important and common errors) could help students learn their mistakes and could change their behavior so as not to make the mistakes next time, especially for low level students. In fact giving 'Feedback loop' and 'Individual feedback' is collecting all comments/feedback to a student for the next assignment i.e. by considering the previous student's errors/comments, we need to give 'Feedback loop' and 'Individual feedback' if the student made errors in the next assignment (or resubmission of the same assignment). In implementing giving 'Feedback loop' and 'Individual feedback', we need the TAs to consider the previous student's errors by viewing the student's profile to compare the number of student's errors in the previous submission with the current one of the same assignment. Evaluation studies indicate that all participants accepted the approaches by taking into account the history of student's errors. Some suggested that we should provide the student's previous code and not to compare the number of student's errors because it does not reflect the student's learning. Therefore, we should encourage the TA to add previous comments of each student to the student's feedback report if the student made the same type of errors again in order to measure their giving 'Feedback loop' and 'Individual

feedback' whether the student made common or critical errors by summarizing each type of error and sequencing them.

2) Should the explanation of the history of a student's errors not be provided again if the students repeat the same errors as before?

Even though the student had been provided with the detailed history of student's errors of the same type as before, the student still made the same errors. If the student made such errors again more than once, the explanation of the history of student's errors may not help them make fewer errors. Thus, it is necessary to explain the history of student's errors again if the students repeat the same errors as before. If the repetition becomes a problem then eliminating repetition appropriately may be needed to understand who needs the repetition and who does not need it. To do this, we need to have further empirical work to find out what the facts are. It may be the case that there are occasions when this information would be omitted but it is necessary to determine the circumstances that need to be taken into account. This could be a point for further work.

3) Should the explanation of the history of student's errors depend on the student's level?

Basically, regarding individual difference, giving feedback depends on the student's skill level. However, providing quality feedback to an individual should not only consider the learners skill level, but should also regard help from the tutor provided alongside the learning environment (see Chapter 3). It is not only low level students who may need to be provided with the explanation of the history of their errors, but high level students may make errors similar to those made in the past. So the history of student's errors could help them learn from their mistakes too.

4) Should the explanation of the history of student's errors not be provided for simple errors?

McFeSPA sorted the errors from critical errors to simple errors (design, implementation, and style error, see Chapter 4). It is not only the important errors that should be explained to the students as a history of student's errors, but also simple errors should be provided again if the student continues to repeat such

errors, this may trigger their memory to make less errors next time. To do this, we need to have further empirical work to test the system with real students and simple errors to see whether the explanation of simple errors is important to students.

5) Should 'Detailed feedback' depend on important errors but common errors should be pointed out only through the error line number?

It is an interesting issue to take into account that important errors should be explained as 'detailed feedback'. However, if the students made a common error repeatedly, we should provide 'detailed feedback' for common errors to the student as well. Thus, it is not necessary that 'detailed feedback' should be provided for only important errors.

6) Should giving 'Positive feedback' depend on the number of errors that the student made i.e. if the student made less errors, we should give more 'Positive feedback' while if the student made more errors, we should ignore 'Positive feedback'?

Several researchers (Docheff, 1990; Oxford_Centre_for_Staff_and_Learning_Development, 2002; Albrecht, 2005) have suggested that giving a feedback sandwich can motivate the student to improve their learning. Giving such feedback, and therefore providing effective feedback, has been used in the Open University in the UK and a great number of organizations. By starting with specific praise of an individual performance, then suggestions to improve the errors or report what was wrong in a constructive way, and ending with praise in terms of suggestions to improve. Thus, we should not give less praise or ignore giving 'positive feedback' when the students made more errors i.e. to praise the students' current performance and also praise by suggestion on how to improve their learning can be a way of encouraging and motivating the students to learn better.

7) Should the marker who is trained to give quality feedback know and understand the errors before writing the feedback?

It is true that the marker should know and understand the errors before writing the feedback. McFeSPA generated three types of errors with a few

examples of each error types to test the hypotheses. However, PT5 may not know that McFeSPA can provide feasibility of editing the error messages because PT5 never used this feature, according to the log-files and observation. In fact, McFeSPA was implemented in order to emphasize training a TA to give good feedback by pre-provided errors so we selected to implement a few errors of each error type (four design errors, three implementation errors, and two style errors, see Chapter 4) in order to teach TAs and help them learn to give quality feedback. However, PT5 may expect using McFeSPA in a realistic way beyond our aim. Nevertheless, PT5's suggestion is helpful to be considered in line with improving the next version of McFeSPA.

In the previous section we presented which kind of feedback was accepted. Regarding our analysis of the results, we argue that McFeSPA's approaches provide a novel way of training people to give feedback even though currently McFeSPA is implemented to rely on only a few samples of three error types (design, implementation, and style issue, see Chapter 4) and doesn't deal with several important errors. (This feature should be added to the next version.) If we could update our model and re-implement McFeSPA's learning environment according to participants' suggestions, McFeSPA could be a facilitating tool in a new environment in training people to give feedback in marking Prolog assignments and can be a basis of new ITSs for scaffolding people to learn to give quality feedback. Furthermore, our approaches could apply to any marking programming assignment (not only Prolog), according to all participants' perspective. Therefore, we believe that McFeSPA's principles could provide a novel way in training people to give good feedback.

8.12 Summary

In this chapter, we have presented the methodology of evaluation of McFeSPA's learning environment, results, analysis of the results and discussion. We found that our participants learned differently by using McFeSPA. According to the hypotheses (in section 8.2), the evaluation study indicates that

1) Providing content scaffolding (i.e. detailed feedback in contingent hint, also see Chapter 5) in McFeSPA can help almost all TAs increase knowledge/ understanding in the issue of learning to give feedback according to their perspective. In addition, there were some TAs who improved their giving of feedback. Not all TAs improved, perhaps because they are experienced TAs and some had previously been trained in giving feedback.

2) Providing metacognitive scaffolding[28] in McFeSPA helped all TAs reflect/rethink their skills in giving feedback.

3) When the TAs obtained knowledge of giving quality feedback, the approach in McFeSPA of providing adaptable fading (see Chapter 5) allowed the TAs to learn alone without any support; however there is one TA in the experimental group that had a need to access a help-file to help them use the system. This suggests that even for adults there is a need to encourage help seeking.

To conclude, McFeSPA was implemented in a manner sufficient to test the hypotheses; however, experienced TAs required the system to help them to mark quickly with convenient tools supported properly in real use. Their requirements correspond with the design of the system but some do not correspond with the implementation of the system. We need to take into account their comments so that the next version of McFeSPA could support all the needs of both novice and experienced TAs. In the next chapter, we draw out our contribution, and speculate about the direction of future work.

[28] Metacognitive scaffolding (also see Chapter 5) in this chapter means each level of detailed feedback in contingent hint. This can also be general a pop-up message in using the system apart from feedback that encourages the participants to give good feedback.

Discussion and Conclusion

9.1 Introduction

The major contribution of this research lies in the design of the main components of a metacognitive feedback scaffolding system and their synthesis in a design framework that led to an innovative computer-support system for training people to improve the quality of their feedback to students. This work therefore focuses on the general area of constructing principles for giving better feedback. These principles are then used to enable people to be trained to give feedback through the use of a computer-support system. In particular, this research is relevant to building effective individualized tutoring systems, which are capable of adapting their help in order to improve the quality of feedback constrained to some extent by the user's requirement. We have emphasised the classification of student's error types by identifying critical and common errors which provides a model for the analysis of the student's performance. Our approach to giving quality feedback has been developed with regard to the investigation and analysis of a great deal of the available research literature on giving feedback. We then implemented our approach and re-implemented it again based on the suggestions made by evaluators during the usability evaluation. Finally, we evaluated our approach and implementation with a best fit of people whose qualifications corresponded with our needs and then analysed the results. In this chapter we present a summary of the results. We will then describe the main achievements of this research, and the contributions in relation to research fields will be outlined. After that we will propose directions of feasible research with the current architecture and applications. We will follow this by running through some limitations of our approach and possible extensions.

9.2 Synthesis and summary of the research

This research was carried out in order to answer three main questions. First, how does such a computer-support system help TAs provide quality feedback?

Second, how does the computer-support system help TAs to improve their skills at giving feedback? Third, can a computer-support system promote better help-seeking activities by providing feedback to the TAs? To answer these questions, this research described a framework of metacognitive feedback scaffolding. This enabled the development of a computer-support system that involved the TAs or novice teachers directly in helping them to improve their feedback giving and allowing them to construct feedback according to McFeSPA's approach. We have summarised four main aspects suggested as the basis for the design of such systems. We also briefly described our usability evaluation and learnability evaluation. We describe this next.

9.2.1 An investigation of the provision of feedback from marking systems and the design of a framework for the analysis of a student's performance

This is presented in Chapter 4. We investigated providing feedback from semi-automatic/ automatic marking systems deployed in the real world and we have described the domain knowledge used for the analysis of student's errors/ weaknesses. We assigned the classification of types of student's errors in the programming domain from critical errors to common errors (i.e. design errors, implementation errors, and style errors) as a framework for presenting error messages which supports constructing/organising the feedback. The programming domain was selected for training TAs to give quality feedback in relation to marking programming assignments. It could also help the students to structure their programming.

9.2.2 An investigation of the way in which people can be trained to give quality feedback

This is presented in Chapter 3. From the investigation of the types of feedback used in the learning process and in marking assignments, we have analysed and described the knowledge of giving feedback that is suited for training the TA and the kinds of feedback that should be provided to students. This also included the investigation and analysis of the level of feedback content. We also proposed an

ontology for giving feedback with regard to our investigation. We have selected and synthesised the essential feedback from this ontology to design feedback sufficient for training a TA to give quality feedback to students.

9.2.3 An investigation of scaffolding systems and a design for training people in giving quality feedback

This is presented in Chapter 5. We took into account a number of systems that help the learners learn according to their contexts, we outlined our approach based on the analysis of these systems and adopted what appeared to be the most interesting approaches for designing a scaffolding framework to help people to learn to give quality feedback. We compared our system design with other scaffolding systems and found that those systems help the end user (student) to learn rather than help the user (TA) to learn to give help. We also designed McFeSPA's behaviour based on a general design of tutoring systems. Specifically, we designed McFeSPA's architecture to support a cognitive apprenticeship approach. McFeSPA's principles were based on the available literature on giving good feedback. We employed Knowles' model of andragogy to help in designing the system with regard to training adults for improving giving feedback. Finally, we presented an overview of our contextual design.

9.2.4 The Design of our Scenario-Based Scaffolding System

This is presented in Chapter 6. We have defined the scenario for the scaffolding system based on the theory of scenario-based system design (Carroll, 1995). We built scenarios and designed principles for using scaffolding. This design is helpful in implementing a system according to the TA's needs in an appropriate learning environment.

The formalisation of the above aspects (see Section 9.2.1-9.2.4) supports the implementation of a robust computer-support system for training adult learners to provide good feedback to students on their programming assignments in a realistic situation. Following the framework defined here, we developed a computer-based system to help the feedback giver improve their skills at giving feedback – called McFeSPA (described in Chapter 5). The system utilises the

"McFeSPA principles" and makes use of a contingent help approach. Its robust architecture has allowed us to demonstrate instantiations of McFeSPA's principles in two terminological domains – the programming domain and the feedback domain. This thesis has demonstrated an application of the framework sufficient to test the hypotheses under consideration (see Chapter 1).

9.2.5 The Implementation and its usability evaluation

This is presented in Chapter 7. In order to evaluate the usability of the system, we selected a promising design based on the potentially useful approaches described in Chapter 5. After revising the design and implementation of McFeSPA, we re-implemented the system based on the evaluators' suggestions. Results from the usability evaluation indicated that McFeSPA is a useful tool for helping people learn to give feedback but it needed to be improved to some extent to help adult learners learn properly.

9.2.6 Evaluation results

McFeSPA has been used for the validation of the metacognitive feedback scaffolding system framework proposed in this thesis. An empirical study with the system (presented in Chapter 8) was conducted to discover problems and explore potential advantages of the approach. As a whole, the framework has been considered adequate for testing the hypotheses of the research i.e. simulating effective scaffolding approaches. In order to aid the reproduction of the approaches in computer-support systems, we have discussed practical issues of the current implementation, pointing out problems and feasible improvements to the current architecture.

The empirical study (presented in Chapters 7 & 8) with McFeSPA provided us with an assessment of the advantages of the approach. We were also able to observe an improvement once some encouraging messages were removed. This reduced the feeling of frustration that users had reported. The presentation of the skill meter in giving different feedback to students can help us observe TA's performance. This also provides an explanation to TA's skill meter. The interpretation of the TAs' behaviour has been validated through the triangulation

of data. Triangulation is a valuable approach. It helps us understand the results we obtained based on the consistency of the evidence. Triangulation can reduce problems that arise due to incomplete evidence.

The study also allowed us to monitor the presence of metacognitive feedback with McFeSPA. The empirical triangulation allowed us to argue that McFeSPA is a useful approach, which may be employed in intelligent learning environments with three objectives with reference to our hypotheses. Firstly, we found that providing 'content scaffolding'[29] in McFeSPA can help most TAs increase their knowledge/ understanding on the issue of learning to give feedback according to their perspective. Secondly, providing metacognitive scaffolding (i.e. each level of detailed feedback in contingent hint) in McFeSPA helped all TAs reflect/rethink about their skills in giving feedback. Thirdly, when the TAs obtained knowledge of giving quality feedback, providing an adaptable fading approach in McFeSPA allowed the TAs to learn alone without any support. However, all participants in our study were experienced TAs. Most required the system to help them mark quickly with convenient tools. Their requirements correspond with the design of the system but some do not correspond with the implementation of the system. Moreover, because adults learn a little differently from students, we achieved a different perspective of using McFeSPA from our participants.

9.3 Contributions

With regard to the achievements of our work, we highlight our contribution to two research areas. These are described below.

[29] Content scaffolding in McFeSPA is detailed feedback using contingent hint or via a general pop-up message. Providing this kind of feedback by the system is apart from giving general feedback issues that encourage the participants to give good feedback.

9.3.1 Contributions to Artificial Intelligence in Education and Intelligent Tutoring Systems

We argue that we have provided a new and unique approach that can help TAs learn to give quality feedback. We would expect novice teachers or novice TAs to learn with our approach. Based on the combination of approaches as presented in Chapter 5 to be the McFeSPA's approach (e.g. providing adequate help and a supportive interface in using the system to learn to give feedback to students), the approach has been validated with a small group of TAs. A working system, McFeSPA, was developed in that it could be used for helping TAs to mark Prolog programs and train them to give quality feedback to students. There is little research on using a scaffolding approach with a computer-support system to help the novice teachers or teaching assistants learn to give quality feedback. For example, a semi-automatic/automatic marking system rarely provides any support to train novice teachers to give quality feedback on students' programming assignments. As far as we can tell, there is no comparable system. In the respect of this thesis, PRAM (Mansouri et al., 1998) developed as the Ceilidh system (Foxley et al., 1999) and upgraded to become the CourseMaster system (Foxley et al., 2001; Higgins et al., 2002) might be the nearest to our system in terms of domain of application. These systems do not provide any support or training to the feedback giver to help them provide better feedback to students. Even though several systems scaffold students to learn in a particular context or support help-seeking behavior (e.g. Ecolab (Luckin & du Boulay, 1999) , SCI-WISE (White et al., 1999), SE-Coach , PACT Geometry tutor etc.), they do not help adults learn in giving feedback to improve their feedback giving.

There are a number of systems that help people to seek help, but rarely do any of these systems explicitly help people to give help (though there may always be some learning). In respect of this thesis, I-help (Bull et al., 2001) and Kumar and colleagues' help system (Kumar et al., 1999) seem to be the nearest systems that help peers (not a trained teacher or tutor) give help to peers in a particular course but these systems do not directly support the feedback giver to give better feedback to students. McFeSPA's framework is different from the "helping the peer helper" framework because McFeSPA does not give help directly, but

gradually supports TAs and helps them learn to give better feedback by using a metacognitive feedback scaffolding system and with respect to metacognitive knowledge[30] i.e. it helps the TA become aware of their thinking about feedback and their knowledge about giving appropriate feedback. The McFeSPA design process, like McFeSPA's architecture, is informed by research on the classification of student's weaknesses, quality feedback, metacognition, contingent help, and scenario based design.

The study suggested that the scaffolding approach helped the learners to succeed in a way that on their own they could not accomplish. Scaffolding approaches have been applied effectively in a number of systems in the field of Artificial Intelligence in Education (AIED) (e.g. Ecolab (Luckin & du Boulay, 1999) , SCI-WISE (White et al., 1999), etc.). According to our study, McFeSPA's approach can help teaching assistants (TAs) improve the quality of feedback to students by training them in marking programming assignments and to improve their own understanding of how to give quality feedback i.e. it can help the TAs directly, and the students indirectly. With respect to the results of our study, we can state the contribution of this work in terms of the effectiveness of using scaffolding approaches in helping the TA give quality feedback to students. Thus, this work is a contribution to the field of AIED and Intelligent Tutoring Systems (ITSs) through its focus on how to help people to give quality feedback by training them in marking assignments applied to programming teaching.

9.3.2 Contributions to Formative Assessment and ICT

Constructive feedback is required by the principles of formative assessment (Gareis, 2006). This thesis proposes the design of constructive feedback with respect to giving appropriate feedback depending on the student's work. We develop a computer-support system to help the teacher to provide constructive feedback to students - mainly focusing on the "feedback sandwich" which starts from a 'positive feedback' message, followed by the classification and

[30] Flavell (1979) describes three kinds of metacognitive knowledge: 1) Awareness of knowledge- understands what one knows, what one does not know, and what one wants to know. This category may also include an awareness of others' knowledge. 2) Awareness of thinking – understanding cognitive tasks and the nature of what is required to complete them. 3) Awareness of thinking strategies – understanding approaches to directing learning.

organization of error messages with a suggestion to improve, and ends with an encouraging 'positive feedback' message. With respect to other systems that help a teacher or TA giving feedback to students (Denton, 2001b, 2001a; Moreale et al., 2002; Denton, 2003; Whitelock et al., 2003), they did not provide constructive feedback in helping a teacher or TA learn to give quality feedback to students. Our approach is innovative in that it provides more opportunities for fostering reflective thinking to the feedback giver. Hence, this research contributes to Information and Communication Technology (ICT) and formative assessment via technology.

9.3.3 Minor Contribution (Generality of our Approach)

Human-Computer Interaction (HCI) is concerned with the design, evaluation and implementation of interactive computing systems and with the study of their associated cognitive and social factors (Hewett et al., 1996). In addition, HCI also has an interest in the issue of feedback. McFeSPA's approaches were designed, implemented, evaluated, and validated with respect to the design principles of HCI. Research on applying different types of feedback (McKendree, 1990) mainly study of the impact of different types of feedback on students. This research has not emphasised helping the TAs give different types of feedback with different detailed feedback from general detail to specific detail.

Our research has demonstrated an original approach for training TAs giving different types of feedback to students with different levels of elaborative feedback provided by the system to the individual TA. This research contributes to the field of HCI based on an adaptive and adaptable interface in relation to helping the TAs use the system to provide quality feedback to student's programming assignments.

In sum, we have adapted several approaches (see Chapter 5) for building computer-support to simulate training people learning to give better feedback. Our framework is fairly general and may be applied in a different domain i.e. marking any programming language assignment as well as ones for Prolog, computer science courses, or mathematics, etc. McFeSPA's approach could help people learn from the system and apply to other contexts in which teachers are

trained to give quality feedback e.g. design of McFeSPA could help people in online assessment, supervision assessment, and student assessment, training supervisors to give feedback to employees or supervisees.

9.3.4 Discussion of Contributions

Broadly speaking, McFeSPA's approach provides a novel way of training people to give better feedback. This thesis demonstrates the empirical study of training people to give feedback with a computer-support system and the results indicate acceptance by most TAs. While most ITSs give feedback to students, McFeSPA is a new ITS that gives feedback to teachers i.e. McFeSPA provides a "new environment" as a basis of a new kind of ITS.

In addition, we expect that our feedback ontology (presented in Chapter 3) could help people perceive that giving quality feedback is complex and depends on a number of factors. With regard to the other factors, we need to redesign McFeSPA to employ the ontology effectively. Therefore, we believe that our approach could contribute to the people who want to design and implement a system for helping people to improve giving quality feedback.

We believe that using McFeSPA as a support and training tool to help novice TAs learn to give feedback can help them achieve improved skills; however, while McFeSPA may not be the best way, it can help TAs to do better when they have not learnt to give appropriate feedback (according to McFeSPA) before. Therefore, we would like to demonstrate that McFeSPA is effective enough for the purpose it is intended for. After iterative refinement of the system, we believe that McFeSPA will be a useful tool in helping TAs learn to improve their feedback giving in the real world.

9.4 Implication for the research

Currently, McFeSPA has been used by TAs. They were trained by the system to give different kinds of feedback. The evidence we have is that the TAs were mainly satisfied. Most TAs in the experimental group improved their feedback giving. However, McFeSPA needs to be improved based on an analysis of their

suggestions (see Appendix I). Therefore, we should provide more varieties of support to the system.

Having proposed our contributions in the previous section, in this section we elaborate the possible lines of development. To do this, there are a number of issues that need to be considered.

- **Could the McFeSPA approach be used to research further issues in training TAs to give good feedback?**

As can be seen from the problem of giving good feedback (see Section 1.2.7 in Chapter 1), it is not easy to give good feedback, especially for TAs in the field of computer science who may not have any experience or training in giving feedback. In addition even if automated marking of assignments can help the teachers mark, novice teachers may not have been trained to give feedback (Dennis et al., 2002). Furthermore, Brinko (1993) also reported that many teachers have not been trained to give feedback and also there is very little research in this area. Even though the evidence so far is positive, more research needs to be done (e.g. Can 'Contigent help' help TAs learn to give feedback if it is possible to phrase feedback well enough to encourage them to learn? How can we encourage adults to learn to give feedback? Thus, the McFeSPA approach could be used to research further issues in training TAs to give good feedback.

- **Could McFeSPA as a stand alone computer-support system (i.e. a cognitive tool) turn out to be very useful?**

Of course, mentors can teach/help TAs in giving quality feedback. However, problems (mentioned in Chapter 1) such as inadequate time to teach TAs to provide quality feedback, and a lack of consistency can cause problems (Brinko, 1993). Research has shown that most ITSs help people to seek help (Bull et al., 2001), but rarely do ITS help people to give help. There are not many systems that are designed to tutor how to give help. Most systems are aimed at students who are given help or who are giving help to peers rather than at adults learning or teachers learning to give help to students. There are rarely systems which employ the particular sequences to teach in the particular situated cognition and culture of learning (Brown et al., 1989) and suitable learning environment (Jeong-Im & Hannafin, 1999) in adult learning (Stein, 1998) such as providing a

scaffolding method in the situated situation[31] of marking an assignment together with learning to give quality feedback. That is to say the combination of McFeSPA's value with these approaches would be very useful. Employing a stand alone computer as a cognitive tool should be helpful for them to learn by themselves, individually, anytime, and anywhere. Because of computer support as a scaffolding system, they may learn repeatedly according to their desire to improve giving feedback skill. As a consequence, learning to give better feedback to students by using the scaffolding system in an appropriate situation, such as marking assignments, could result in the development of skills for providing quality feedback to the feedback givers and help them reflect on their feedback giving.

- **Should McFeSPA be re-implemented in a different language from Visual Basic?**

We have shown that there is potential for McFeSPA to help TAs to improve their feedback giving in the area of marking programming assignments. The program's platform used to create a prototype of McFeSPA is Visual Basic which does provide support for plug-in Prolog. While not as sophisticated a language as C++ or Java, Visual Basic allows an object-oriented, agent-based style of programming; and it can handle message passing and data tracking. For stand-alone use, Visual Basic is sufficient but if McFeSPA were to be used on the web then it might be more efficient to reprogram McFeSPA in Java.

- **Which is more important, providing a system's user with freedom or with guidance and support?**

We argue that McFeSPA should contain two phases 1) providing a system's user with freedom could suit for the experienced TAs who needs to process tasks quickly. Experienced TAs, like experienced adult learners, generally need a comfortable learning environment (Knowles et al., 1998). Experienced TAs do not necessarily give the best feedback. Learning to give feedback is a life long process. McFeSPA tries to coordinate their skills to provide better quality feedback. TAs may have experience in spotting errors but they may never think

[31] Situated situation' means a defined situation or event's specific surroundings or context where the situation is placed

about giving good feedback and may be very inexperienced at managing people. So, it is important to judge the TAs' behaviour in giving feedback on students' programming assignments i.e. TAs may have experience in marking errors but be inexperienced in giving quality feedback. Hence, the system could help them learn to give better feedback. 2) providing a system's user with guidance and support could help the novice TA learn to give better feedback. Novice TAs who are required to achieve their goal of learning have their own self-directed learning skills. They could be motivated to learn by employing the principles of adult learning (Knowles et al., 1998). However, according to the information collected on the participants' backgrounds, we know that all participants who took part in our studies have some experience. Most of them have been trained in giving feedback to students. This contrasts with our wish to use the approach with novice TAs to get the maximum benefit. However, we could not obtain TAs who were just beginning their employment in this role. It might have been possible to recruit novices if the evaluation of McFeSPA had coincided with the start of the academic year. Although TAs in our study are experienced TAs, they provided us with interesting ideas and useful suggestions to improve the system for the betterment of inexperienced TA's who may use the system. All our TA's have significant experience in computer science and some have strong feelings about the usability of the system so their suggestions are helpful to improve later versions of the system. Some of them improved their feedback giving while one who had more familiarity with manual marking needed more time to use the system. This TA has an epistemological belief (Hoffer & Pintrich, 1997) in manual marking which meant that the TA felt he/she could mark students' assignments better without the system. With reference to the notion of the reflective practitioner (Schön, 1983), the TAs will learn better when they have used McFeSPA several times as reflection-in-action in order to help them think about what they are giving as feedback to the students. Thus, using the system iteratively could help the TA change their belief that manual marking is always better.

- **Should we accommodate people with different approaches to giving feedback?**

Currently we train TAs to give different kinds of feedback e.g. 'Feedback loop' and 'Individual feedback', 'Detailed/ Elaborative feedback', 'Important/ Specific feedback', 'Positive feedback'. Although all TAs in the experimental group agreed with the approach, there are some suggestions provided to improve the implementation of the approach. To do this, we should accommodate people in giving different feedback to some extent with our reflected system. Even though it might not be flexible enough, it could go in the right direction toward giving better feedback. All used McFeSPA in a situated situation. Providing preferences is already an important part of the system. There should be a number of further research studies on what additional preferences are needed e.g. preference about which motivating and encouraging feedback message to give to students. The TAs can adapt the feedback. We could try with different kinds of preference with several TAs to determine how the theory of andragogy affects the use of McFeSPA.

- **Can 'contingent help' help TAs learn to give feedback if it is possible to phrase feedback well enough to encourage them to learn?**

Several research results (e.g. (Wood & Wood, 1999; Wood, 2001)) reported that contingent help can improve children/student's learning, however using such an approach with adult learners might distract their learning because adults do not need so much encouragement. Therefore some further work is needed to see if better phrasing of the feedback can help them learn to give better feedback. Currently we provide five levels of help to the TAs which may distract the TAs. Perhaps providing three levels of help could reduce their feeling of distraction. Thus, this needs to be tested with further empirical study on how many levels of contingent help suit the TAs in learning to give quality feedback.

- **How can we encourage adults to learn to give feedback?**

According to the principles of andragogy (the art and science of helping adults learn (Knowles, 1988)), adults need to be involved in the planning and evaluation of their instruction; their experience (including mistakes) provides the basis for learning activities; adults are most interested in learning about subjects

that have an immediate relevance to their job or personal life. To sufficiently test the hypothesis of the research, the current system might not be adequate to help the TAs marking students' assignments according to all their needs. Thus, providing a more complete facilitating tool for use in McFeSPA alongside adaptable and appropriate adaptive help could encourage adults to learn to give better feedback.

- **Should we study the TA's performance by time on task?**

TA's performance by time on task can be measured by timing their processing of several scripts or learning more about feedback over a period of time e.g. TAs may ask themselves whether they have learned to give feedback. They may reply that they did not learn to give better feedback but they do give feedback quite quickly. In other words, they may mark 1,000 scripts in a certain time and take time to learn to give feedback; however after doing that, they could give good feedback quite quickly. For the current study, we did not analyse TA's performance by time on task because we needed to analyse the TA's behaviour and TA's perspective in giving feedback. However we could expect that if TAs do the same task several times, they will become faster. This research is an initial research of building a system for helping TAs gives better feedback. The current version of McFeSPA is an adaptive help system but we describe it as "limited". "Adaptive help" is needed but more adaptation is possible. McFeSPA is aimed mostly on design issues rather than implementation ones. We have implemented a valid and valuable design to produce a simple version for the current study. After further study of help giving with the system by a number of TAs, McFeSPA should be redesigned and re-implemented according to their needs. Thus, a later version of McFeSPA could be useful to deal with time on task, probability, and using McFeSPA with adaptive help that depends on the TAs' level.

- **Should we employ McFeSPA with a TA and real students?**

The current version of McFeSPA was not designed to support standard questions, nor is it designed specifically to support the full range of interactions between the TA and real students - it emphasizes support for giving high quality feedback. In terms of providing support, it is not easy to guarantee that we have a complete set of techniques to help provide the best quality feedback because the

kinds of support that we can provide must be capable of helping the students and the feedback giver to improve their performance on current and later tasks. Thus, employing McFeSPA with a TA and real students needs further design, implementation and evaluation.

- **Should we provide more automated analysis - for every type of error?**

Our system does not give any marks as it is a semi-automated assessment system for helping markers to learn to give feedback i.e. helping TAs by scaffolding, contingent help, help seeking (temporary help). So little focus has been on metric based automated assessment - our system is another dimension of formative assessment that focuses on various types of error/weakness in student's code. We experimented with the current version of the system in a simulated situation with a pre-provided student script. If the completed scaffolding system were found to be satisfactory by a sufficient number of TAs, the system could be enhanced by adding automated analysis of student's errors for a wider range of errors. This could be useful for the direction of long-term research.

- **How well might the system scale for large assignments?**

The current version of McFeSPA was designed to support small assignments in order to help TAs learn to give quality feedback, but was not designed to support large assignments. The system needs to provide more automated analysis, better methods for managing large numbers of errors and improvements in the system's usability. It would be interesting to develop the system for larger assignments as a direction for long-term research because larger assignments would be more complicated therefore the TAs would learn to give feedback similar to a real life situation.

9.5 Future work

We have sketched out the achievements of this work. We now propose possible applications and enhancement of this research. We will first describe our short-term goals that concern the improvement of the current architecture. We then outline and elaborate our long-term research.

9.5.1 Directions of feasible research with the current architecture and applications

- **Employing McFeSPA with TAs to use frequently**

The study has shown that some TAs believe in traditional ways of marking. Using the system only once or twice might not help them learn enough. They need to use the system repeatedly. This is similar to outer loop, a technical term of behavior for tutoring systems (VanLehn, 2006). The outer loop in McFeSPA is to decide which task the TA should do next. McFeSPA needs to have another outer loop as the 3rd loop beyond the inner loop and the first outer loop i.e. the extra loop which is used by the system itself over many training processes with respect to the probability of various events (du Boulay, 2006).

- **Evaluating McFeSPA with more TAs**

Better phrasing might help the TA to learn more. Each level of contingent help should be provided as a feedback sandwich (with better phrasing). The feedback sandwich consists of reporting what was done well, stating what could be changed or improved in a constructive way, and then communicating how the improvement might be achieved e.g. "That is a good move, but it is not the appropriate answer. Try again. I am sure that you could do it better." To phrase feedback better for each level of contingent help, we could have more discussion with a greater number of TAs to seek their agreement before phrasing the examples of feedback for the next version (or plug in applicable good examples). We also need further evaluation of McFeSPA by more TAs to study different levels of users in order to enhance the system with adaptive help. This could help the system distinguish which users should be supported by the system or allow the users to use the system freely. In the evaluation study, we should adopt an electronic pre/post test and analyse the TA's results to see what types of feedback the TA is weak on and therefore focusing on helping them to improve their weak feedback giving areas by generating situations which require the type of feedback for which the inexperienced TA is weak.

9.5.2 Directions of long term research

- **Employing module of analysis of Prolog programming with classification of further error types**

In order to efficiently test the hypotheses, the current version of the system supports the analysis of a Prolog programming assignment with nine errors from three classifications of error types (design, implementation, and style errors according Chapter 4). To use the system in the real world, we need further research on the analysis of Prolog programs with further error types.

- **Employing McFeSPA with a TA and real students**

The current version of the system is not designed for real students to interact with the TAs/new teachers. In real life, students can ask about the feedback which they receive. Additional research can be done by evaluating McFeSPA with real students to study the different levels of each student. Moreover, the system should adapt its advice on giving feedback to the student's level. The system should record the student's level e.g. if it is a high level student, the system should give general feedback (brief feedback) - not detailed feedback.

- **Employing McFeSPA as an online application**

FAQ (Frequently Asked Question) and Answer Gardens are dynamic systems which are popular via E-mail or a web interface and are used to provide help. McFeSPA in the current version does not involve online feedback. Employing McFeSPA on the web could help the TA learn to give feedback anywhere. If this was provided more complete facilitating tool for use in McFeSPA alongside adaptable and appropriate adaptive help, McFeSPA can be a Virtual Learning Environment to support individualise learning in initial teacher or TA training. In addition, McFeSPA can be enhanced by integrating it with I-help (Kumar et al., 1999). McFeSPA could contribute to I-help by adding domain knowledge to the helper's assistant (Kumar et al., 1999), and all rules applied in McFeSPA could be added in the help-plan. Thus, it is necessary to do further design work and anticipate that an improved design will contribute to the people who require a system to help people in learning to give quality feedback.

- **Enhancing usability issues, improving the maintenance of design and implementation**

Enhancing the usability of McFeSPA can be done by improving the system according to the evaluators' and TAs' suggestion (see Chapter 7 and Appendix I). In addition, the current version of McFeSPA consists of asking only 6 questions, which are sufficient to evaluate the system. The next version of the system should contain more cases of 'asking questions' and employ natural language processing (NLP) to analyse the type of questions in order to classify questions and help the TAs learn giving better feedback. Furthermore, it is interesting to deal with NLP e.g. checking for impolite words, analysing language for encouraging people e.g. analysing types of negative/positive feedback. Besides, to motivate adult learners through using McFeSPA in the real world, the system could use spoken language, check user's timing, provide an animated agent, and so on.

Regarding the provision of feedback to students, some teachers do not give students the right answers directly, but they may give suggestions to read more books and give the reasons to the teachers. Teachers may not remember the association between chapters and pages. McFeSPA could help teachers in the case that students did badly in each area, and each area could then have a hyperlink to link into the chapter that relates to the student's mistake and that information should be put into the teacher's feedback.

To improve McFeSPA as a support tool, the system should provide a help button for each interface which can provide help to the user. By clicking the hint button, the system can show each step of help from the general detail to specific detail. This is hard wired for both the feedback pattern and adaptive feedback. Thus, we could need further research on feedback patterns to improve the next version of the system (e.g. by employing probability approaches).

In the long term research, our future research on guiding the TA/teacher will be related to Heuristic Design Patterns to give TAs/teachers plausible choices for the structure of feedback based on an ontology such as that discussed in Mizoguchi and Bourdeau (2000).

9.6 Summary

In this thesis, we designed, and implemented McFeSPA's approaches based on the literature review of how to give quality feedback (in Chapter 2); feedback design (in Chapter 3); analysis of student programming weaknesses relating to Prolog programming (in Chapter 4); scaffolding system design (in Chapter 5); and scenario-based scaffolding system design (in Chapter 6). In addition, we interviewed experts on feedback, on programming, and on Prolog programming in particular (see Appendix A).

We then built the initial system and performed several studies to test McFeSPA's usability after iteratively improving McFeSPA design through pilot evaluation. The last usability evaluation of McFeSPA is presented in Chapter 7. We also provided triangulation data with analysis and discussion of the results in this usability evaluation. In addition, the evaluation discussed in the previous chapter is very much a preliminary exploration of the system and underlying approach, given a particular domain and a particular group of users. We have also provided the evidence by triangulation that providing McFeSPA as a model to help the feedback giver has benefits.

We have shown, as the main contribution of this research, that McFeSPA's design led to an innovative computer-support system for training people to improve the quality of their feedback. McFeSPA can be a tool to help the TA/lecturer reflect - and enough to sharpen the feedback demand. However, there are various aspects of future research that need to be taken into account. We believe that further research on McFeSPA could improve our understanding of how support and training help towards people learning to improve their feedback giving.

Enhancement Themes. (2007). Quality Assurance Agency for Higher Education 2004 - 2007. Available: http://www.enhancementthemes.ac.uk/background/enhancmentThemes.asp [Accessed on 17 June 2007].

QAA Scotland. (2007). Quality Assurance Agency for Higher Education 2004 - 2007. Available: http://www.qaa.ac.uk/scotland/default.asp [Accessed on 17 June 2007].

Bales, R. F. (1950). A set of categories for the analysis of small group interaction. *American Socilogical Review, 15*, 257-263

Bhalero, A., & Ward, A. (2001). Towards electronically assisted peer assessment: a case study. *Association for Learning Technology Journal (ALT-J), 9*(1), 26-37

Bryan, C., & Clegg, K. (Eds.). (2006). *Innovative Assessment in Higher Education*.London, New York: Routledge.

Chickering, Z. F., & Gamson, A. W. (1991). *Applying the seven principles for good practice in undergraduate education*.San Francisco: Jossey-Bass.

Dweck, C. S., Mangels, J. A., & Good, C. (2004). Motivational effects on attention, cognition, and performance. In D. Y. Dai & R. J. Sternberg (Eds.), *Motivation, emotion, and cognition: Integrative perspectives on intellectual functioning and development*.London: Lawrence Erlbaum Associates.

Gibbs, G. (2006). How assessment frames student learning. In C. Bryan & K. Clegg (Eds.), *Innovative assessment in higher education*.London, New York: Routledge.

Matthew, R. (2004). Improving Feedback to Students (Link between Formative and Summative Assessment), Post-workshop reported on the 4th June 2004 in the St.Andrew Building, University of Glasgow. Available: http://www.enhancementthemes.ac.uk/documents/events/20040604/postworkshop7reportrevised.doc [Accessed on 17 June 2007].

Mueller, C. M., & Dweck, C. S. (1998). Praise for intelligence can undermine chaildren's motivation and performance. *Journal of Personality and Social Psychology, 75*(1), 33-52

Nicol, D., & Miligan, C. (2006). Rethinking technology-supported assessment in terms of the seven principles of good feedback practice. In C. Bryan & K. Clegg (Eds.), *Innovative Assessment in Higher Education* (pp. 64-77). London: Taylor and Francis Group Ltd.

Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: a model and seven principles of good feedback practice. *Studies in Higher Education, 31*(2), 199-218 Available: http://tltt.strath.ac.uk/REAP/public/Resources/DN_SHE_Final.pdf

Wenger, E. (1998). *Communities of Practice: Learning, Meaning, and Identity*.New York: Cambraidge University Press.

Whitelock, D., Watt, S., Raw, Y., & Moreale, E. (2003). Analysing tutor feedback to students: first steps towards constructing an electronic monitoring system. *Association for Learning Technology Journal (ALT-J), 11*(3), 31-42 Available: http://openmentor.comp.rgu.ac.uk/development/article.jsp

---

Additional bibliography

Bibiography

Ackerman, M. S., & Malone, T. W. (1990). *Answer Garden: A Tool for Growing Organizational Memory.* In Proceedings of the ACM Conference on Office Information Systems. Cambridge, MA. April, 1990, pp. 31-39.

Ackerman, M. S., & McDonald, D. W. (1996). *Answer Garden 2: Merging Organizational Memory with Collaborative Help.* In Proceedings of the 1996 ACM Conference on Computer support cooperative work, November 16-20, 1996, Boston, Massachusetts, United States

Ala-Mutka, K., & Järvinen, H.-M. (2004). *Assessment Process for Programming Assignments.* In Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT' 04)

Aleven, V., & Koedinger, K. R. (2000). *Limitations of student control: Do students know when they need help?* In Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000 Available: http://www-2.cs.cmu.edu/~aleven/aleven-koedinger-its2000.pdf [Accessed on 19 March 2004].

Aleven, V., & Koedinger, K. R. (2001). *Investigations into Help Seeking and Learning with a Cognitive Tutor* (Working Notes of the AIED 2001 Workshop "Help Provision And Help Seeking In Interactive Learning Environments."). Working Notes of the AIED 2001 Workshop "Help Provision And Help Seeking In Interactive Learning Environments."

Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallance, R. (2003). Help Seeking and Help Design. *Review of Educational Research, 73*(3), 277-320 Available: http://www.msu.edu/user/mccrory/pubs/Alevenetal.pdf

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons Learned. *The Journal of the Learning Sciences, 4*(2), 167-207 Available: http://act-r.psy.cmu.edu/papers/Lessons_Learned.html [Accessed on 14 October 2004].

Askew, S., & Lodge, C. (2000). Gift, ping-pong and Loop --linking feedback and learning. In S. Askew (Ed.), *Feedback for Learning*:Routledge Falmer, London.

Back, D. M. A. R. (1999). *Models of mentoring in initial teacher training: case studies within a partnership scheme in secondary school-based initial teacher training.* Unpublished Ph.D., East Anglia.

Bangert-Drowns, R. L., & Kozma, R. B. (1989). Assessing the design of instructional software. *Journal of Research on Computing in Education, 21*, 241-262

Beck, J., Stern, M., & Haugsjaa, E. (2002). Applications of AI in Education. ACM Crossroads: Student Magazine. Available: www.acm.org/crossroads/xrds3-1/aied.html [Accessed on 31 March 2003].

Bental, D. S. (1994). *Recognising design decisions in Prolog programs as a prelude to critiquing.* Unpublished Ph.D. thesis, Edinburgh.

Benyon, D., & Imaz, M. (1999). Metaphors and Models: Conceptual Foundations of Representations in Interactive Systems Development. *Human-Computer Interaction, 14*(1/2), 159-189 [Accessed on 12 March 2004].

Black, P., & Wiliam, D. (1998). Assessment and classroom learning. *Assessment in Education, 5*(1), 7-74

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher, 13*, 4-16

Bonk, C. J., Malikowski, S., Angeli, C., & Supplee, L. (2001). *Holy COW: Scaffolding Case-Based "Conferencing on the Web" With Preservice Teachers, CRLT Technical Report No. 15-01,June, 2001, Indiana University.*

Bratko, I. (2001). *Prolog programming for artificial intelligence.* Harlow: Addison-Wesley.

Brinko, K. T. (1993). The practice of giving feedback to improve teaching. *Journal of Higher Education, 64*(5), 575-594

Brna, P., Cox, R., & Good, J. (2001). Learning to Think and Communicate with Diagrams: 14 Questions to Consider. *Artificial Intelligence Review, 15*, 115-134

Brown, D. C., & Chandrasekaran, B. (1989). *Design Problem Solving: Knowledge Structures and Control Strategies*: Pitman.

Brown, G., Bull, J., & Pendlebury, M. (1997). *Assessing student learning in higher education*:London:Routledge.

Brown, G. A. (1973). *The effects of training upon the performance of students in teaching situations.* Unpublished D.Phil., Ulster.

Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning. *Educational Researcher, 18*(1), 32-42 Available: http://www.uog.edu/coe/ed451/tHEORY/SituatedCognitition1.pdf; http://www.exploratorium.edu/IFI/resources/museumeducation/situated.html [Accessed on 2 April 2004].

Brown, S., & Knight, P. (1994). *Assessing Learning in Higher Education.* London: Kogan Page Limited.

Bruner, J. (1984). Vygotsky's zone of proximal development: The hidden agenda. In B. Rogoff & J. Wertsch (Eds.), *Children's learning in the "Zone of proximal development: New directions for child development.* San Francisco: Jossey-Bass.

Brunot, S., Huguet, P., & Monteil, J. (2000). Performance feedback and self-focused attention in the classroom: When past and present interact. *Social Psychology of Education, 3*, 277-293

Bryc, W., & Pelikan, S. (1999). Online Exercises System (Authors: Wlodek Bryc and Stephan Pelikan). University of Cincinnati, Cincinnati, OH. Available: http://math.uc.edu/onex/demo.html [Accessed on 20 May 2004].

Bull, S., Greer, J., McCalla, G., & Kettel, L. (2001). *Help-Seeking in an Asynchronous Help Forum.* In Proceedings of the Workshop on Help Provision and Help Seeking in Interactive Learning Environments, International Conference on Artificial Intelligence in Education, San Antonio, Texas, 2001

Burstein, J., & Marcu, D. (2000). *Towards Using Text Summarization for Essay-Based Feedback.* In Proceedings of the Le 7e Conference Annuelle sur Le Traitement Automatique des Langues Naturelles TALN'2000.

Buscemi, C. (2003). Feedback in Computer Assisted Instruction and Computer Assisted Language Learning. Available:

http://www.edb.utexas.edu/mmresearch/Students96/Buscemi/researchproject.html [Accessed on 17 March 2003].
Butler, D., & Winne, P. (1995). Feedback and self-regulated learning: A theoretical synthesis. *Review of Educational Research, 65*(3), 245-281
Cagiltay, K. (2006). Scaffolding strategies in electronic performance support systems: types and challenges. *Innovations in Education and Teaching International, 43*(1), 93-103 [Accessed on 19 May 2006].
Carnell, E. (2000). Dialogue, discussion and feedback. In S. Askew (Ed.), *Feedback for Learning*:Routledge Falmer, London.
Carroll, J. M. (2000). *Making Use: Scenario-Based Design of Human-Computer Interactions*. London, England; Cambridge Massachusetts: the MIT Press.
Carroll, J. M. (Ed.). (1995). *Scenario-Based Design: Envisioning Work and Technology in System Development*. Canada: Hohn Wiley &Sons.
Carroll, J. M., & Rosson, M. B. (1992). Getting Around the Task-Artifact Cycle: How to Make Claims and Design by Scenario. *ACM Transactions on Information Systems, 10*(2), 181-212
Chai, L. (2003). *To Have or Have Not: An Examination of Feedback, Learner Control and Knowledge Type in Online Learning.* In Proceedings of the 36th Hawaii International Conference on System Sciences - 2003, Big Island, Hawaii Available: http://csdl.computer.org/comp/proceedings/hicss/2003/1874/01/187410006a.pdf Institute of Electrical and Electronics Engineers(IEEE), [Access: 07 April 2003]. [Accessed on 7 April 2003].
Chee, Y. S. (1995). Cognitive apprenticeship and its application to the teaching of Smalltalk in a multimedia interactive learning environment. *Instructional Science, 23*, 133-161 Available: http://www.comp.nus.edu.sg/~cheeys/Publications/1995/InstSc1995Chee.pdf [Accessed on 31 March 2004].
Chi, M. T. H., Siler, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25*(4), 471-533
Chung, G., & O'Neil, H. (1997). *Methodological Approaches to Online Scoring of Essays* (CSE Technical report 461): National Center for Research on Evaluation, Standards, and Student Testing.
Clarke, S. (2000). Getting it right- distance marking as accessible and effective feedback in primary classroom. In S. Askew (Ed.), *Feedback for Learning*. London: Routledge Falmer.
Cohen, P. A., Kulik, J. A., & Kulik, C. C. (1982). Educational outcomes of tutoring: a meta-analysis of findings. *American Educational Research Journal, 19*, 237-248
Collins-Brown, E. (2001). Successful Strategies for Using Asynchronous Discussion in College Courses. Available: http://www.connectedcreativity.com/Treatise/Treatise_EliCollinsBrown.doc [Accessed on 6 April 2004].
Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (pp. 453-494.). Hillsdale, NJ: Erlbaum.
Conati, C., & VanLehn, K. (1999). Teaching Meta-Cognitive Skills: Implementation and Evaluation of a Tutoring System to Guide Self-Explanation While Learning from Examples. In S. P. Lajoie & M. Vivet

(Eds.), *Artificial Intelligence in Education, Proceedings of AIED-99* (pp. 297-304). Amsterdam: IOS Press.

Conati, C., & VanLehn, K. (2000). Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. *International Journal of Artificial Intelligence in Education, 11*, 389-415

Conlan, J., Grabowski, S., & Smith, K. (2003). Current trends in adult education. In M. Orey (Ed.), *Emerging perspectives on learning, teaching, and technology.*

Cook, C., Burnett, M., & Boom, D. (1997). *A bug's eye view of immediate visual feedback in direct-manipulation programming systems.* In Proceedings of the Proceedings of Empirical Studies of Programmers, October, 1997.

Corbett, A., & Trask, H. (2000). *Instructional Interventions in Computer-Based Tutoring: Differential Impact on Learning Time and Accuracy.* In Proceedings of the ACM CHI' 2000 Conference on Human Factors in Computing Systems Available: http://www-2.cs.cmu.edu/~pact/corbett/Corbett&TraskCHI2000.pdf [Accessed on 12 July 2004].

Corbett, A. T., & Anderson, J. R. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction, 4*, 253-278

Corbett, A. T., & Anderson, J. R. (2001). *Locus of Feedback Control in Computer-Based Tutoring: Impact on Learning Rate, Achievement and Attitudes.* In Proceedings of the SIGCHI'01, March 31-April 4 2001, Seattle, WA, USA

Cousins, J. B., & Leithwood, K. A. (1986). Current Empirical Research on Evaluation Utilization. *Review of Educational Research, 56*(3), 331-364

Dempsey, J. V., & Wager, S. U. (1988). A taxonomy for Timing of Feedback in Computer Based Instruction. *Educational Technology, 28*(10), 20-25

Dennis, L., Mills, S., Smith, P., & Tucker, A. (2002). Automated Assessment for Large Groups. Available: http://www.cs.nott.ac.uk/~smx/PGCHE/ [Accessed on 11 July 2003].

Denton, P. (2001). Generating and e-mailing Feedback to Students Using MS Office. In M. Danson & C. Eabry (Eds.), *the fifth International CAA Conference Proceedings. Loughborough University.*

Denton, P. (2001). Generating Coursework Feedback for Large Groups of Students Using MS Excel and MS Word. *U.Chem.Ed, 5*, 1-8

Denton, P. (2003). Returning Feedback to Students via Email Using Electronic Feedback 9. 2(1): Assessment, The Learning and Teaching Unit, Manchester Metropolitan University. Available: http://www.ltu.mmu.ac.uk/ltia/issue4/denton.pdf [Accessed on 8 July 2003].

Deterline, W. A. (1962). *An Introduction to Programmed Instruction.* New York: Prentice-Hall.

Dick, W., & Carey, L. (1990). *The Systematic Design of Instruction, 3rd ed.*New York: Harper Collins.

Doig, S. M. (1999). Developing an Understanding of the Role of Feedback in Education. Griffith University. Available: http://www.tedi.uq.edu.au/TEN/TEN_previous/TEN2_99/ten2_doig.html [Accessed on 25 April 2003].

Doyle, W. (1986). Classroom organization and management. In M. C. Wittrock (Ed.), *Handbook of Research on Teaching* (3 ed.). New York: Macmillan.

Draper, S. W. (1999). Feedback. Available: http://www.psy.gla.ac.uk/~steve/feedback.html [Accessed on 7 March 2003].

du Boulay, B. (2006). Commentary on Kurt VanLehn's "The Behavior of Tutoring Systems". *International Journal of Artificial Intelligence in Education, 16*, 267-270

Eckstein, J., Bergin, J., & Sharp, H. (2002). *Patterns for Feedback.* In Proceedings of the EuroPLoP 2002. Seventh European Conference on Pattern Languages of Programs, Irsee, Germany Available: http://csis.pace.edu/~bergin/patterns/FeedbackPatterns.html [Accessed on 8 September 2003].

Elawar, M. C., & Corno, L. (1985). A factorial experiment in teachers' written feedback on student homework: changing teacher behavior a little rather than a lot. *Journal of Educational Psychology, 77*, 162-173

Ericsson, K. A., & Simon, H. A. (1993). *Protocol analysis: verbal reports as data* (Rev. ed.). Cambridge, Mass: MIT Press.

Ersoy, H. (2001). *Effects of different types of feedback on online self-assessment tests in web based learning environment in distance education* (A final Research Proposal. CEIT 520): Research methods in Computer Education and Instructional Technology, METU.

Fernandes, E., & Kumar, A. N. (2004). *A Tutor on Scope for the Programming Languages Course.* In Proceedings of the ACM SIGCSE Bulletin, the 35th SIGCSE technical symposium on Computer science education SIGCSE '04

Fitz-Gibbon, C. T. (1977). *An analysis of the literature of cross-age tutoring.* Washington, DC: National Institute of Education, (ERIC Document Reproduction Service No. ED 148 807).

Foote, C. J. (1999). Attribution feedback in the elementary classroom. *Journal of Research in Childhood Education, 13*(2), 155-167

Foxley, E., Higgins, C., Hegazy, T., Symeonidis, P., & Tsintsifas, A. (2001). *CourseMaster CBA System: Improvements over Ceilidh.* In Proceedings of the fifth International CAA Conference, Loughborough University

Foxley, E., Higgins, C., Tsintsifas, A., & Symeonidis, P. (1999). *Ceilidh, a System for the Automatic Evaluation of Student Programming Work.* In Proceedings of the 4th International Conference on Computer Based Learning in Science, University of Twente, Holland, 2-6 July, 1999.

Gareis, C. R. (2006). Collaborative Leadership: The Forgotten Art of Formative Assessment. Training and Technical Assistance Center. Available: http://www.wm.edu/ttac/corner/2006febmar.html [Accessed on 12 September].

Gibbs, G. (1992). *Improving the quality of student learning: based on the Improving Student Learning Project funded by the Council for National Academic Awards*: Bristol: Technical and Educational Services.

Gibbs, G., & Habeshaw, T. (1992). *Preparing to teach: An introduction to effective teaching in higher education.* Bristol: Technical and Education Services.

Gibbs, G., & Simpson, C. (2003). Does your assessment support your students' learning? *Journal of Learning and Teaching in Higher Education, 1*(1)

Available: http://www.open.ac.uk/science/fdtl/documents/lit-review.pdf [Accessed on 18 September 2003].

Gibbs, G., & Simpson, C. (2003). *Measuring the response of students to assessment: the Assessment Experience Questionnaire.* In Proceedings of the International Improving Student Learning Symposium, Hinckley, UK

Gibbs, G., Simpson, C., & Macdonald, R. (2003). *Improving student learning through changing assessment - a conceptual and practical framework.* In Proceedings of the European Association for Research into Learning and Instruction, Padova, Italy

Gibbs, I. (1978). *The effectiveness of three methods of teacher training.* Unpublished D.Phil., New University of Ulster.

Graesser, A. C. (1993). *Questioning mechanisms during tutoring, conversation, and human computer interaction*: Memphis State University, Memphis, TN 9ERIC Document Reproduction Service no. TM 020 505.

Graesser, A. C., Person, N., & Magliano, J. (1995). Collaborative dialogue patterns in naturalistic one-on-one tutoring. *Applied Cognitive Psychology, 9*(6), 495-522

Graesser, A. C., Wiemer-Hastings, K., Wiemer-Hastings, P., Kreuz, R., & Tutoring-Research-Group. (1999). AutoTutor: A simulation of a human tutor. *Cognitive Systems Research, 1*(1), 35-51

Guzdial, M. (1995). Software-Realized Scaffolding to Facilitate Programming for Science Learning. *Interactive Learning Environments, 4*(1), 1-44 Available: http://quixotic.cc.gt.atl.ga.us/~dnguyen/papers/ILE4_Guzdial.pdf [Accessed on 6 April 2004].

Hambleton, R. K., & Swaminathan, H. (1985). *Item Response Theory: Principles and Applications*. Boston: Kluwer.

Hammer, D., & Elby, A. (2000). *Epistemological Resources.* In Proceedings of the fourth International Conference of the Learning Sciences Available: http://www.umich.edu/~icls/proceedings/pdf/Hammer.pdf [Accessed on 21 April 2004].

Hancock, T. E., Stock, W. A., & Kulhavy, R. W. (1992). Predicting feedback effects from response-certitude estimates. *Bulletin of the Psychonomic Society, 30*, 173-176

Hargreaves, E., Mc Callum, B., & Gipps, C. (2000). Teacher feedback strategies in primary classrooms -new evidence. In S. Askew (Ed.), *Feedback for Learning*:Routledge Falmer, London.

Heift, T. (1998). *Designed Intelligence: A Language Teacher Model.* Unpublished Ph.D. Dissertation, Simon Fraser University.

Herrington, J., & Oliver, R. (2000). An instructional design framework for authentic learning environments. *Educational Technology Research and Development, 48*(3), 23-48 Available: http://elrond.scam.ecu.edu.au/gcoll/4141/HerringtonETRD.pdf. [Accessed on 15 April 2004].

Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong, & Verplank. (1996). Curricula for Human-Computer Interaction. ACM SIGCHI. Available: http://acm.org/sigchi/cdg/cdg2.html#2_1 [Accessed on 12 September 2006].

Higgins, C., Symeonidis, P., & Tsintsifas, A. (2002). *The Marking System for CourseMaster.* In Proceedings of the 7th annual conference on Innovation and technology in computer science education

Hilem, Y., & Futtersack, M. (1994). *COMPANION: An Interactive Learning Environment Based on the Cognitive Apprenticeship Paradigm for Design Engineers Using Numerical Simulations.* In Proceedings of the Educational Multimedia and Hypermedia, 1994. Proceedings of ED-MEDIA 94World Conference on Educational Multimedia and Hypermedia (Vancouver, British Columbia, Canada, June 25-30, 1994)

Hoffer, B. K., & Pintrich, P. R. (1997). The Development of Epistemological Theories: Beliefs about Knowledge and Knowing. *Review of Eucational Research, 67*(1), 88-140

Hudspeth, D. R. (1993). Feedforward. In J. V. Dempsey & G. C. Sales (Eds.), *Interactive instruction and feedback* (pp. 287-300): Englewood Cliffs, NJ: Educational Technology Publications.

Huitt, W. (1994). Feedback. Available: http://chiron.valdosta.edu/whuitt/files/feedback.html [Accessed on 7 March 2003].

Illingworth, V., & Pyle, I. C. (1996). *A Dictionary of computing*:Oxford University Press.

Jackson, M. (1995). Making the grade: the formative evaluation of essays. UtiliBASE. Available: http://ultibase.rmit.edu.au/Articles/dec96/jacks1.htm [Accessed on 1 October 2006].

Jackson, S. L., Krajcik, J., & Soloway, E. (1998). *The design of guided learner-adaptable scaffolding in interactive learning environments.* In Proceedings of the CHI 98, Los Angeles CA. Available: http://www.si.umich.edu/UMDL/CHI98paper.pdf [Accessed on 27 September 2004].

Jang, S., Kim, Y., & Baek, J. (2001). *A Study on Design of Feedback-Contents in Cyber Learning Environment.* In Proceedings of the 9th International Conference on Computers in Education, (ICCE 2001), Seoul, Korea Available: http://www.icce2001.org/pdf/p09/kr043.pdf [Accessed on 20 March 2003].

Jeong-Im, & Hannafin, M. (1999). Situated Cognition and learning Environments: Roles, Structures, and Implications for design. Available: http://tecfa.unige.ch/staf/staf-e/pellerin/staf15/situacogn.htm [Accessed on 18 May 2004].

Joy, M., Chan, P.-S., & Luck, M. (2000). *Networked Submission and Assessment.* In Proceedings of the 1st Annual Conference of the LTSN, Center for Information and Computer Science, LTSL-ICS Available: http://www.ics.ltsn.ac.uk/pub/conf2000/Papers/mjoy.htm [Accessed on 9 September 2003]

Joy, M., & Luck, M. (1998). The BOSS System for On-line Submission and Assessment, Monitor. *Journal of the CTI Centre for Computing, 10*, 27-29

Kawachi, P. (2002). Intelligent feedback initiating the intrinsic motivations. *Staff and education development international, 6*(3), 233-243

Kerka, S. (1999). Self-Directed Learning. Washington, DC: Eric Clearinghouse on Adult, Career, and Vocational Education. Available: [Accessed on 19 April 2004].

Kirakowski, J. (2000). Questionnaires in Usability Engineering: A List of Frequently Asked Questions (3rd Ed.). Human Factors Research Group, Cork, Ireland. Available: http://www.ucc.ie/hfrg/resources/qfaq1.html [Accessed on 14 April 2005].

Kluz'niak, F., Szpakowicz, S., & Bien', J. S. (1985). *Prolog for Programmers*:London: Academic press.

Knowles, M. (1988). *The modern practice of adult education: From pedagogy to andragogy*.Wilton, Conn: Association Press.

Knowles, M. (1988). *Self-directed learning: A guide for learners and teachers*.New York: Cambridge Book Co.

Knowles, M. (1990). *The adult Learner: a neglected species* (4th ed.). Houston: Gulf Publishing.

Knowles, M. S., HoltonIII, E. F., & Swanson, R. A. (1998). *The adult learner: the definitive classic in adult education and human resource development* (5th ed.). Houston: Gulf Pub.

Kochakornjarupong, D., Brna, P., & Vickers, P. (2005). *Who Helps the Helper? A situated Scaffolding System from Supporting Less Experienced Feedback Givers.* In Proceedings of the 12th International Conference on Artificial Intelligence in Education, Amsterdam, The Netherlands, 18-22 July 2005

Kulhavy, W. R., & Wager, W. (1993). Feedback in Programmed Instruction: Historical Context and Implications for Practice. In J. V. Dempsey & G. C. Sales (Eds.), *Interactive instruction and feedback* (pp. 3-20): Englewood Cliffs, NJ: Educational Technology Publications.

Kullman, J. (1998). Mentoring and the development of reflective practice: concepts and context. *System, 26*(4), 471-484

Kumar, A. N. (2003). *Model-based generation of demand feedback in a programming tutor.* In Proceedings of the Innovations in Teaching Programming, volume vii of AIED2003 supplemental proceedings Available: http://www.cs.usyd.edu.au/~aied/vol7/vol7_Kumar.pdf

Kumar, V., McCalla, G., & Greer, J. (1999). *Helping the Peer Helper.* In Proceedings of the International Conference on Artificial Intelligence in Education, Le Mans, France Available: http://www.sfu.ca/~vivek/personal/papers/AIED99.pdf [Accessed on 8 April 2005].

Lajoie, S. P. (2000). *Computer as Cognitive Tools: No More Walls*:NJ: Lawrence Erlbaum, Associates.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to Latent Semantic Analysis. *Discourse Processes, 25*, 259-284

Larson, J. R., Jr., Glynn, M. A., Fl eenor, C. P., & Scontrino, M. P. (1986). Exploring the Dimensionality of Managers' Performance Feedback to Subordinates. *Human Relations, 39*, 1083-1102

Laurillard, D. (2002). *Rethinking University Teaching*.London: Routledge Falmer.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*.New York: Cambridge University Press.

Lepper, M. R., Aspinwall, L. G., Mumme, D. L., & Chabay, R. W. (1990). Sef-perception and social-perception processes in tutoring: subtle social control strategies of expert tutors. In J. M. Olson & M. P. Zanna (Eds.), *Self-inference processes: the Ontario Symposium* (pp. 217-237). hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Lepper, M. R., Woolverton, M., Mumme, D. L., & Gurtner, J. L. (1993). Motivational techniques of expert human tutors: lessons for the design of computer-based tutors. In S. P. Lajoie & S. Derry (Eds.), *Computers as cognitive tools*.Hillsdale, NJ: Lawrence Erlbaum Associateds, Inc.

Lewis, C., & Rieman, J. (1994). Task-Centered User Interface Design: A Practical Introduction. Lewis & Rieman 1993, 1994. Available: http://www.hcibib.org/tcuid/ [Accessed on 3 February 2006].

Lhyle, K. G., & Kulhavy, R. W. (1987). Feedback Processing and Error Correction. *Journal of Educational Psychology, 79*(3), 320-322

London, M. (1995). Giving feedback: Source-centered antecedents and consequences of constructive and destructive feedback. *Human Resource Management Review, 5*(3), 159-188

Luckin, R., & du Boulay, B. (1999). Ecolab: The Development and Evaluation of a Vygotskian Design Framework. *International Journal of Artificial Intelligence in Education, 10*, 198-220

Luckin, R., du Boulay, B., Yuill, N., Kerawalla, C., Pearce, D., & Harris, A. (2003). *Using Software Scaffolding to Increase Metacognitive Skills amongst Young Learners.* In Proceedings of the 11th International Conference on Artificial Intelligence in Education. Available: http://www.cs.usyd.edu.au/~aied/vol2/vol2_Luckin.pdf [Accessed on 18 March 2004].

Luckin, R., & Hammerton, L. (2002). Getting to Know Me: Helping Learners Understand Their Own Learning Needs through Metacognitive Scaffolding. In S. A. Cerri & G. Gouardères & F. Paraguaçu (Eds.), *Proceedings of the 6th International Conference, ITS 2002, Biarritz, France and San Sebastian, Spain, June 2-7, 2002/Lecture Notes in Computer Science* (Vol. 2363, pp. 759-771): Springer-Verlag Berlin/Heidelberg.

MacDonald, J. (2000). Student views on careers education and guidance. In S. Askew (Ed.), *Feedback for Learning*:Routledge Falmer, London.

Mansouri, F. Z., Gibbon, C. A., & Higgins, C. A. (1998). *PRAM: Prolog Automatic Marker.* In Proceedings of the Proceedings of ITiCSE'98, Dublin

Markle, S. R. (1964). *Good Frames and Bad.*Wiley: New York.

Marsh, T., & Wright, P. (1999). Co-operative Evaluation of a Desktop Virtual Reality System. In S. Smith & M. Harrison (Eds.), *Workshop on User Centered Design and Implementation of Virtual Environments* (pp. 99-108). King's Manor, University of York.

McArthur, D., Stasz, C., & Zmuidzinas, M. (1990). Tutoring Techniques in algebra. *Cognition and Instruction, 7*, 197-244

McKendree, J. M. (1990). Effective Feedback Content for Tutoring Complex Skills. *Human-Computer Interaction, 5*(4), 381-413

Methaneethorn, J., Vickers, P., & Brna, P. (2004). *Analyzing the influence between learners' motivational characteristics and ILE features during the interaction with an ILE.* In Proceedings of the 7th International Conference on Intelligent Tutoring Systems (Workshop on Social and Emotional Intelligence in Learning Environment), Maceio, Brazil, August 31, 2004

Mizoguchi, R., & Bourdeau, J. (2000). Using Ontological Engineering to Overcome Common AI-ED Problems. *International Journal of Artificial Intelligence in Education, 11*, 107-121

Mohan, M. (1972). *Peer tutoring as a technique for teaching the unmotivated.*Fredonia, NY: State University of New Youk, Teacher Education Research Center (ERIC Document Reproduction Service No. ED 061 154).

Molka-Danielsen, J. (2000). System Success and Failure: Implementation of IT. Available: http://home.himolde.no/~molka/in320/t00c13.html [Accessed on 8 June 2006].

Moreale, E., Whitelock, D., Raw, Y., & Watt, S. (2002). *What Measures do we Need to Build an Electronic Monitoring Tool for Postgraduate Tutor Marked Assignments.* In Proceedings of the 6th Annual CAA Conference, Loughborough University

Moursund, D. G. (2002). Increasing your expertise as a problem solver: Some roles of computers. Eugene, OR: ISTE. Available: http://www.uoregon.edu/~moursund/PSBook1996/chapter-6.htm [Accessed on 25 June 2003].

Muda, M. A. (2000). *The potential of telematic technology in the training of in-service teachers of English Language in Malaysia.* Unpublished Ph.D. Thesis, Brighton.

Nicol, D. J. (2000). Preparation and Support of Part-time Teachers in Higher Education. *Teacher Development, 4*(1), 115-129 [Accessed on 16 May 2005].

Nielsen, J. (1993). *Usability Engineering.* London: Academic Press.

Nielsen, J. (2000). Why You Only Need to Test With 5 Users. Available: http://www.useit.com/alertbox/20000319.html [Accessed on 16 August 2006].

Noonan, J. V. (1984). *Feedback procedures in computer-assisted instruction: Knowledge-of-response knowledge-of-correct-response, process explanation, and second attempts after errors.* Unpublished Ph.D. Thesis, University of Illlinois, Urbana Champaign.

Norman, D. A., & Draper, S. W. (1986). *User Centered System Design: New Perspectives on Human-Computer Interaction.*Hillsdale, New Jersey; London: Lawrence Erlbaum Associates.

Norvell, T. (2001). Programming with style. Available: http://www.engr.mun.ca/~theo/Courses/ds/STYLE.HTM [Accessed on 1 April 2004].

Page, E. B., Poggio, J. P., & Keith, T. Z. (1997). *Computer Analysis of Student Essays: Finding trait differences.* In Proceedings of the the student profile in AERA/NCME Symposium on Grading Essays by Computer

Pardoe, J., & Vickers, P. (1994). *Using a Prototype Program Assessment Tool.* In Proceedings of the 2nd All-Ireland Conference on the Teaching of Computing, Dublin, 5-7 September 1994 Available: http://www.ulster.ac.uk/cticomp/paradoc.html

Penny, A. J., Harley, K. L., & Jessop, T. S. (1996). Towards a language of possibility: critical reflection and mentorship in initial teacher education. *Teachers and Training: Theory and Practice, 2*(1), 57-69

Pountain, D. (2001). *The new Penguin dictionary of computing.*London: Penguin.

Race, P. (2001). *The Lecturer's Toolkit: A Practical Guide to Learning, Teaching and Assessment*:Kogan Page.

Ramsden, P. (1992). *Assessing for understanding, Learning to teach in higher education*.London: Loutledge.

Randoll, S., & Kali, Y. (2002). Design principles for the use of scaffolds. Center for the Study of Critical Transitions (Spring 2002 Seminar: Design Principles & Cases). Available: http://kie.berkeley.edu/transitions/scaffold_principles.html [Accessed on 30 June 2004].

Rawles, S., Joy, M., & Evans, M. (2002). *Computer-Assisted Assessment in Computer Science: Issues and Software, research report 387* (html 387). Coventry: Department of Computer Science, University of Warwick.

Recker, M. M., & Pirolli, P. (1992). *Student Strategies for Learning Programming from a Computational Environment.* In Proceedings of the 2nd International Conference on Intelligent Tutoring Systems, ITS'92

Rehwinkel, S. (2003). Communication. Available: http://garnet.acns.fsu.edu/~syr5527/communication.htm [Accessed on 13 March 2003].

Renkl, A., & Atkinson, R. K. (2002). Learning From Examples: Fostering Self-Explanations in Computer-Based Learning Environments. *Interactive Learning Environments, 10*(2), 105-119

Rieman, J., Franzke, M., & Redmiles, D. (1995). *Usability Evaluation with the Cognitive Walkthrough.* In Proceedings of the CHI'95 MOSAIC OF CREATIVITY Available: http://delivery.acm.org/10.1145/230000/223735/p387-rieman.pdf?key1=223735&key2=2667859311&coll=GUIDE&dl=ACM&CFID=68133214&CFTOKEN=89415196

Rimmer, A., Pardoe, J., & Vickers, P. (1995). Interactive Program Assessment Using SPROUT. In S. Alexander & P. Magee (Eds.), *the 3rd Annual Conference on the Teaching of Computing: Computing Curriculum Development and Delivery, 29 August-1 September, 1995, Dublin, Ireland* (pp. 285-295): Centre for Teaching Computing.

Robson, C. (2002). *Real world research* (Vol. 2nd Ed). Oxford: Blackwell.

Roehler, L. R., & Cantlon, D. J. (1996). Scaffolding: A Powerful Tool in Social Constructivist Classrooms. Available: http://ed-webs.educ.msu.edu/Literacy/pdf/Scaffolding2.pdf [Accessed on 30 June 2004].

Ross, S. M., & Morrison, G. R. (1993). Using feedback to adapt instruction for individuals. In J. V. Dempsey & G. C. Sales (Eds.), *Interactive instruction and feedback* (pp. 177-196): Englewood Cliffs, NJ: Educational Technology Publications.

Rudner, L. M. (2002). Computer Grading using Bayesian Networks- Overview. Available: http://edres.org/betsy/ [Accessed on 12 December 2003].

Sadler, D. R. (1998). Formative assessment: revisiting the territory. *Assessment in Education, 5*(1), 77-84

Saitio, H. (1994). Teachers' practices and students' preferences for feedback on second language writing: A case study of ELS adult learners. *TESL Canada Journal, 11*(2), 46-70.

Sales, G. C. (1993). Adapted and Adaptive Feedback in Technology-Based Instruction. In J. V. Dempsey & G. C. Sales (Eds.), *Interactive instruction*

and feedback (pp. 159-176): Englewood Cliffs, NJ: Educational Technology Publications.

Salter, G. (2002). *Strategies for the Use of Synchronous Computer-Mediated Communication in Education.* In Proceedings of the International Conference on Computers in Education (ICCE'02), Auckland, New Zealand [Accessed on 07-11-03].

Sapir, M. (1999). The WebTester and the Linear Algebra WebNotes. *ALN Magazine, 3.*

Schön, D. A. (1983). *The reflective practitioner: how professionals think in action*:Aldershot:Avebury.

Schwarez, R. M. (1969). *Towards a Computational Formalization of Natural Language Semantics.* In Proceedings of the Conference On Computational Linguistics, Sånga-Säby Available: http://www.nodali.sics.se/bibliotek/kval/coling69/COLI69-29/COLI69-29.txt [Accessed on 2 Febuary 2004]

Schworm, S., & Renkl, A. (Eds.). (2002). *Learning by solved example problems: Instructional explanations reduce self-explanation activity.*:(pp. 816-821). Mahwah, NJ: Erlbaum.

Shapiro, E. Y. (1983). *Algorithmic Program Debugging*:MIT Press.

Skinner, B. F. (1958). Teaching machines. *Science, 128*, 969-977

Slavin, R. E. (1987). Making Chapter 1 make a difference. *Phi Delta Kappan, 69*(2), 110-119

Smith, S. P. (1997). *Developing an authoring environment for procedural task tutoring systems.* Unpublished Ph.D. Thesis, Massey University, New Zealand.

Sommerville, I. (2001). *Software engineering*.Harlow: Addison-Wesley.

Stein, D. (1998). Situated learning in adult education (Digest #195). Columbus, OH: ERIC Clearinghouse: Adult, Career, and Vocational Education. Available: http://www.ericfacility.net/ericdigests/ed418250.html [Accessed on 19 April 2004].

Steinberg, E. R. (1991). *Computer-assisted instruction: a synthesis of theory, practice, and technology*:NJ:L Erlbaum Associates.

Sterling, L., & Shapiro, E. (1986). *The art of Prolog:advanced programming techniques*.Cambridge, Mass.;London: MIT Press.

Sugrue, B. (2000). Cognitive Approaches to Web-Based Instruciton. In S. P. Lajoie (Ed.), *Computer as Cognitive Tools: No More Walls*:NJ: Lawrence Erlbaum, Associates.

Suhonen, J., Sutinen, E., & Higgins, C. (2001). *Model for a Semi-Automatic Assessment Tool in Web-Based Learning Environment.* In Proceedings of the 9th International Conference on Computers in Education (ICCE/SchoolNet2001), Seoul, Korea, November 12-15, 2001

Thomas, P. (1998). Assignment Submission in a Distance Education Environment. Available: http://www.ulst.ac.uk/cticomp/Thomas.html [Accessed on 20 October 2003].

Thomas, S., Smees, R., & Elliot, K. (2000). Value added feedback for purpose of school self-evaluation. In S. Askew (Ed.), *Feedback for Learning*.London: Routledge Falmer.

Totterdell, P. A., Norman, M. A., & Browne, D. P. (1987). *Levels of adativity in interace design.* In Proceedings of the Interact '87

Tsintsifas, A. (2002). *A Framework for the Computuer Based Assessment of Diagram Based Coursework.* Unpublished PhD Thesis, University of Nottingham.

Tunstall, P., & Gipps, C. (1996). Teacher feedback to young children in formative assessment: A typology. *British Educational Research Journal, 22*, 389-404

VanLehn, K. (1996). *Conceptual and meta learning during coached problem solving.* In Proceedings of the 3rd International Conference on Intelligent Tutoring System ITS' 9

VanLehn, K. (2006). The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education, 16*, xxx-xxx Available: http://www.cs.pitt.edu/~mosse/courses/cs2001/SimpleIntroToITS20.pdf [Accessed on 2 June 2006].

VanLehn, K., Siler, S., Murray, C., Yamauchi, T., & Bagget, W. B. (2003). Human tutoring: why do only some events cause learning? *Cognition and Instruction, 21*(3), 209-249 Available: http://www.isp.pitt.edu/~chas/Papers/03CI_KVL_SS_CM_TY_WBB.pdf [Accessed on 1 October 2006].

Vygotsky, L. S. (Ed.). (1978). *Mind in society: the development of higher psychological processes*:(M. Cole, V. John-Steiner, S. Scribner, E. Souberman, Trans.). Cambridge, MA: Harvard University press.

Wager, W., & Mory, E. H. (1993). The Role of Questions in Learning. In J. V. Dempsey & G. C. Sales (Eds.) (pp. 55-74). New Jersey: Educational Technology Publications Englewwood Cliffs.

Wasik, B. A., & Slavin, R. E. (1993). Preventing early reading failure with one-to-one tutoring: A review of five programs. *Reading Research Quarterly, 28*, 178-200

Watt, D. A. (1990). *Programming language concepts and paradigms*.London: Prentice-Hall International.

Watt, D. A. (1991). *Programming language syntax and semantics*.New York;London: Prentice-Hall.

White, B. Y., Shimoda, T. A., & Frederiksen, J. R. (1999). Enabling Students to Construct Theories of Collaborative Inquiry and Reflective Learning: Computer Support for Metacognitive Development. *International Journal of Artificial Intelligence in Education, 10*, 151-182

Winograd, T., & Flores, F. M. (1986). *Understanding Computers and Cognition, A New Foundation for Design*.Norwood, New York: Ablex Publishing Corporation.

Wood, D. (2001). Scaffolding, contingent tutoring and computer-supported learning. *International Journal of Artificial Intelligence in Education, 12*, 180-292 Available: http://computing.unn.ac.uk/staff/cgpb4/ijaied/members01/archive/vol_12/wood/full.html [Accessed on 18 March 2004].

Wood, D., Bruner, J., & Ross, G. (1976). The role of tutoring and problem solving. *Journal of Child Psychology and Psychiatry, 17*, 89-100

Wood, D., Shadbolt, N., Reichgelt, H., Wood, H., & Paskiewicz, T. (1992). EXPLAIN: Experiments in Planning and Instruction. *Society for the Study of Artificial Intelligence and Simulation of Behaviour Quarterly Newsletter, 81*, 13-16

Wood, H. (1999). The Principle of Contingency in Instruction. Available: http://www.psychology.nottingham.ac.uk/staff/Heather.Wood/contingency.html [Accessed on 19 March 2004].

Wood, H., Wood, D., & Cheng, J. (1999). The Development of Contingent Tutoring Systems (CTSs). ESRC Centre for Research in Development, Instruction and Training (CREDIT), School of Psychology, University of Nottingham. Available: http://www.psychology.nottingham.ac.uk/research/credit/projects/contingent_tutoring_systems/main.html [Accessed on 19 March 2004].

Wood, H. A., & Wood, D. J. (1999). Help Seeking, learning and contingent tutoring. *Computers and Education, 33*, 153-169

Wootton, S. (2002). Encouraging learning or measuring failure? *Teaching in Higher Education, 7*(3), 353-357

Yin, R. K. (1994). *Case Study Research: Design and Methods* (2nd ed. Vol. 5). London: sage.

Zohar, T. (2000). *The education of prospective teachers in the use of classroom based alternative assessment tasks.* Unpublished Ph.D. Thesis, Liverpool.

---

[i] In addition there are a number of questions that need to be asked, those are

(1) When the TA obtains knowledge concerning giving quality feedback, implementation of a **fading scaffolding approach in computer-support could allow the TA to learn alone without any support. (- known as adaptive fading).** The conceptual approaches are the same as 4.1 except andragogical approach (Knowles, 1988, 1990). The design & implementation are the same as 4.1 - 4.3. The evaluation is the same as 4.3

(2) Implementation of a scaffolding approach in computer-support could **train the TA to give better quality feedback in the situation of marking programming assignments.** (This is like a real situation.) The conceptual approaches 4.1. The design & implementation are the same as 4.1 - 4.3. The evaluation is the same as 4.3

(3) Implementation of scaffolding approach in computer-support could **help the TA provide quality feedback in a short period of time.** The conceptual approaches are the same as (2). The design & implementation are the same as 4.1 - 4.3. The evaluation is to test Effectiveness of the system, after testing the hypothesis 4.4 by either interview based questionnaire survey (Robson, 2002) or open-ended question (hybrid), and observation.

Appendix A: Survey of how to teach people to give good feedback

A.1 Introduction

We have discussed with three lecturers who have experience on giving good feedback so we call them as expert tutors. Expert A, PhD, is a foreign lecture in Computer Engineering, Prince of Songkhla University, Thailand. Expert B, PhD, is a lecturer in School of Pharmacy and Chemistry, Liverpool John Moores University. Expert C is a lecturer in School of Informatics, Northumbria University in which his lecture associates with teaching programming language. We also discussed with two TAs who were requested to mark students' assignments. TA A is an experience TA- final year PhD student- of C++/C course in, School of Informatics, Northumbria University. TA B is a novice TA –first year PhD student - for Principle of Information Systems Course, Massey University. The further detail will be explored in the following.

A.2 Expert teachers' perspectives in giving feedback

A.2.1 Expert A

Expert A mentioned about half of all learners copy assignment (plagiarism) when he taught prolog lab and gave students three assignments. He gave two small assignments and a big assignment. He pointed out that usually learners who got their assignments back from the tutor did not care about their mistakes. He said "They should see me and ask any questions they did not understand but they did not come to see the tutor." He felt that the main problem is that the students prefer to copy assignments. Mostly, he found the main problems on teaching Prolog and he described the order from high to low from his point of view. Firstly, recursion –needs more time to explain —about 6 hours; secondly, logical variable –about 6 hours; thirdly, choice point; fourthly, backtracking –learners should know recursion before backtracking, fifthly, data structure (e.g. List), and sixthly, predicate (e.g. multi-predicate). He stated that the reason why many teachers teach functional programming is that it has no backtracking and predicate, so it's easy to explain the learner. In prolog, when weak students did not know recursion they would not know logical variable, choice point, backtracking, data structure and predicate then most of them try to copy assignment. He said that when he gave students two prior small assignments, he would give them feedback after marking

assignment to everybody in his class then he gave students' assignments back and tell them all about the most mistakes. Currently, he does not have any TAs – TAs in his perspective are Postgraduate Students- because in his department now, TAs or senior undergraduate students are not good enough to mark assignments. It's hard to train them to do that. It differs from the University of Melbourne in Australia –where he taught for five years. There are PhD students who can mark assignments they can help the qualified teachers. He asserted that good feedback should be two ways: the feedback from the learner and the questions which students ask the tutor about the problem that the tutor taught and did not make clear explains to. Another is from the teacher on marking assignment or explains to each learner in the class. In this manner, he gave assignments to the learner and then explains all of them. However, he found that the main problem between Thai students and foreign tutors is communicating and interacting with each other. He meant that conversation of feedback is important in Thailand because there is much plagiarism that he does not know about. Further, he argued that feedback should depend on culture e.g. the educational system in Melbourne is better than in his current department. He said that after finishing teaching, there will be tutorial for small group in Melbourne University; however, it depends on the module taught for the selective programming or compulsory programming. He pointed out that this depends on curriculum i.e. if prolog is an important course, TAs should be paid. He said that it is an economic way to ensure quality feedback. This should depend on the size of course and importance of the course. Therefore, he concluded that we should have more than one structure of feedback and also we should consider what is the core concept of feedback. In general, feedback is quite a general word. That is how many students understand the core concept and the feedback should relate to core concept and quality feedback. He said that the learner should be told why they have answered wrongly. However, Expert A has never trained TAs to give feedback. He felt that they should be trained to work and give feedback for the main course of big classes.

A.2.2 Expert B

Expert B's publications (Denton, 2001b, 2001a, 2003) are about giving feedback to students assignment during two weeks after students submission via feedback report in which the kind of feedback can be seen from his publications (e.g. prepare feedback message according to grade criteria; add the learner name; advice how to avoid error in

the future, and encouraging with positive feedback. From his marking, he marks for general error, every error such as tell the learner about how to present data correctly from the LAB report, provide the correct answer directly. He has felt pleasant by using his system to give feedback to the learner, and also his student favour his feedback provision and he also suggest to the other lecturer follow his approach. In addition, he also has ever trained the TAs to give feedback to students by using his system. However, this is just tell the TAs how to use the system but not for how to teach good feedback.

A.2.3 Expert C

When programming assignments are assigned to students, Expert C's marking emphasises analysis of the problem, design and complete the system. He allowed the students to submit their assignment only once and send feedback via email to each student. Even though he has a number of feedback examples for giving to the students, he cannot retrieve such data from his old machine to the new one so we could not describe that how his detailed feedback provided to the students to the researcher. Furthermore, he does not have any TA so he has not trained any TAs to give good feedback.

A.3 Teaching assistants (TAs)' perspectives in giving feedback

A.3.1 TA A

TA A teaches in many LAB sessions and provides oral feedback to each learner (around 100 students) who is conducting the assignment in the LAB (e.g. "How about using 'switch' instead of 'if' "- as indirect feedback). He also achieves the principle of giving feedback (e.g. "fair", "the format is good but I would suggest improve the syntax") from his tutor. From the written feedback, he always indicates to every error and provides the correct answer to the students' assignment; however sometimes, he provides the important error e.g. wrong structure in programming. For the other year students, not the first year students who can submit their assignment only once, can resubmit assignment- because for the first year student they submit assignment only once then the students will receive the a feedback report from the tutor. From his point of view, he does not prefer marking and providing feedback via e-mail because it is inconvenient. Even though he was trained to give feedback by the training program provided by the School, he has still required learn how to give good feedback to the students.

A.3.2 TA B

TA B usually is a TA for the tutorial group work of 30 students and has to mark assignment weekly without any principles of giving feedback from his tutor. From his marking, he indicated every error without the correct answer and never gives important error or any kinds of feedback message to the students because his tutors told the students to submit the assignment only once. He stated that it is inadequate time to give every good feedback to the student because he has never been trained to give good feedback so he require to learn how to provide good feedback to the students in a short period of time. Thus, it seems helpful to have a system help him improve giving good feedback.

A.4 Materials

A.4.1 Scripts for Experts

The questions as below are adapted from Gosse's (2001, p. 20-21) dissertation.

(Programming) Assignment: To All Teachers I would like to meet each of you for one period. When we meet, be ready to discuss the following:

1. What kinds of (programming) assignment do you use with your students? How often? Any preparation?

   <Type of (programming) assignment /Purpose of (programming) assignment>

2. What kinds of feedback do you give the students on this (programming) assignment? What do you hope to achieve? (e.g. Positive feedback, Differentiated feedback, Early warning, etc.) Have you ever taught teaching assistants (TAs) to give such feedback to students? If so, please give the example of those feedbacks.
3. Do your students make a lot of mistakes in (programming) assignment? What kind of mistakes? What makes a good (programming) assignment?
4. What kinds of errors do you feel are very important/not so important? Why?
5. Do you only mark from general errors correction, design problem, and style problem? If not, what else do you give feedback on?
6. Do you mark every mistake?
7. Do you only concentrate on certain kinds of errors?
8. Which editing symbols do you use when you are correcting? e.g.
9. Do you only indicate where the error is or do you provide the correct answer? If you only indicate where the error is, do the students fix their mistakes?
10. Are you confident that you correct work accurately?

11. Are you happy with the type of error correction format you use? Why or why not? (e.g. of type of error correction)
12. How long does it take you to correct students' work? (each assignment for all students)
13. What is your students' response to the type of error correction you provide?
14. What area would you like to focus on or find out more about in the area of error correction (implementation problem/design problem/style problem)? What will/can you try that is new?
15. Do you agree to give any advice how avoid this problem in the future (e.g. identify error, identify how to discriminate answer, identify how to avoid any error in the future).
16. Further, the rest of this sheet is any comment from you.

Focus on the particular assignment (e.g. Prolog/Java assignment)

1. How many programming (Prolog, Java, etc.) assignments that you give to students each taught course?
2. How did you provide feedback to students?
3. Does it depend on their implementation problem, design problem, style problem?
4. Could you please order the important error that you think it should tell the student first –for 5 levels from high to low level?
5. How many times that you give students feedback each assignment before final submission?
6. Have you had TAs to help you marking assignment?
7. If you have TAs, your TAs have been trained to give students feedback on Prolog/Java assignment, haven't they?
8. In your point of view, do you think how quality feedback is?

A.4.2 Scripts for TAs

In our discussion, the TA means the TAs for students either year 1 or 2 and we use the questions below alongside discussion

1. I am a teaching assistant (TA) in Module of

   .....

2. I have to take lab session[**Y / N**]
3. I have to mark assignments **[Y / N]** (assignment type e.g. lab report)
4. How many students do I marks their works each assignment.....
5. How many assignments do I mark in this module.....
6. How often do I mark each assignments (for all students) .....
7. I got the principle of giving feedback to students from teacher(e.g. feedback pattern to help students to improve their learning next time)

   If so, for example,
8. From 7, if not, I have my own principle to give feedback to students (Please indicate the feedback message that you provide to the students)

   For example,
9. If yes, I use these approaches to give feedback to students (Please indicate by circle).

   a. I indicate each error but do not provide the correct answer for each assignment **[always | often | some | a few | never]** and **[I had to provide | the teacher provides me]** the correct answer.

   b. **[I give | the teacher provides me]** feedback to every error for each assignment **[always | often | some | a few | never]** by **[myself | teacher told]**, and I **[strongly agree | agree | N/A | disagree | strongly disagree]** with this approach.

   c. **[I give | the teacher provides me]** feedback to only important errors for each assignment **[always | often | some | a few | never]** (What are important errors? Please indicate the feedback message that you provide to the students) e.g. .....

   d. **[I give | the teacher provides me]** indirect feedback to students (guide students, not give correct answer directly) **[always | often | some | a few | never]** (What are indirect feedback? Please indicate the feedback message that you provide to the students e.g. .....)

   e. **[I give | the teacher provides me]** individual feedback (face-to-face) **[always | often | some | a few | never]** to students.

f. **[I give | the teacher provides me]** feedback via E-mail to each students' assignment (every students) **[always | often | some | a few | never]**

g. I return assignment to each students <u>quickly</u> including **[quality | normal]** feedback **[always | often | some | a few | never]**

h. I return assignment to each students <u>quickly</u> with <u>no</u> feedback **[always | often | some | a few | never]**

i. I return assignment to each students <u>slowly</u> including **[quality | normal]** feedback **[always | often | some | a few | never]**

j. I return assignment to each students <u>slowly</u> with <u>no</u> feedback **[always | often | some | a few | never]**

k. I give feedback for each assignment to students and students resubmit their assignment again **[Y / N]** (if yes, resubmit 1 / 2 / 3 /4 /... times)

l. **[I give | the teacher provides me]** different feedback to each student even though it is the same errors **[always | often | some | a few | never]**

10. I feel bored when I mark assignment and give feedback to students **[strongly agree | agree | N/A | disagree | strongly disagree]**

11. If teacher told me to give feedback in my style to student, I feel difficult to give feedback to students (what problems I found? e.g. I would like students to submit their assignments via computer system. **[Y / N]**)

12. I would like to mark and give feedback to students via computer system. **[Y/ N]**

13. I would like students to return assignment to students via computer system. **[Y / N]**

14. I would like students to resubmit their assignment to me to give feedback to them to improve their understanding/learning again (may be more than 2 times) before final submission.**[Y / N]**

15. Have you been trained to give feedback to students before real marking assignment **[Y / N]**

16. From 15, if so, **[formally | informally]** by which approaches? e.g. ..... (Do you remember what he/she say/help?)

17. Have my teachers prompted/supported me during giving feedback to students? **[always | often | some | a few | never]**

18. From 17, if so, by which approaches? e.g. ..... (Do you remember what he/she say/help?)

19. My feedback to students' assignment is good enough to help them improve their learning (e.g. .....) **[strongly agree | agree | N/A | disagree | strongly disagree]**
20. I'd like to learn more how to give quality feedback to students to improve their learning **[strongly agree | agree | N/A | disagree | strongly disagree]**
21. I'd like to learn more how to give quality feedback to students to improve my giving feedback to students **[strongly agree | agree | N/A | disagree | strongly disagree]**
22. If there are more than one assignment, I give feedback to every assignment (may be some assignment no scores) **[strongly agree | agree | N/A | disagree | strongly disagree]**
23. I have some advice/suggestion for giving quality feedback to students to improve their learning/understanding (any suggestions please provide below)

References

Denton, P. (2001a). Generating and e-mailing Feedback to Students Using MS Office. In M. Danson, and Eabry, C. (Ed.), *the Fifth International CAA Conference Proceedings. Loughborough University.*

Denton, P. (2001b). Generating Coursework Feedback for Large Groups of Students Using MS Excel and MS Word. In M. Danson & C. Eabry (Eds.), *the Fifth International CAA Conference Proceedings*. Loughborough University.

Denton, P. (2003). *Returning Feedback to Students via Email Using Electronic Feedback 9*. 2(1): Assessment, The Learning and Teaching Unit, Manchester Metropolitan Univearsity. Available: http://www.ltu.mmu.ac.uk/ltia/issue4/denton.pdf. [Accessed on 8 July 2003].

Gosse, A., E. (2001). *Error Correction and Feedback Techniques: A Journey of Exploration.* Unpublished Master Dissertations, Aston University.

Appendix B: Preliminary Design and Implementation of McFeSPA (Version 1.1)

B.1 Flowchart for Preliminary Design for Implementation

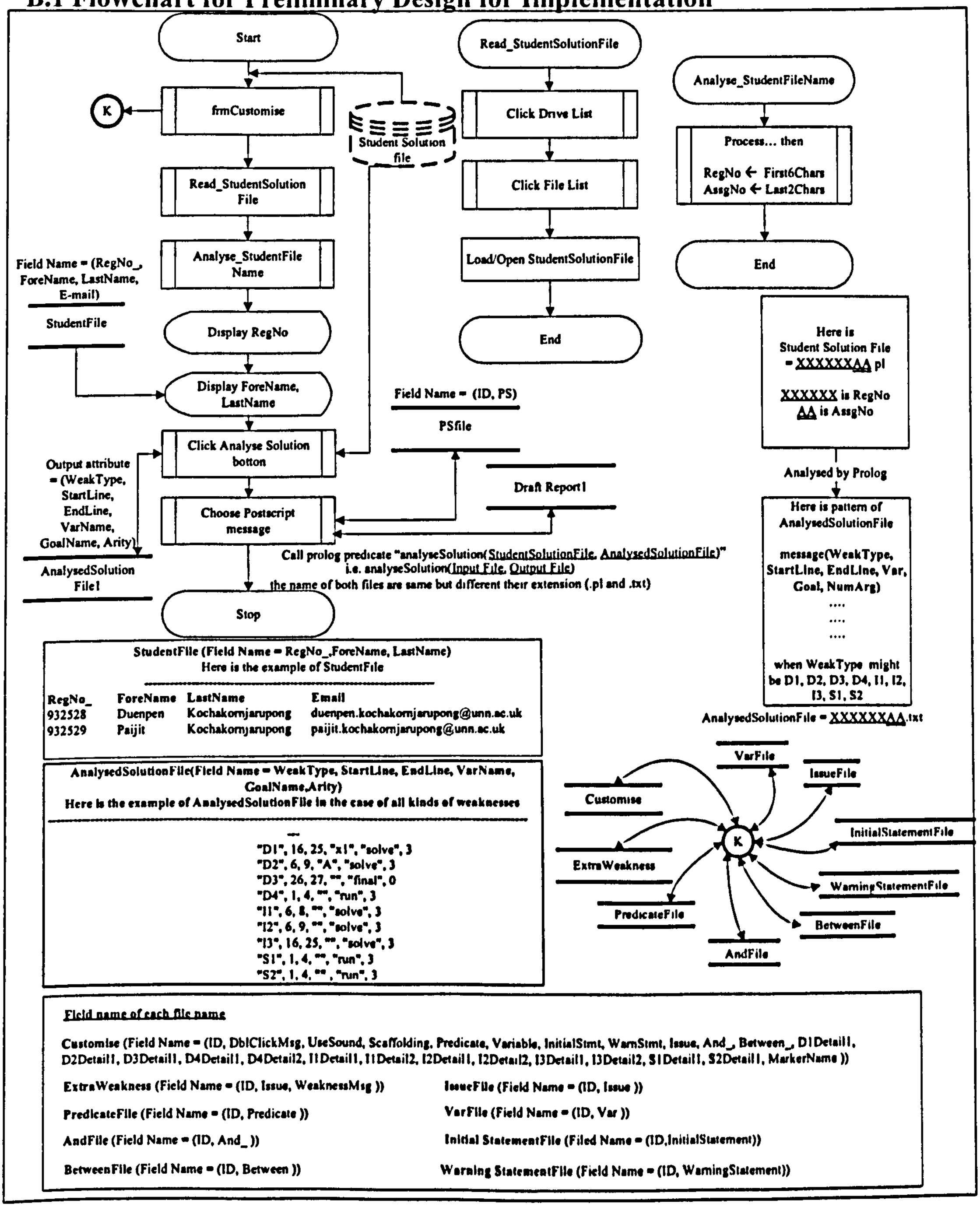


Figure B.1 Flowchart for Design for Implementation (1)

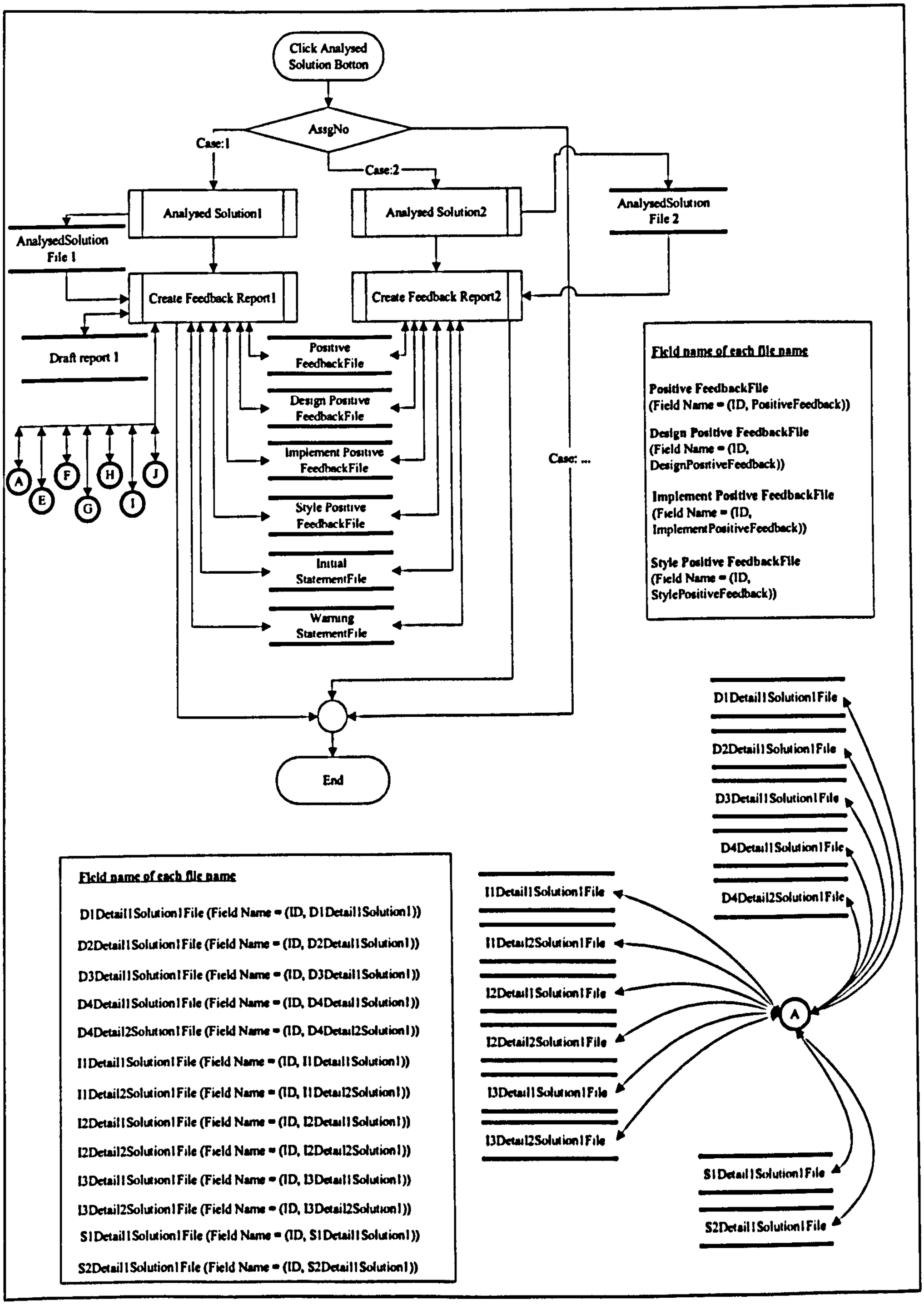
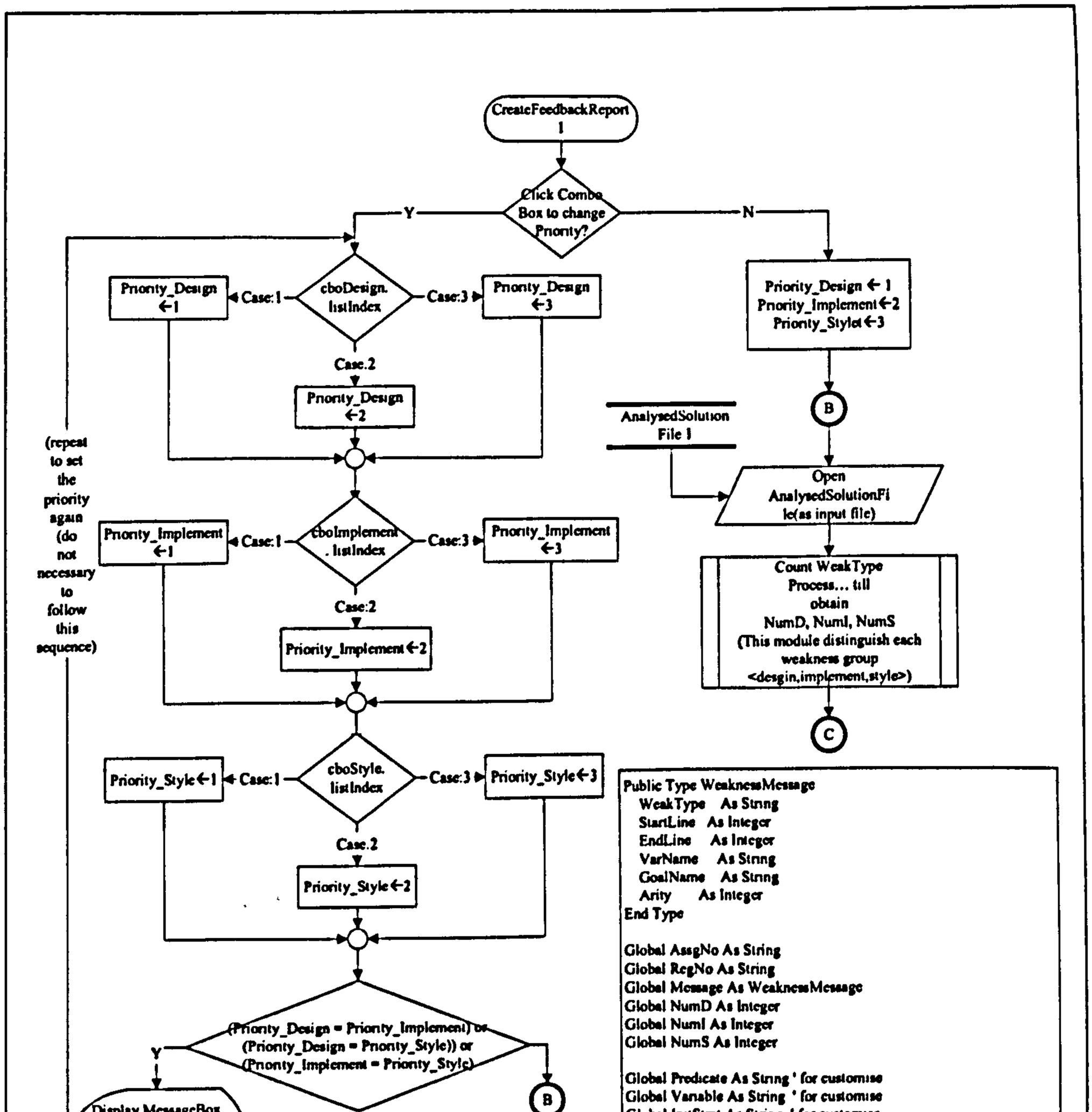


Figure B.2 Flowchart for Design for Implementation (2)




```
Public Type WeaknessMessage
   WeakType   As String
   StartLine  As Integer
   EndLine    As Integer
   VarName    As String
   GoalName   As String
   Arity      As Integer
End Type

Global AssgNo As String
Global RegNo As String
Global Message As WeaknessMessage
Global NumD As Integer
Global NumI As Integer
Global NumS As Integer

Global Predicate As String ' for customise
Global Variable As String  ' for customise
Global InitStmt As String  ' for customise
Global WarnStmt As String  ' for customise
Global Issue    As String  ' for customise
.....
.....
.....

'— for more detail, see from programming Code in Visual Basic
```

Limitation

- Our system support two assignments now; however in the future any software developer can add it till 99 assignments [Max_Assignment = 99]
- For RegNo, we follow the user login of Northumbria University e.g. m912528 only last 6 characters.
- The maximum length of student solution file is 8 characters + dot(.) + extension (2letters = .pl) i.e. XXXXXXAA pl

the difference of StudentSolutionFile and AnalysedSolutionFile is their extension (pl and .txt)

Next Step

- Scaffolding + Synchronous + immediate Feedback (TA system + human TA)

Figure B.3 Flowchart for Design for Implementation (3)

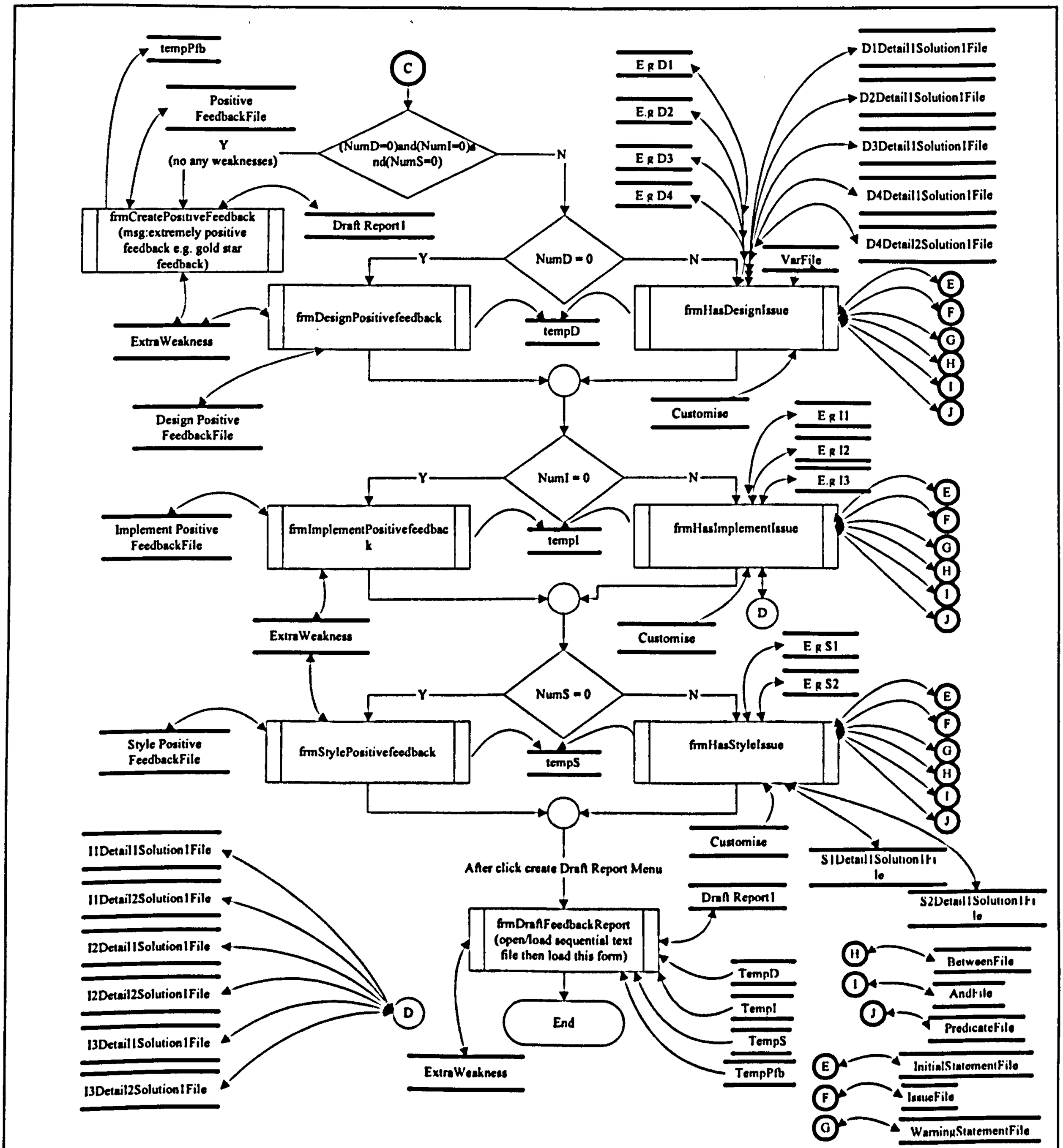


Figure B.4 Flowchart for Design for Implementation (4)

B.2 Preliminary Partial Automated Marking Assignment

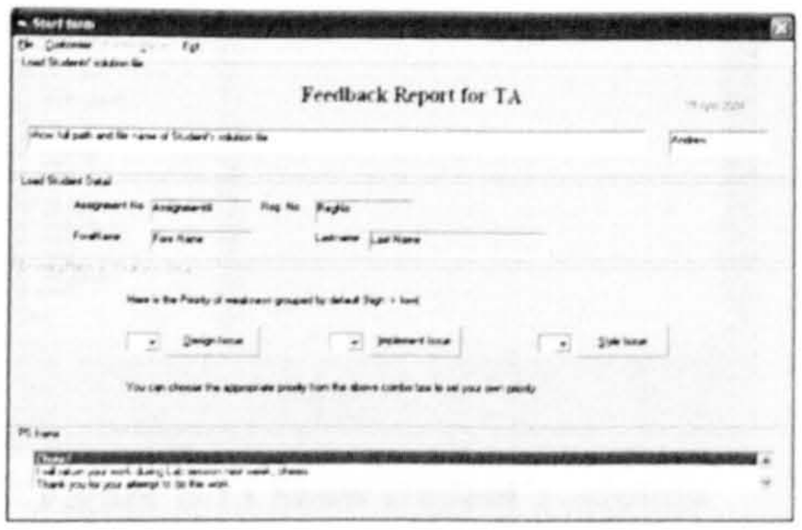


Figure B.5 First window of Feedback System

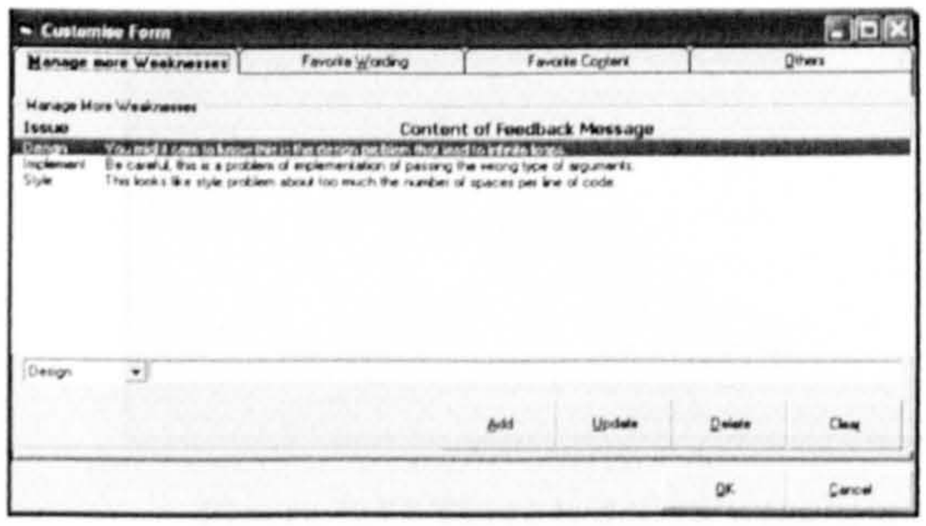


Figure B.6 Customise Menu: Manage more Weaknesses

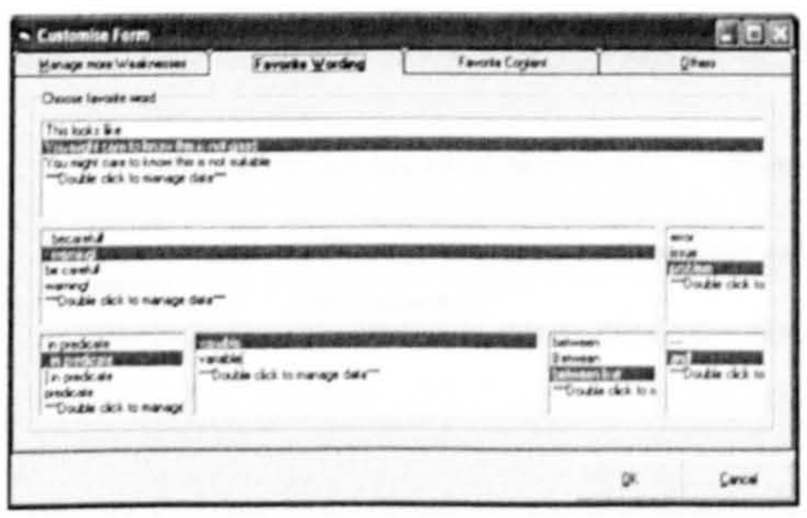


Figure B.7 Customise Menu: Favorite Wording

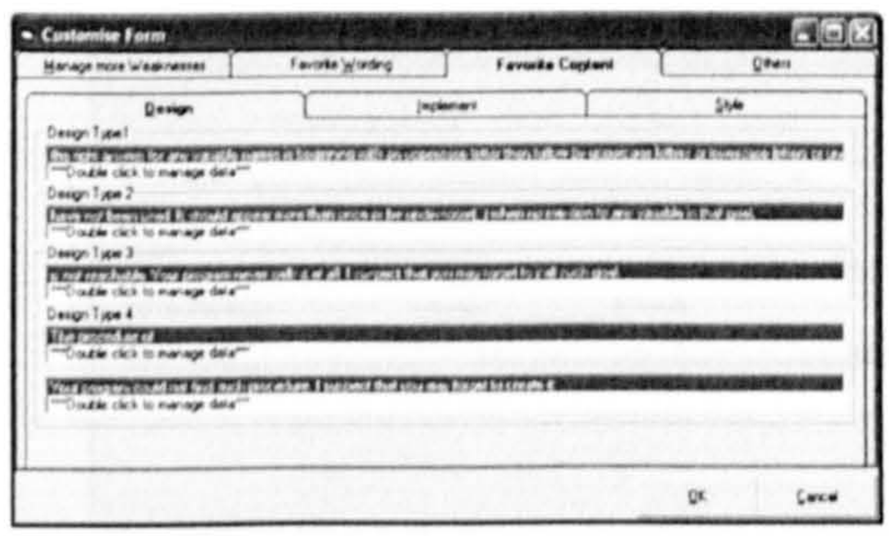


Figure B.8 Customise Menu: Favorite Content (Design)

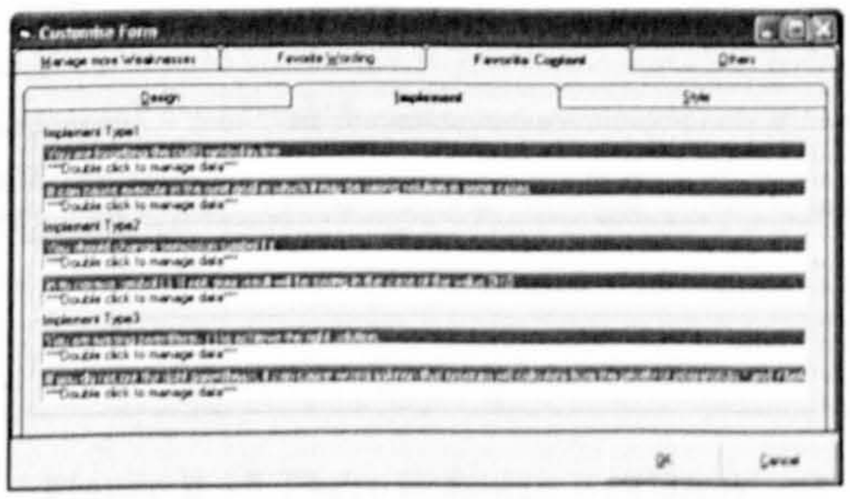


Figure B.9 Customise Menu: Favorite Content (Implement)

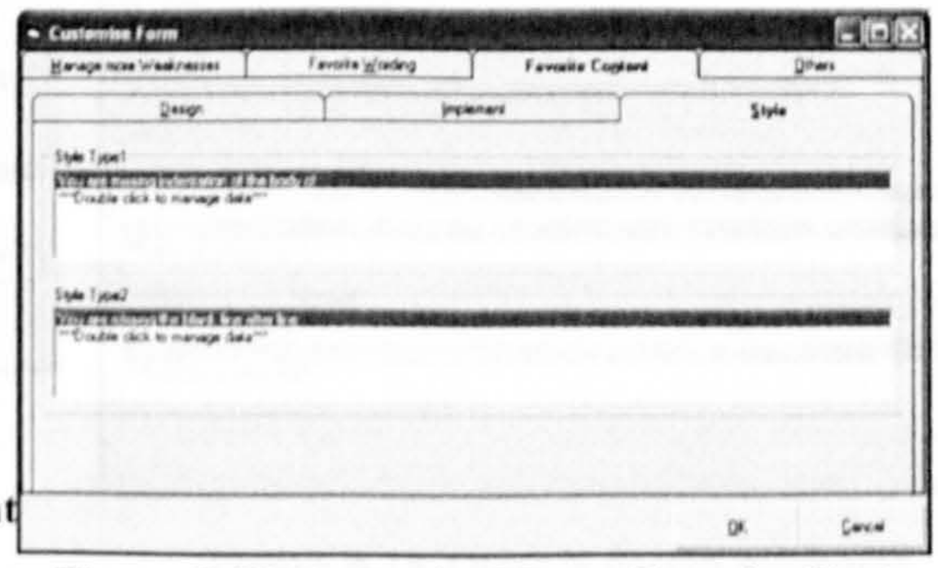


Figure B.10 Customise Menu: Favorite Content (Style)

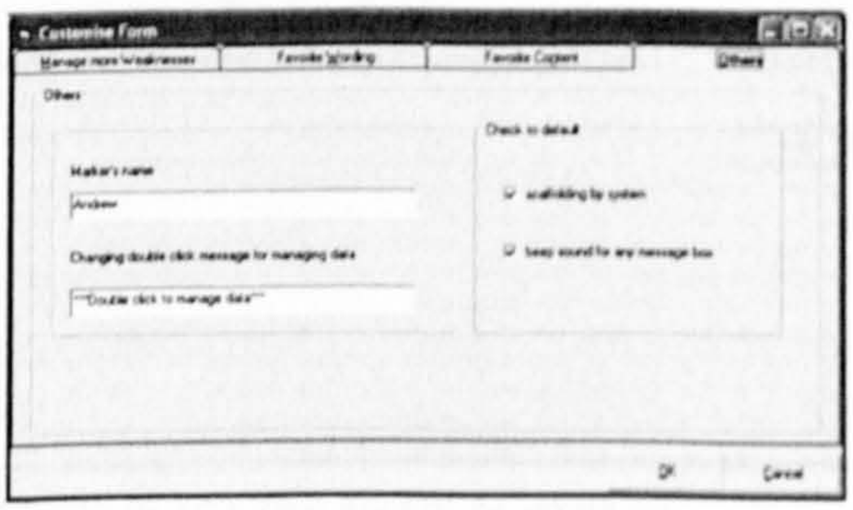


Figure B.11Customise Menu: Others

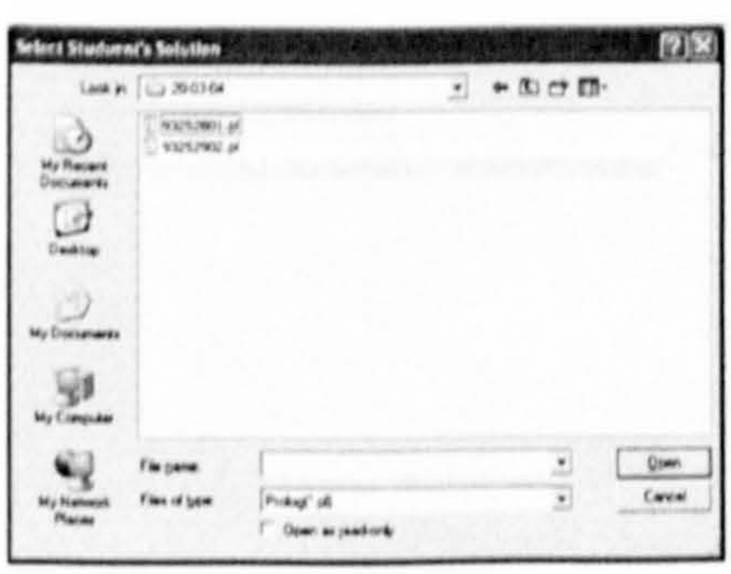


Figure B.12 File Menu (for choosing student's solution)

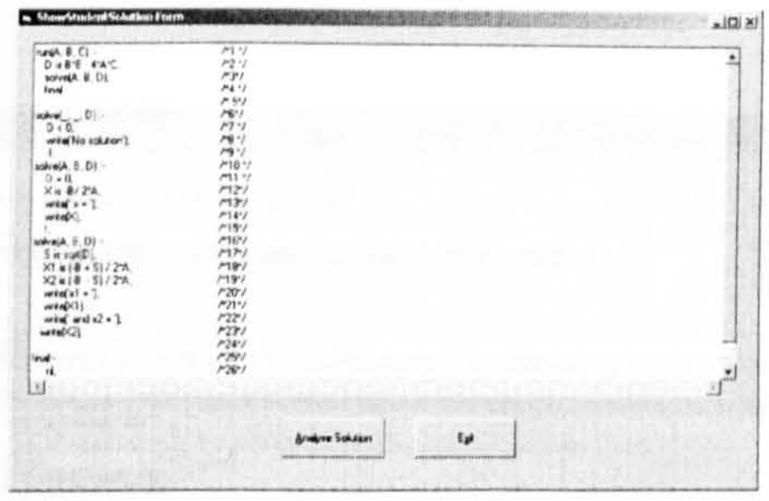


Figure B.13 Show student's solution

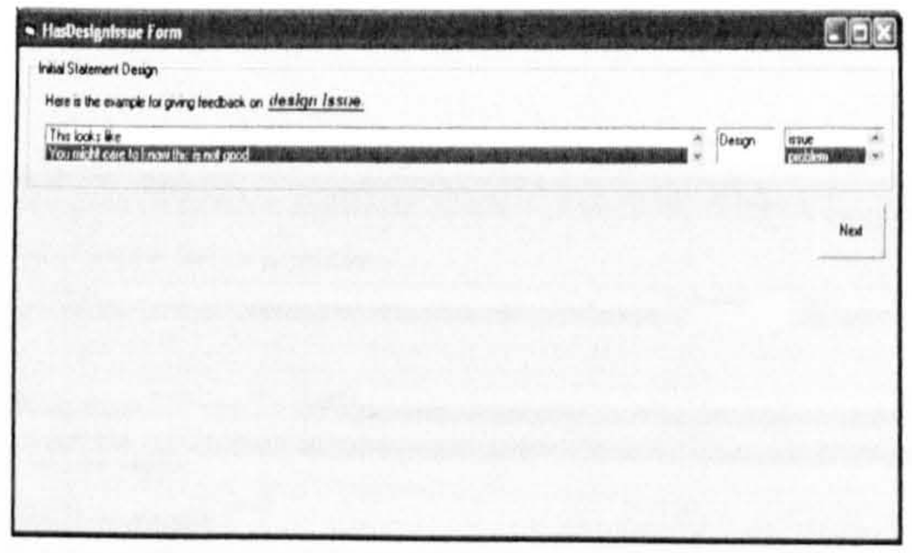


Figure B.14 There is design weakness

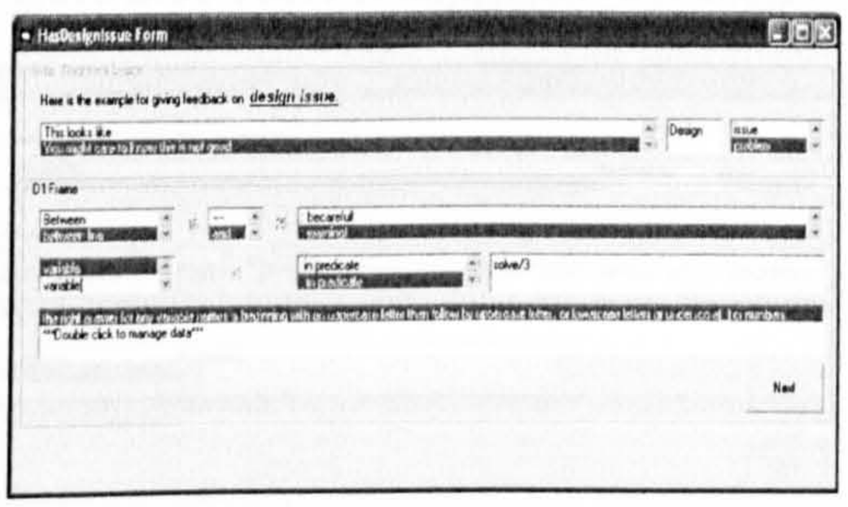


Figure B.15 There is design weakness (Type 1: D1)

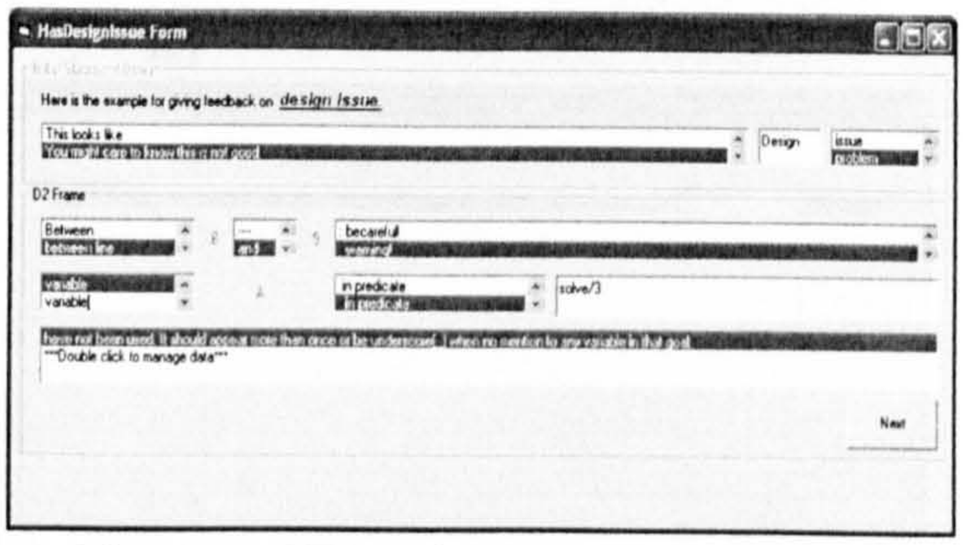


Figure B.16 There is design weakness (Type 2: D2)

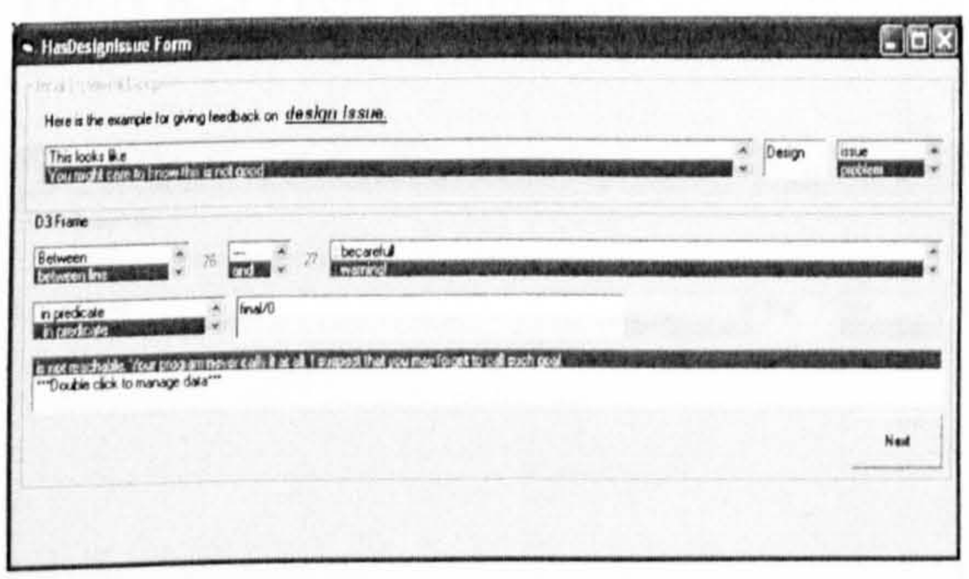


Figure B.17 There is design weakness (Type 3: D3)

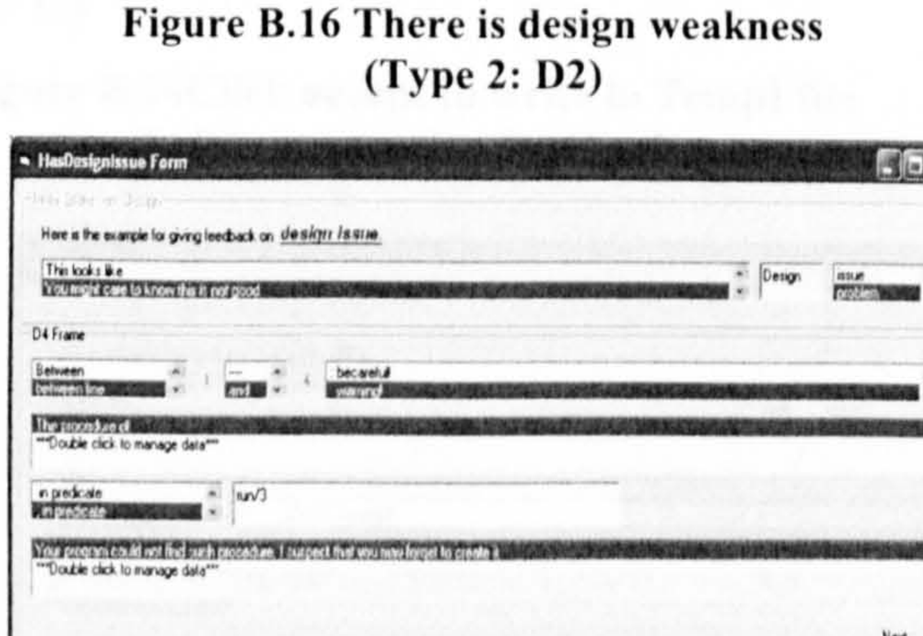


Figure B.18 There is design weakness (Type 4: D4)

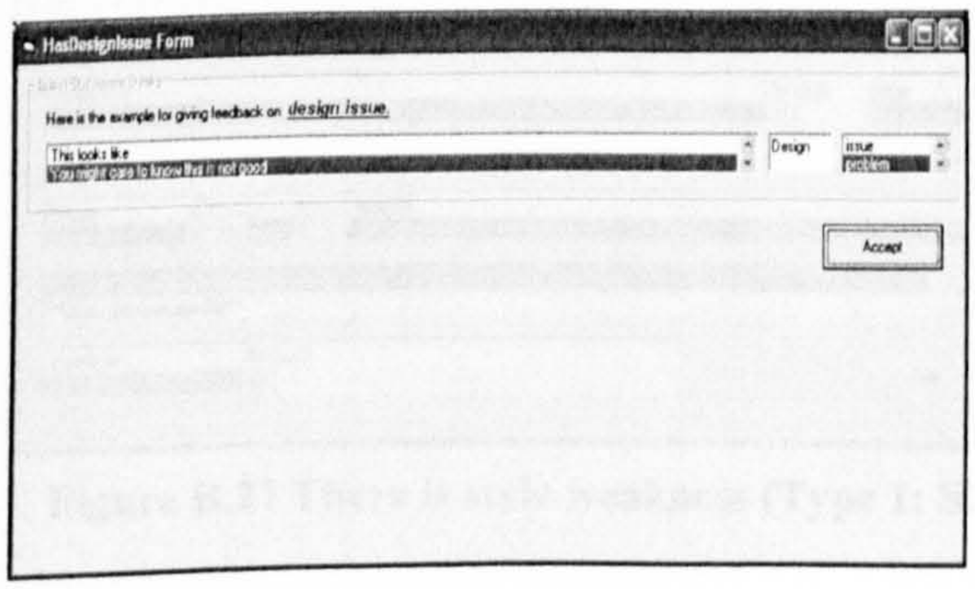


Figure B.19 Click accept to write to TempD file before generating feedback report

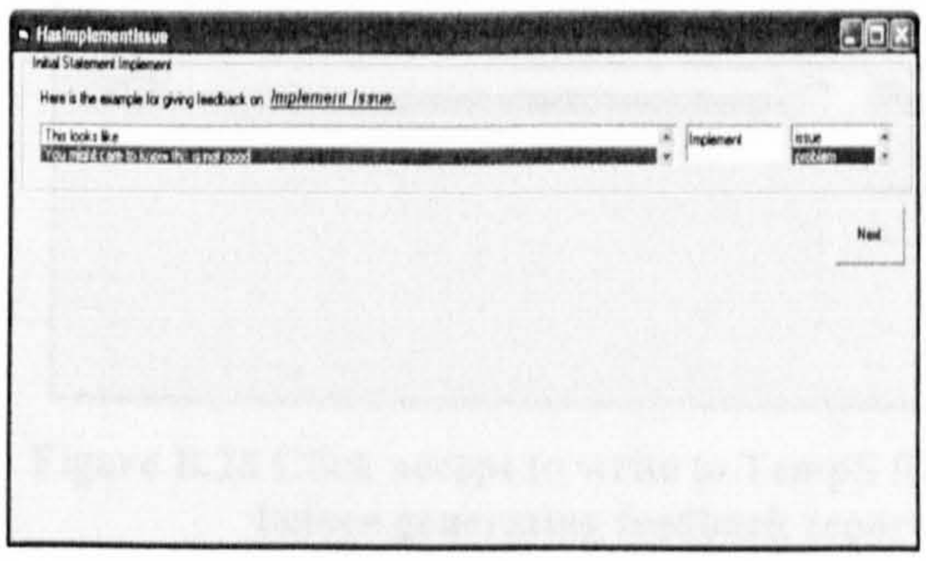
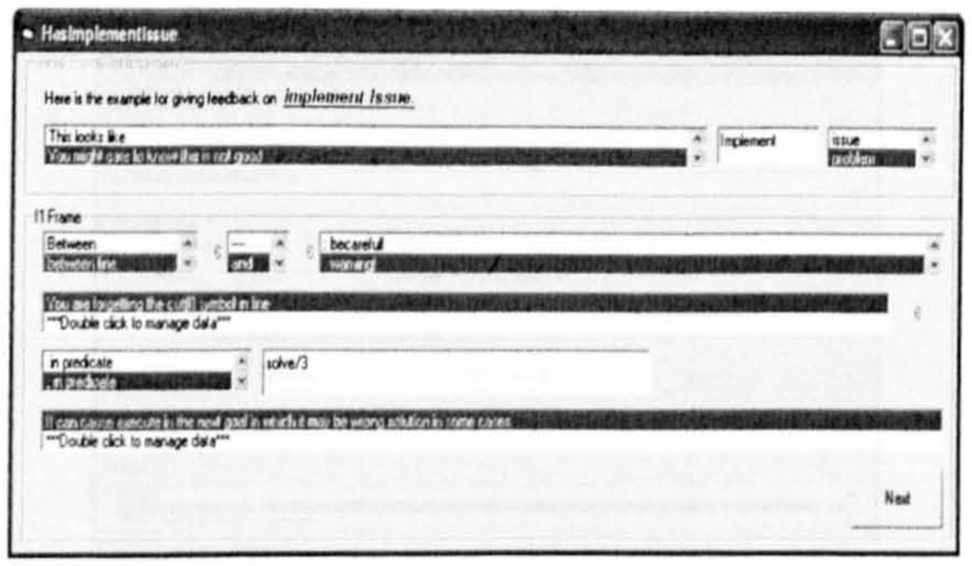


Figure B.20 There is implement weakness



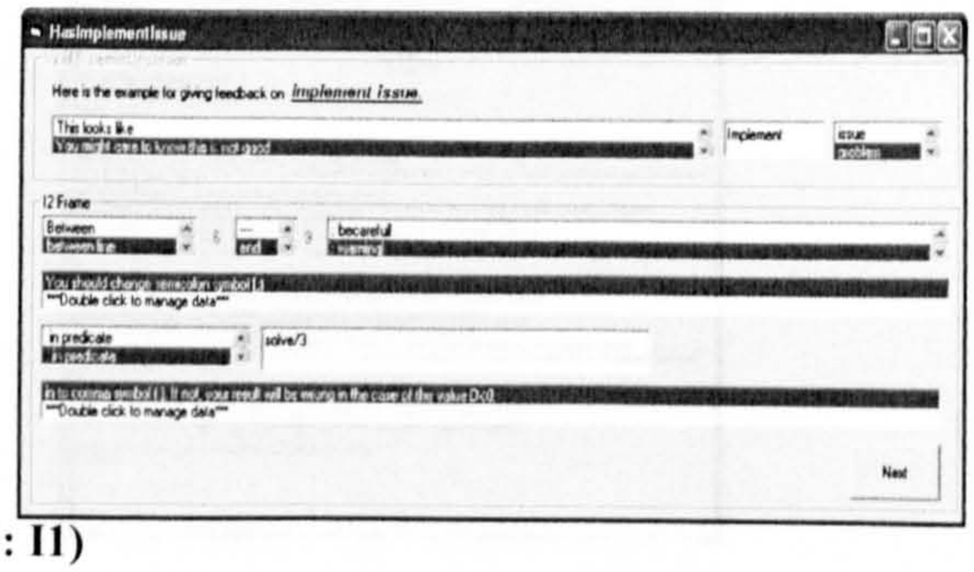


Figure B.21 There is implement weakness (Type 1: I1)

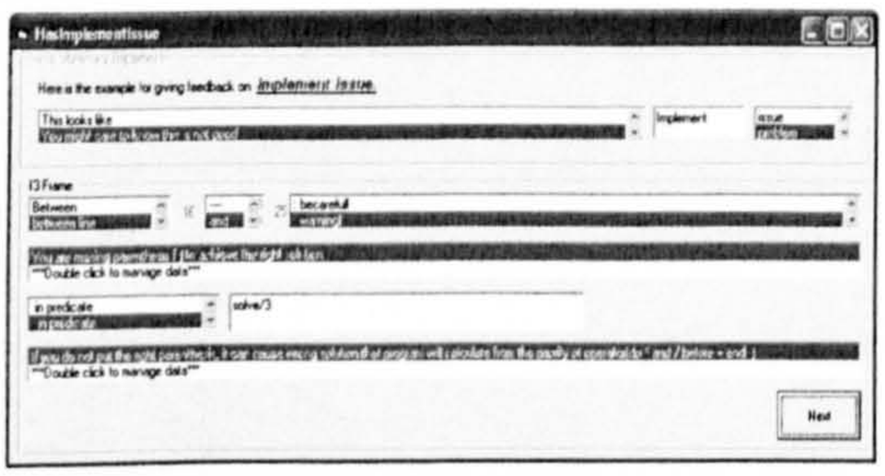


Figure B.22 There is implement weakness (Type 2: I2)

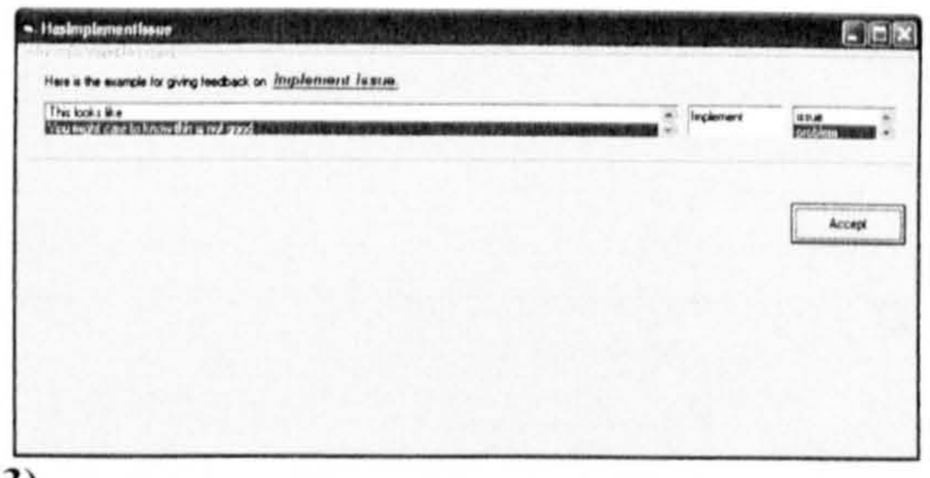


Figure B.23 There is implement weakness (Type 3: I3)

Figure B.24Click accept to write to TempI file before generating feedback report

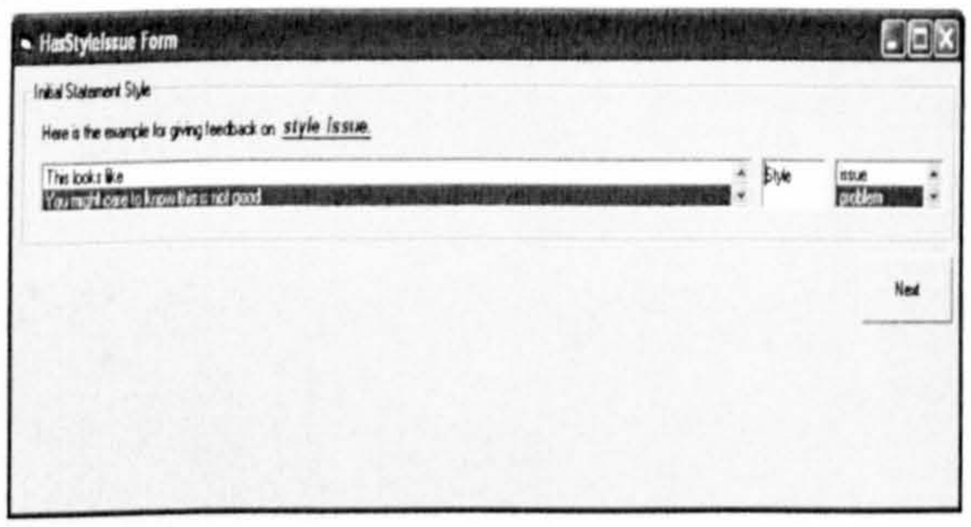


Figure B.25 There is style weakness

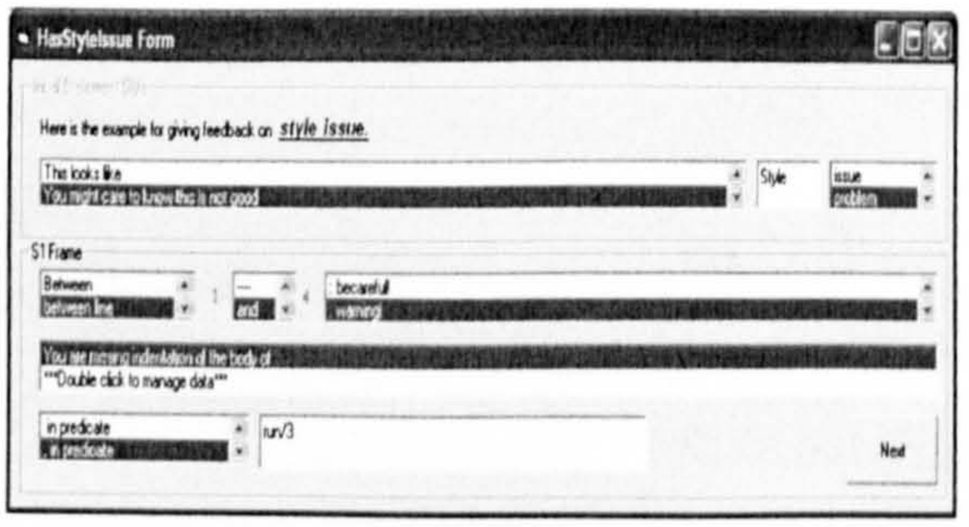


Figure B.26 There is style weakness (Type 1: S1)

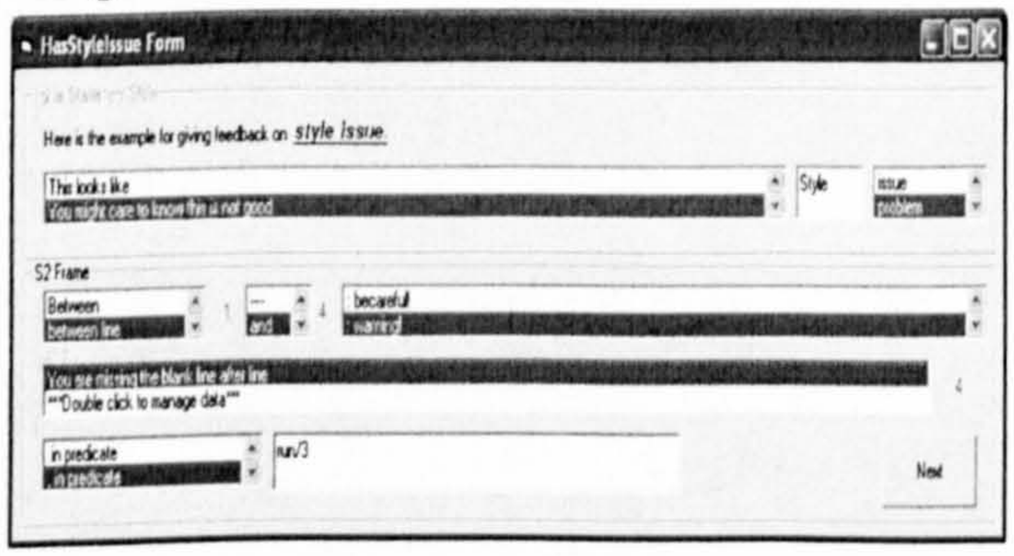


Figure B.27 There is style weakness (Type 1: S2)

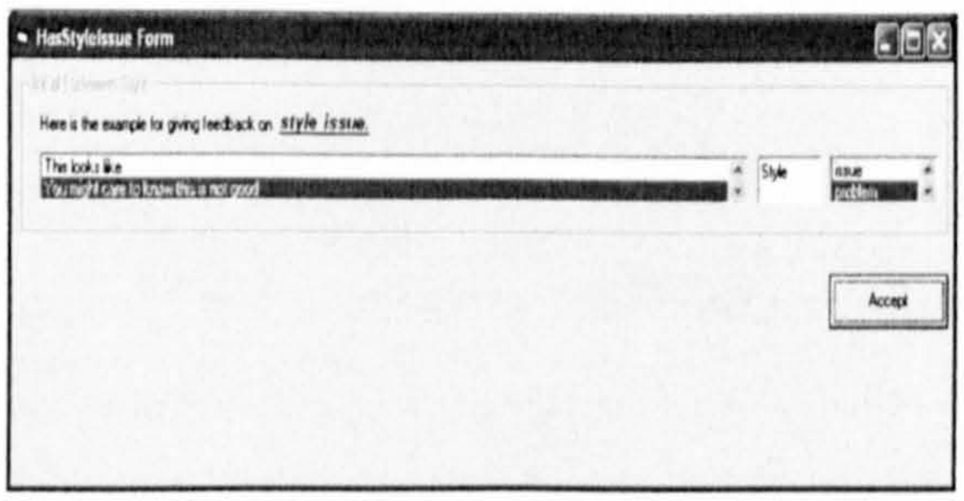


Figure B.28 Click accept to write to TempS file before generating feedback report

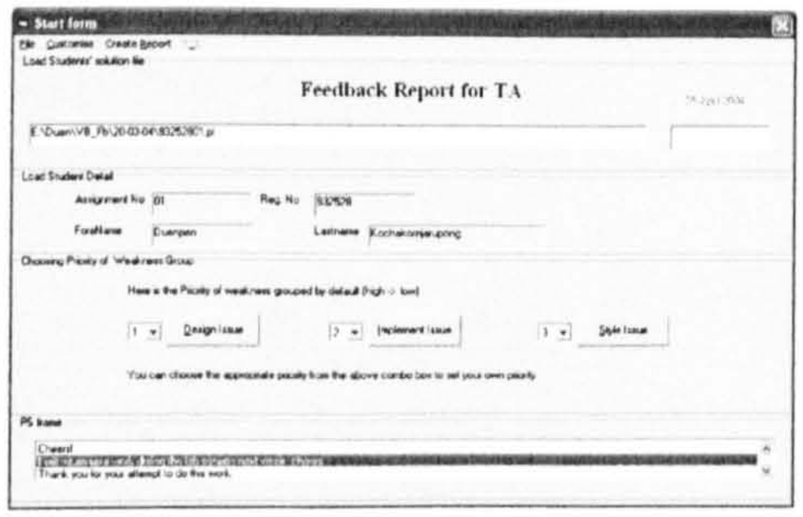


Figure B.29 Choosing Postscript sentence (PS)

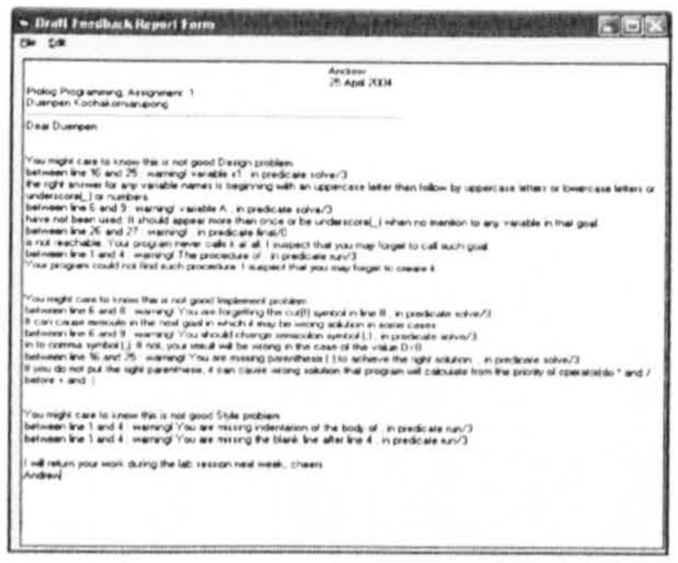


Figure B.30 Generating final feedback report

Appendix C: Design and Implementation of McFeSPA (Version 1.2)

C.1 Flowchart for Design of McFeSPA (Version 1.2)

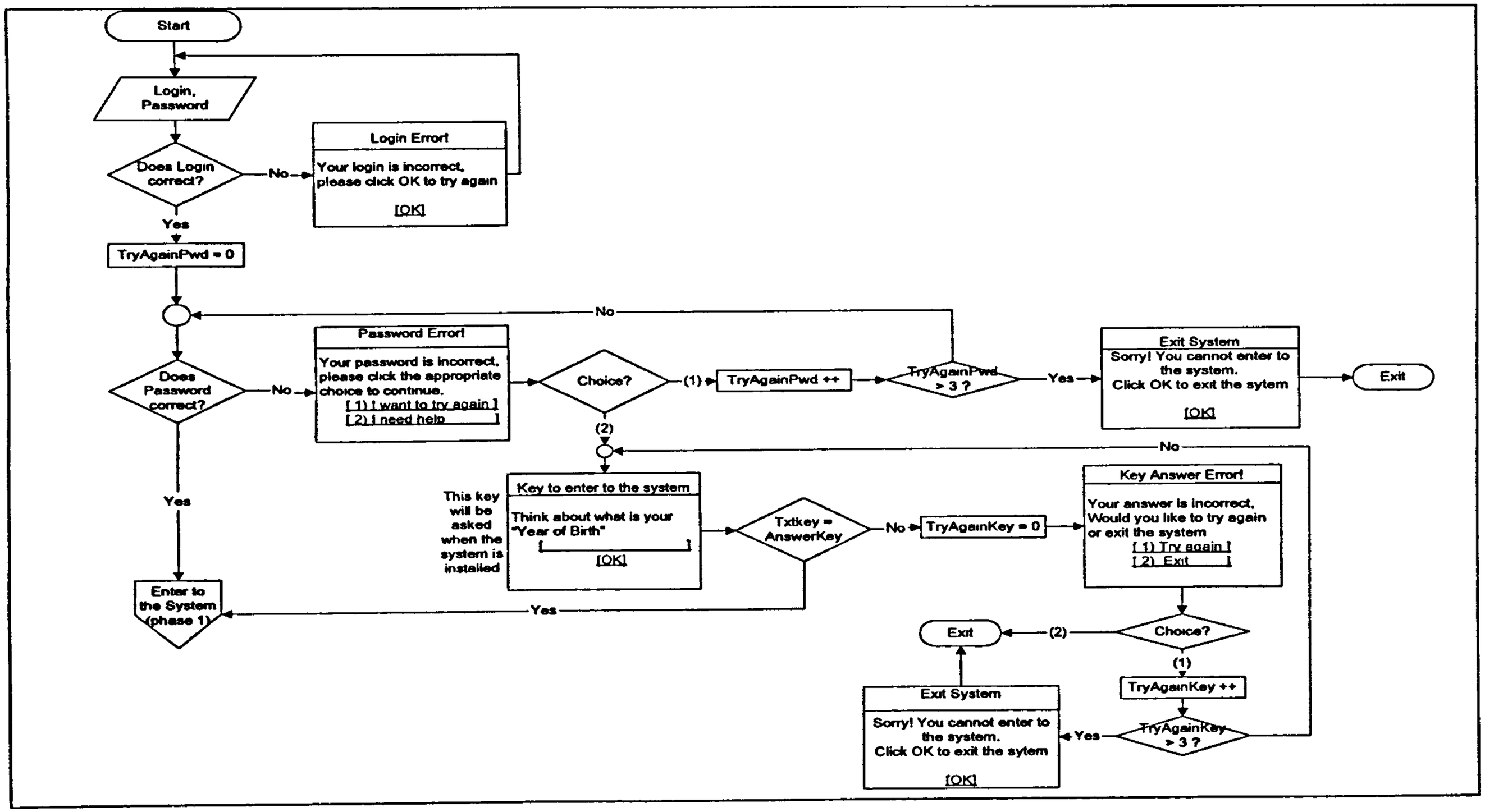


Figure C.1 Design of McFeSPA (Version 1.2): Login

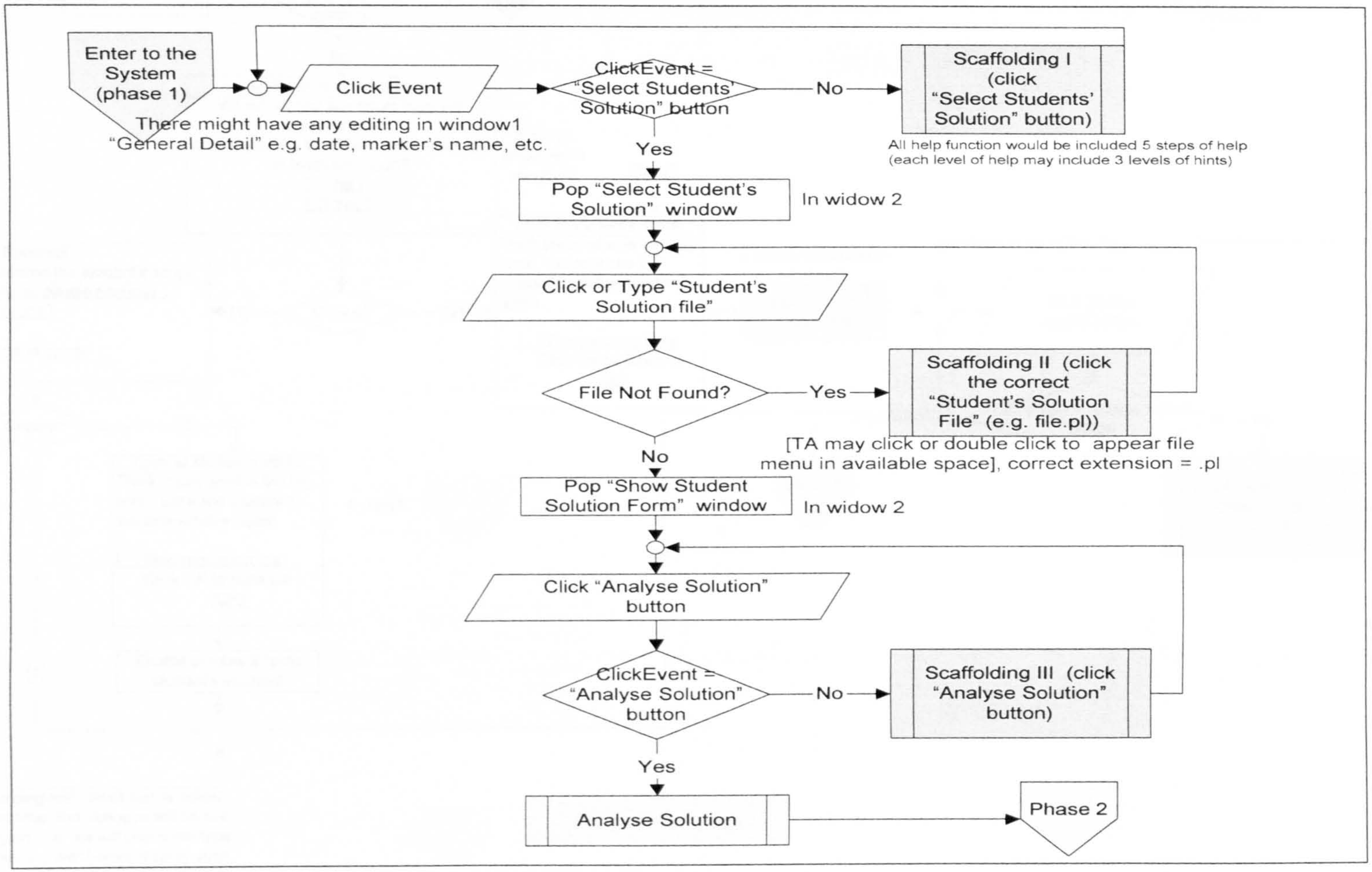


Figure C.2 Design of McFeSPA (Version 1.2): Enter to system

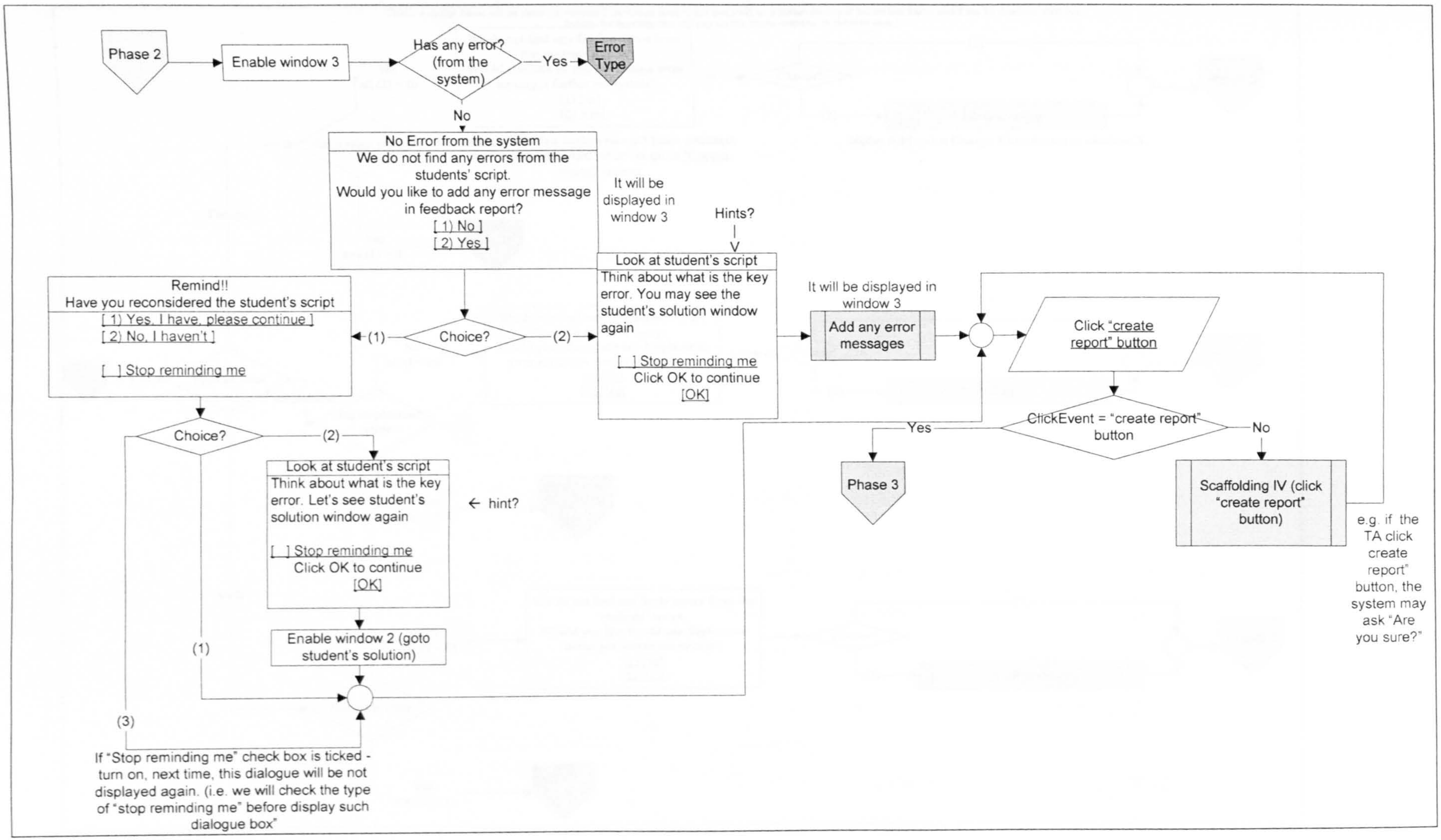


Figure C.3 Design of McFeSPA (Version 1.2): Phase 2

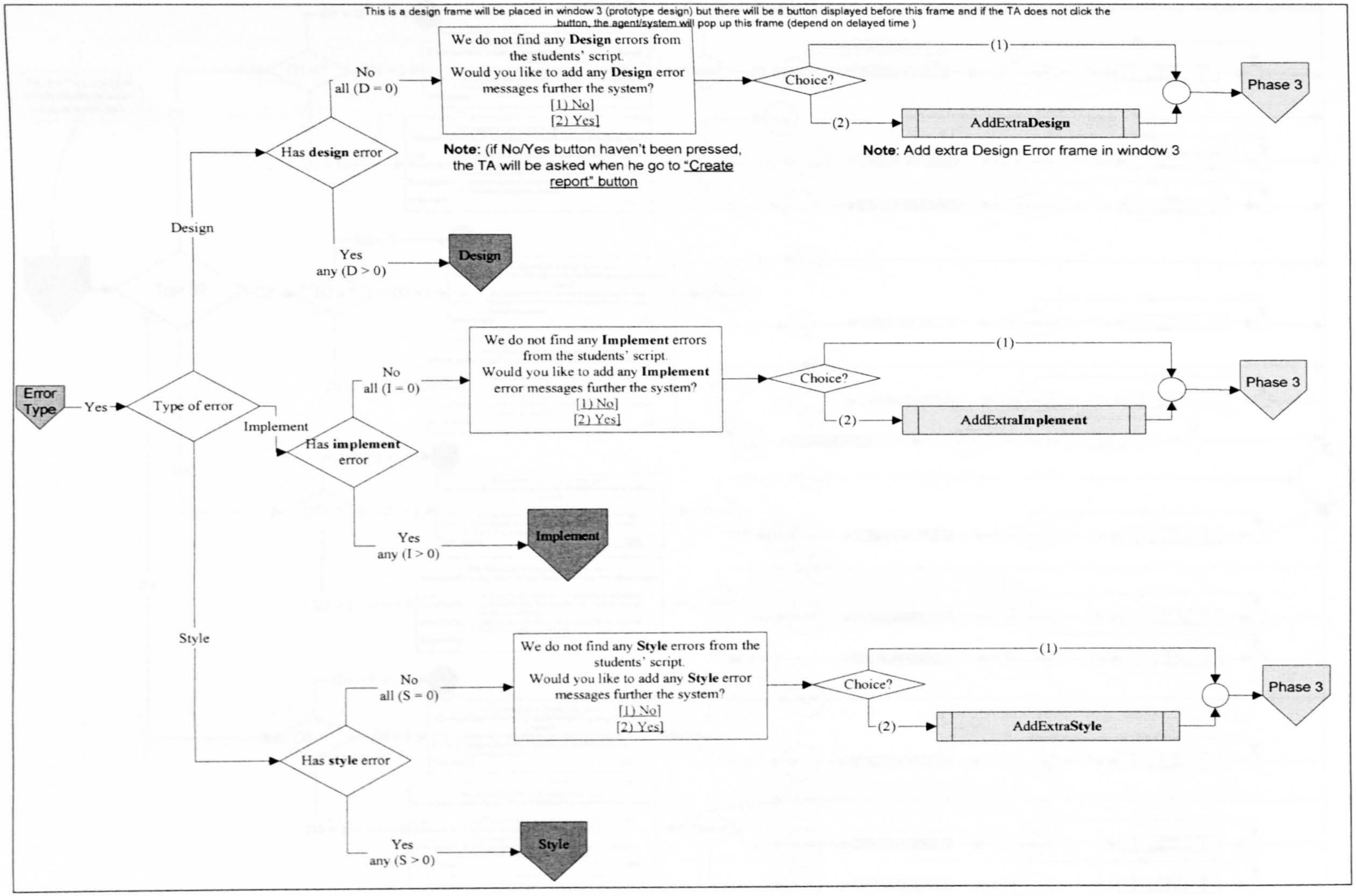


Figure C.4 Design of McFeSPA (Version 1.2): Error Type

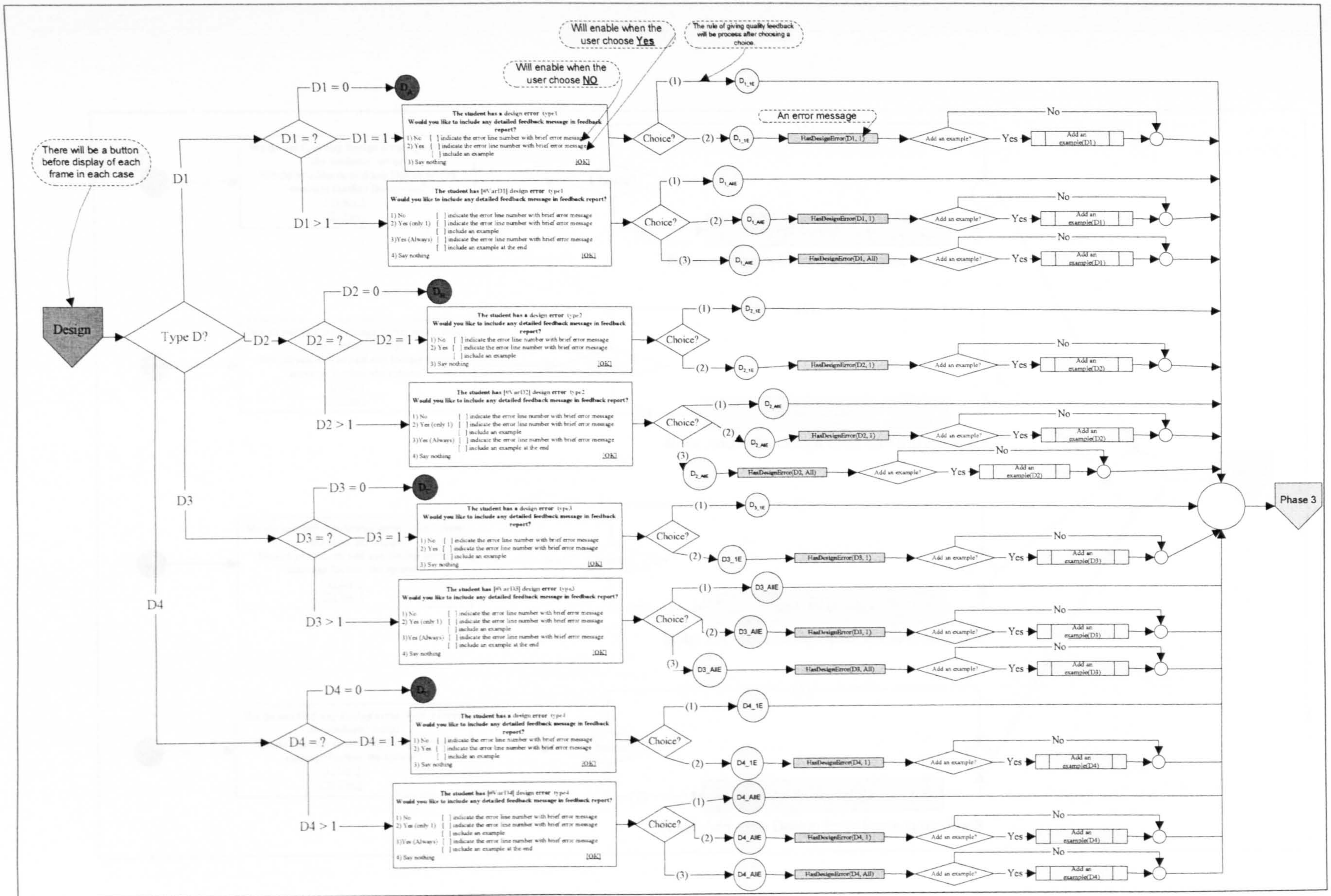


Figure C.5 Design of McFeSPA (Version 1.2): Design type

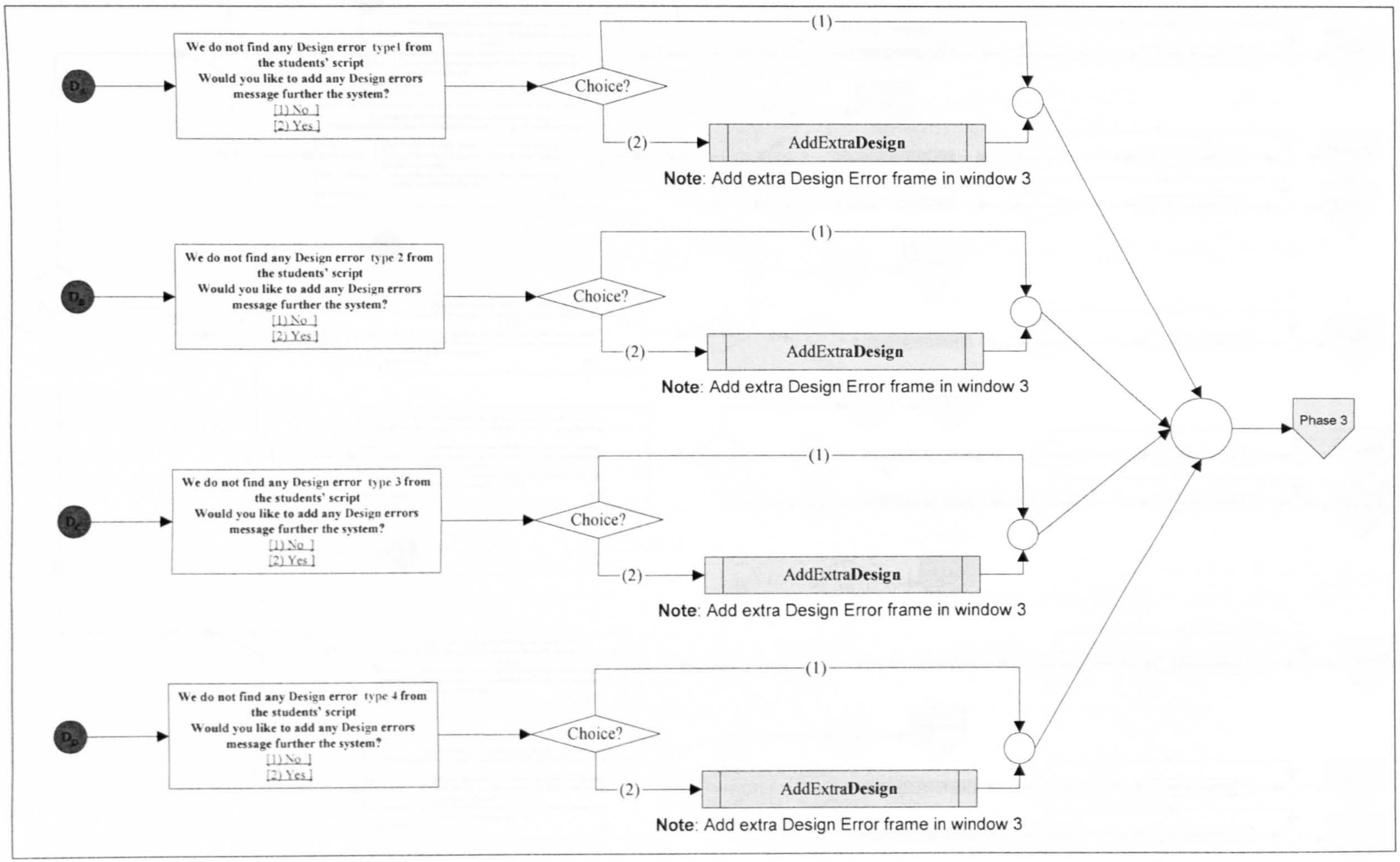


Figure C.6 Design of McFeSPA (Version 1.2): Design type (contd.)

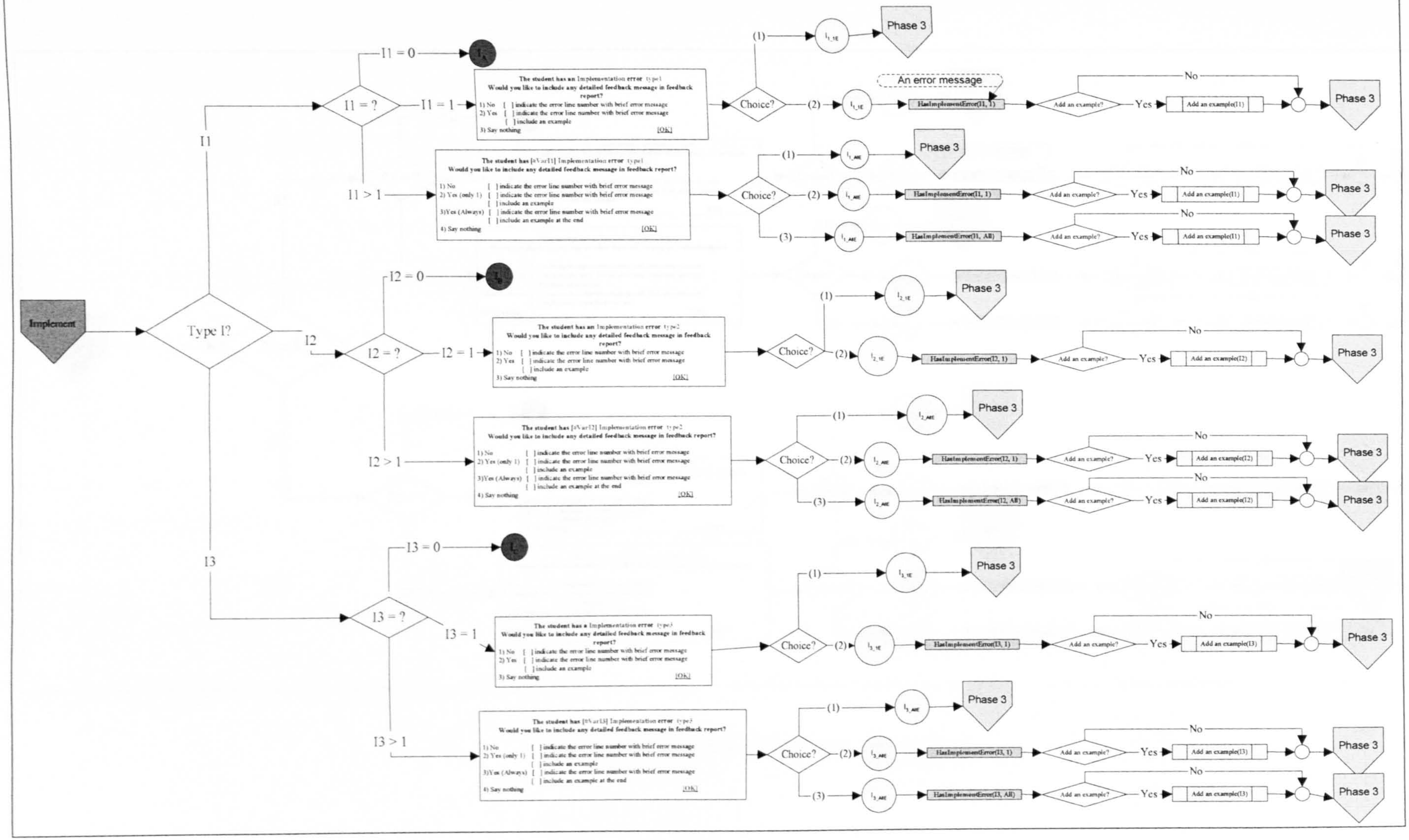


Figure C.7 Design of McFeSPA (Version 1.2): Implementation type

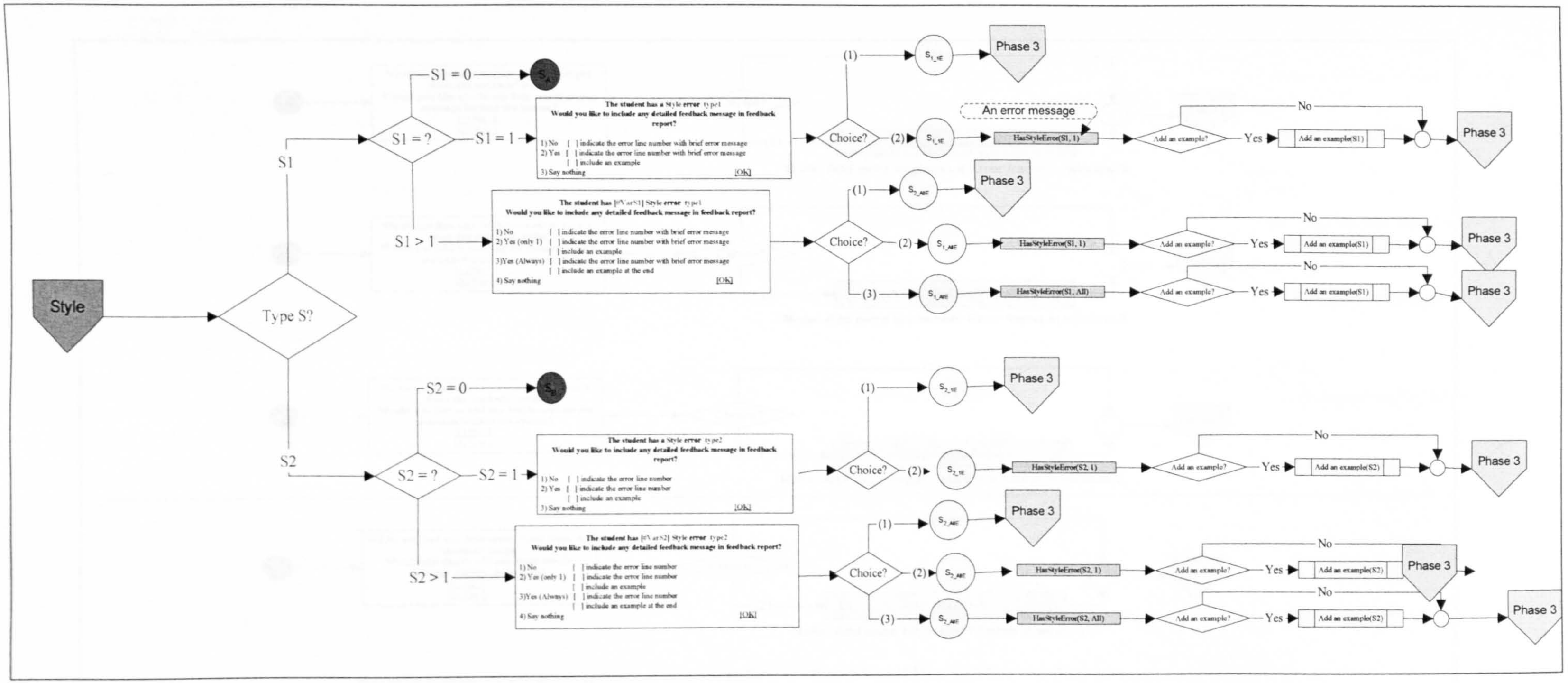


Figure C.8 Design of McFeSPA (Version 1.2): Style type

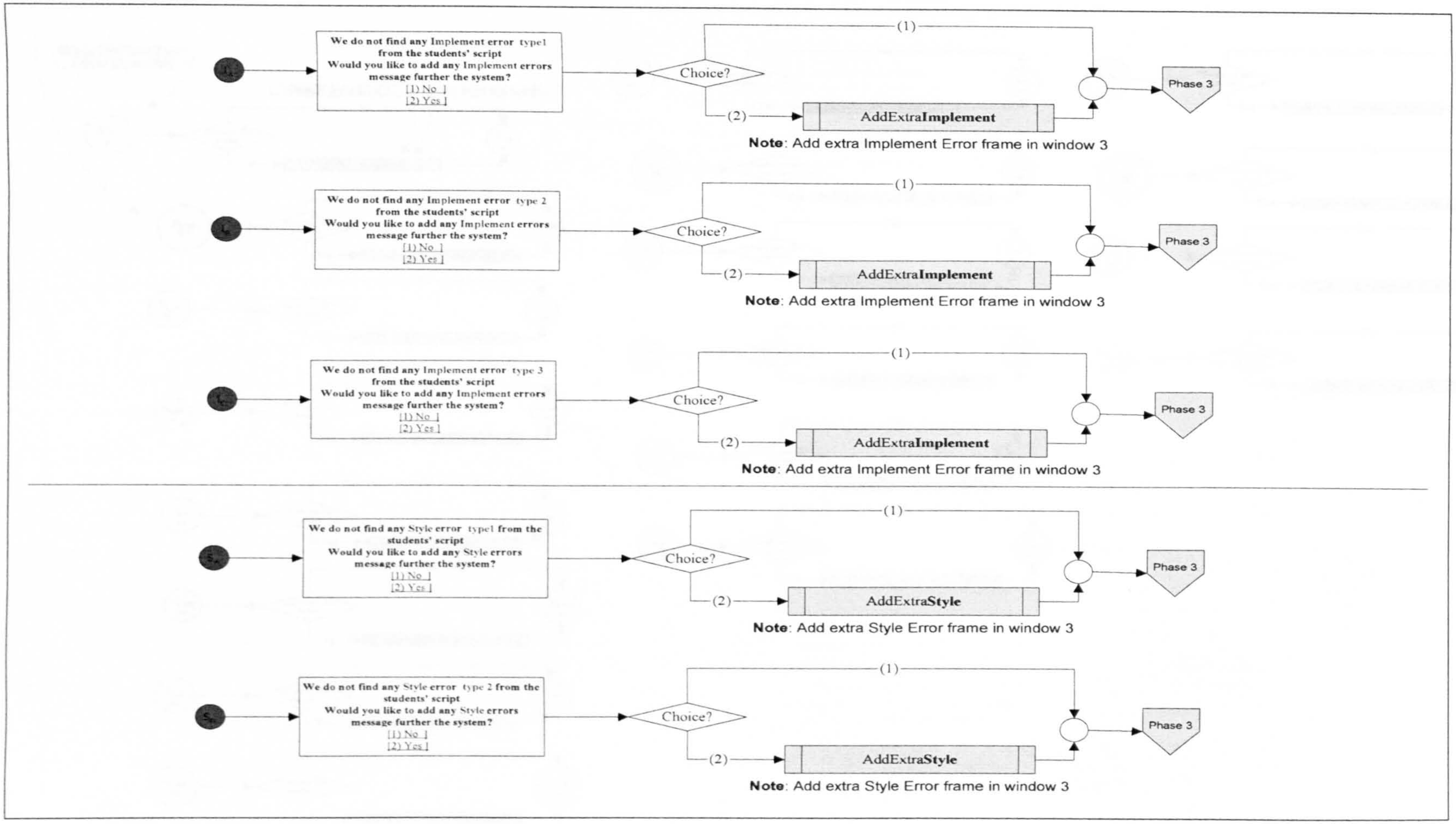


Figure C.9 Design of McFeSPA (Version 1.2): Implementation type & Style type (contd.)

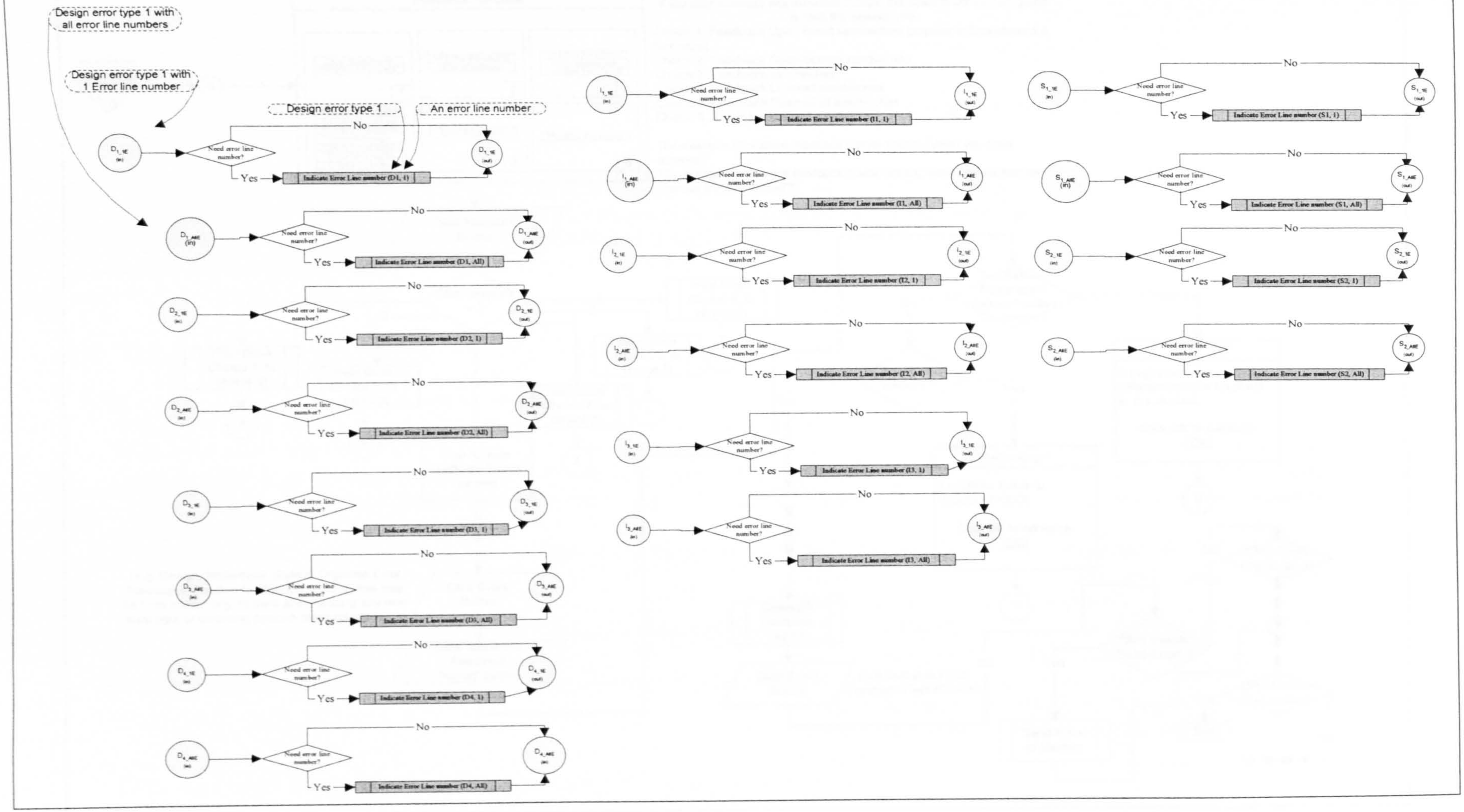


Figure C.10 Design of McFeSPA (Version 1.2): Add line

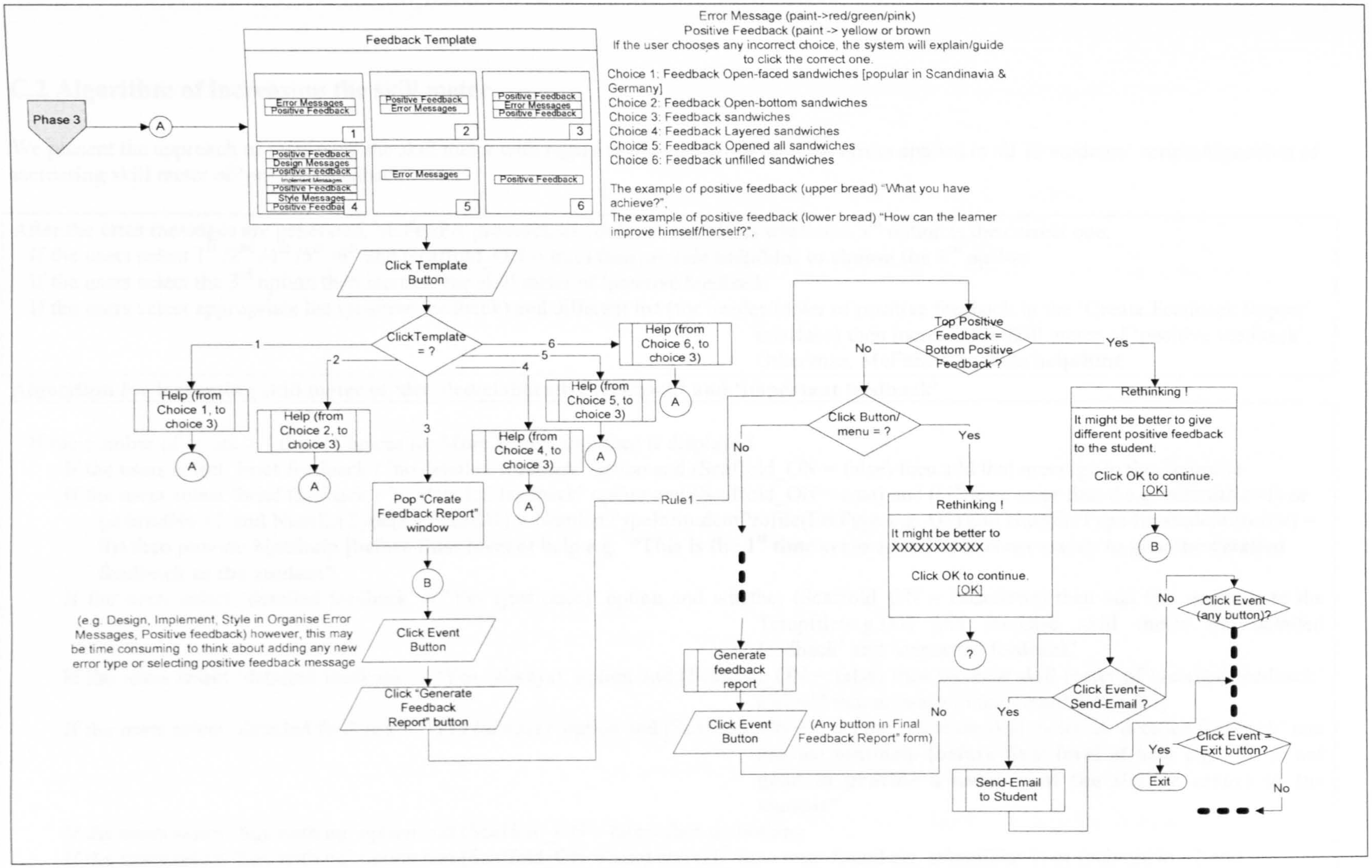


Figure C.11 Design of McFeSPA (Version 1.2): Phase 3

C.2 Algorithm of increasing the skill meter

We present the approach of increasing the skill meter with regard to the number of possible errors applied to all 12 students' scriptsAlgorithm of increasing skill meter of 'positive feedback'

After the error messages are generated, McFeSPA provides six feedback templates which the 3rd option is the correct one.
If the users select 1st /2nd /4th /5th /6th and (scaffold_ON = true) then provide **help/hint to choose the 3rd option**
If the users select the 3rd option then increase the skill meter of 'positive feedback'
If the users select appropriate list (positive feedback) and different list (for header/footer of positive feedback in the 'Create Feedback Report' interface) then increase the skill meter of 'positive feedback';
Otherwise, McFeSPA provide **help/hint**

Algorithm for increasing skill meter of 'detailed/elaborative feedback' and 'important feedback'

If the number of errors > 1 then 'Choices for More Errors' interface is displayed
If the users select 'brief feedback'/ 'no detailed feedback' option and (Scaffold_ON = false) then add that message to the Tempfile
If the users select 'brief feedback'/ 'no detailed feedback' option and (Scaffold_ON = true) and ((1st time error found e.g. submitNo=1) or (submitNo =2 and NumErrType(e.g.NumD1) > NumErrTypeInStudentProfile(ErrType e.g. D1) and NumErrType (in studentProfile) = 0)) then provide **hint/help [before final level of help e.g. "This is the 1st time error found so it is necessary to give the detailed feedback to the student"**
If the users select 'detailed feedback' – 'Yes (just once)' option and whether (Scaffold_ON = **false/true**) then add that message to the Tempfile(e.g.D1) and increase skill meter of 'detailed feedback' and 'important feedback'
If the users select 'detailed feedback' – 'Yes (always)' option and (Scaffold_ON = false) then increase skill meter of 'detailed feedback' and add that message to the Tempfile(e.g.D1)
If the users select 'detailed feedback' – 'Yes (always)' option and (Scaffold_ON = true) then increase skill meter of 'detailed feedback' and provide **hint/help [before final level of help e.g. "It is not good to provide a number of the similar errors to the student"**
If the users select 'Say nothing' option and (Scaffold_ON = false) then do nothing
If the users select 'Say nothing' option and (Scaffold_ON = true) and ((1st time error found e.g. submitNo=1) or (submitNo =2 and NumErrType(e.g.NumD1) > NumErrTypeInStudentProfile (ErrType e.g. D1) and NumErrType (in studentProfile) = 0)) then provide

hint/help [before final level of help e.g. "This is the 1st time error found so it is necessary to give the detailed feedback to the student"

Else if the number of errors = 1 then the 'Choices for One Error' is displayed
If the users select 'brief feedback'/ 'no detailed feedback' option and (Scaffold_ON = false) then add that message to the Tempfile
If the users select 'brief feedback'/ 'no detailed feedback' option and (Scaffold_ON = true) and ((1st time error found e.g. submitNo=1) or (submitNo =2 and NumErrType(e.g.NumD1) > NumErrTypeInStudentProfile (ErrType e.g. D1) and NumErrType (in studentProfile) = 0)) then provide **hint/help [before final level of help e.g. "This is the 1st time error found so it is necessary to give the detailed feedback to the student"**
If the users select 'detailed feedback' option whether (Scaffold_ON = **false/true**) then add that message to the Tempfile(e.g.D1) and increase the skill meter of 'detailed feedback'
If the users select 'Say nothing' option and (Scaffold_ON = false) then do nothing
If the users select 'Say nothing' option and (Scaffold_ON = true) and ((1st time error found e.g. submitNo=1) or (submitNo =2 and NumErrType(e.g.NumD1) > NumErrTypeInStudentProfile (ErrType e.g. D1) and NumErrType (in studentProfile) = 0)) then provide **hint/help [before final level of help e.g. "This is the 1st time error found so it is necessary to give the detailed feedback to the student "**

Algorithm for increasing skill meter of 'feedback loop' and 'individual feedback'

If submitNo >1 [not the 1st submission] **then**
While not(Eof(Student-Profile)) **do**
If ChkOneD1(fromD1toS2) = True/ If ChkManyD1(fromD1toS2) = True then
If NumD1 > NumErrTypeInStudentProfile (D1) and (NumErrTypeInStudentProfile (D1) <> 0)
Select choice (below)
If the users select the 1st option and (Scaffold_ON = false) then add that message to the Tempfile(D1)
If the users select the 1st option and (Scaffold_ON = true) then provide **hint/help [before final level of help e.g. "This is the 2nd time error found (of error type name) that the student made errors more than previous so you should encourage the student to avoid this error"]**

If the users select the 2nd option (at first of error found) then increase skill meter of 'feedback loop' and the skill meter of 'individual feedback
If the users select the 3rd, the 4th, the 5th option then provide **hint/help same as the 1st option**

Else if NumD1 = NumErrTypeInStudentProfile (D1) and (NumErrTypeInStudentProfile (D1) <> 0)
If the users select the 1st option and (Scaffold_ON = false) then add that message to the Tempfile(D1)
If the users select the 1st option and (Scaffold_ON = true) then provide **hint/help [before final level of help e.g. : "This is the 2nd time error found (of error type name) that the student made errors same as previous so you should encourage the student to avoid this error"]**
If the users select the 2nd, the 3rd option then provide **hint/help same as the 1st option**
If the users select the 4th option (at first of error found) then increase the skill meter of 'feedback loop' and the skill meter of 'individual feedback
If the users select 5th option then provide **hint/help same as the 1st option**
Else [NumD1 < NumErrTypeInStudentProfile (D1) and (NumErrTypeInStudentProfile (D1) <> 0)]
If the users select the 1st option and (Scaffold_ON = false) then add that message to the Tempfile(D1)
If the users select the 1st option and (Scaffold_ON = true) then provide **hint/help [before final level of help e.g. "This is the 2nd time error found (of error type name) that the student made errors less than previous so you should encourage the student to avoid this error"]**
If the users select the 2nd option then provide **hint/help same as the 1st option**
If the users select the 3rd option (at first of error found) then increase the skill meter of 'feedback loop' and the skill meter of 'individual feedback
If the users select the 4th, the 5th option then provide **hint/help same as the 1st option**
If NumD1 > NumErrTypeInStudentProfile (D1) and NumErrTypeInStudentProfile (D1) = 0 [first error found of the 2nd submission] **then**
If the users select 1st option then add that message to the Tempfile(D1) + (if select this choice at first, increase skill meter of 'individual feedback'
If the users select 2nd option and (Scaffold_ON = false) then add that message to the Tempfile(D1)
If the users select 2nd option and (Scaffold_ON = true) then provide **hint/help [before final level of help e.g. "This is the 1st time error found (of error type name) so you should encourage the student to avoid this error"]**
(also the 3rd, 4th, 5th option are applied same as the 2nd option)
End [while]
Else [there is no student's profile to compare with because this is the 1st submission]
Select choice:

1: This is the 1st error found, error type name (e.g. of D1), hopefully next time you could improve your script to avoid this type of error

2: This is the 2nd error found, error type name (e.g. of D1); however, you made errors more than previous. Hopefully next time you could improve your script to avoid this type of error.

3: Well done, student's name, you made errors less than previous even this is the 2nd error found, error type name (e.g. of D1). Hopefully next time you could improve your script to avoid this type of error.

4: This is the 2nd error found, error type name (e.g. of D1); hopefully next time you could improve your script to avoid this type of error.

[In the case of the number of previous errors types same as the current number of errors types]

5: Say nothing

If the users select the 1st option then add that message to the Tempfile(D1) + (if select this choice at first, increase skill meter of 'individual feedback'

If the users select the 2nd option and (Scaffold_ON = false) then add that message to the Tempfile(D1)

If the users select the 2nd option and (Scaffold_ON = true) then provide **hint/help [final level of help e.g. "This is the 1st time error found (of error type name) so you should encourage the student to avoid this error"]**

(also the 3rd, 4th, 5th option are applied same as the 2nd option)

(This algorithm is applied not only the design error type 1(D1), also D2, D3, D4, I1, I2, I3, S1, and S2)

In increasing the skill meter of 'Asking question' is quite simple by count the number of presenting of WH-question word (e.g. who, where, why, what, how, when.

In UserProfile, this will be unique for each user and provide only 1 table for each user to correct the current level of help.

All hints at start, first used, will be start from 0 -never obtain any help.

When the users login McFeSPA if the current level of help for each hint is 5 then McFeSPA will shift to 4.

While McFeSPA is processing if the current level of help for each hint is 5 then it will be stable.

Appendix D: Context of Hints

This appendix consists of context of hints and all levels of help for each hint in McFeSPA. Currently, we provide 12 hints, 12 different contexts for which contingent help is available, in the main context of learning how to provide quality feedback.

The contexts, the purposes, and the forms of the hint for all hints in McFeSPA in the current version can be seen in Table D.1.

Table D.1: contexts, purposes and forms of hints in McFeSPA

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
1	When the student has made **more** errors of the same kind than previously.	Help the TA to give feedback message to individual student who has made **more** errors of the same kind than previously to avoid errors with regard to student's error history from student's profile i.e. help the TA to give 'individual feedback' and 'feedback loop'.	Level 1: "Are you absolutely sure? Try again, [TA's Name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Look for the meaning of 'Feedback loop' in the glossary. Have another go, [TA's name]." Level 4: "This is the 2nd occurrence of an error of [type name]) which the student has been making more than previously. You should encourage the student to avoid this error. Have another go, [TA's name]." Level 5: "The right answer is the 2nd choice which gives you a good 'Feedback loop'."
2	When the student has made **the same number of** errors of the same kind as previously.	Help the TA to give feedback message to individual student who has made **the same number of** errors of the same kind as previously to avoid errors with regard to student's error history from student's profile i.e. help the TA to give 'individual feedback' and 'feedback loop'.	Level 1: "Are you absolutely sure? Try again, [TA's Name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Look for the meaning of 'Feedback loop' in the glossary. Have another go, [TA's name]." Level 4: "This is the 2nd occurrence of an error of [type name]) which the student has been making same as previously. You should encourage the

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
			previously. You should encourage the student to avoid this error. Have another go, [TA's name]." Level 5: "The right answer is the 4th choice which gives you a good 'Feedback loop'."
3	When the student has made **less** errors of the same kind than previously.	Help the TA to give feedback message to individual student who has made **the less number of** errors of the same kind than previously to avoid errors with regard to student's error history from student's profile i.e. help the TA to give 'individual feedback' and 'feedback loop'.	Level 1: "Are you absolutely sure? Try again, [TA's Name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Look for the meaning of 'Feedback loop' in the glossary. Have another go, [TA's name]." Level 4: "This is the 2nd occurrence of an error of [type name]) which the student has been making less than previously. You should encourage the student to avoid this error. Have another go, [TA's name]." Level 5: "The right answer is the 3rd choice which gives you a good 'Feedback loop'."
4	When the student has made an error of the **1st** time.	Help the TA to give feedback message to individual student who has made an error the 1st time to avoid errors with regard to student's error history from student's profile i.e. help the TA to give 'individual feedback' and 'feedback loop'.	Level 1: "Are you absolutely sure? Try again, [TA's Name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Look for the meaning of 'Feedback loop' in the glossary. Have another go, [TA's name]." Level 4: "This is the 1st occurrence of an error of [type name]). You should encourage the student to avoid this error. Have another go, [TA's name]." Level 5: "The right answer is the 1st choice which gives you a good 'Feedback loop'."

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
5	When the student has made a particular error the 1st time	Help the TA to explain more detail feedback i.e. help the TA to give 'detailed/elaborative feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Think! What makes 'Detailed/Elaborative feedback' good? Have another go, [TA's name]." Level 3: "Look for the meaning of 'Detailed/Elaborative feedback' in the glossary. Have another go, [TA's name]." Level 4: "It would be better to provide 'Detailed/Elaborative feedback' at first time (of this type error found). Have another go, [TA's name]." Level 5: "The best answer which gives you a good 'Detail/Elaborative feedback' should be the 'Yes' option - provide 'Detailed/Elaborative feedback."
6	When there are a number of the same kinds of error found whether such errors happened at 1st time or not	Help the TA not to give too much comment or to every error message i.e. help the TA to give 'important/specific feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Think! What makes 'Important/Specific feedback' good? Have another go, [TA's name]." Level 3: "Look for the meaning of 'Important/Specific feedback' in the glossary. Have another go, [TA's name]." Level 4: "It would be better to provide 'Important/specific feedback' (of this type error found) only once. Have another go, [TA's name]." Level 5: "The best answer which gives you a good 'Important/Specific feedback' should be the 'Yes- just once' option -provide 'Important/Specific feedback' only once."

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
7	When the TA does not select the "feedback sandwich" template which is the error message is between two positive feedbacks	Help the TA to select the best feedback template for generating feedback report. The best feedback template is giving feedback sandwiches, giving either 'negative feedback' or error messages between 'positive feedback' i.e. help the TA to give 'positive feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5: "The best feedback is the error message between two 'Positive feedback', select upper rightmost button - that is the best feedback pattern."
8	When the TA does not put the student's name or not the right name into the feedback report.	Help the TA to be careful to give the right student's name in the student's script marking while generating the feedback report i.e. help the TA to give 'individual feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "The student's name should be retrieved from the student's table. Try again, [TA's name]." Level 3: "Think! What makes 'Individual feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Individual feedback' in the glossary. Have another go, [TA's name]." Level 5: "The student's name should be student's name or student's surname or both."
9	When the TA does not give feedback sandwiches with regard to the starting detail of 'positive feedback'	Help the TA give feedback sandwiches with regard to the starting detail of 'positive feedback'.	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive feedback' good? Have another go, [TA's name]."

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
			Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5: "The upper feedback position should be the beginning feedback."
10	When the TA does not give feedback sandwiches with regard to the ending detail of 'positive feedback'	Help the TA give feedback sandwiches with regard to the ending detail of 'positive feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5; "The lower feedback position should be the ending feedback."
11	When the TA does not give feedback sandwiches with regard to the positive detail of in the starting 'positive feedback'	When the TA give feedback sandwiches with regard to the positive detail of in the starting 'positive feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5: "Look at the upper feedback position, this seems not quite right to provide a kind of 'Negative feedback'."
12	When the TA does not give feedback sandwiches with regard to the positive detail of in the ending 'positive feedback'	Help the TA give feedback sandwiches with regard to the positive detail of in the ending 'positive feedback'	Level 1: "Are you absolutely sure? Try again, [TA's name]." Level 2: "Good, but it is possible to improve. Try again, [TA's name]." Level 3: "Think! What makes 'Positive

Hint No.	Context of Hint	Purpose of Hints	Form of Hints
	feedback'		feedback' good? Have another go, [TA's name]." Level 4: "Look for the meaning of 'Positive feedback' in the glossary. Have another go, [TA's name]." Level 5: "Look at the lower feedback position, this seems not quite right to provide a kind of 'Negative feedback'."

Giving hint#1-4 is aimed to help the TA to consider individual student with regard to student's error history from student's profile as well as this is associate with giving 'feedback loop'.

Giving hint# 9-12 is aimed to help the TA to consider detail and position for giving 'positive feedback' (of feedback sandwich).

There are similar forms of hints but the contexts and purposes are different. We aware that later version could be improved the language used and the systematic could be changed

Our first version is aimed to provide 12 hints to test usability. Later versions the number of hints can be whether decreased or increased from the current amount of hint depend upon the users' response.

Appendix E: Materials for usability evaluation

This appendix presents useful materials for usability evaluation i.e. handout of usability of the interface, semi-structured interview sheet (for evaluators to comment on the system's interface), system checklist questionnaire sheet, observation sheet. The materials can be seen from the following.

E.1 Hand out for evaluation of the usability of the interface

Empirical study on

Metacognitive Feedback Scaffolding system for Pedagogical Apprenticeships

Environment Interface

Section 1

Description:

The goal of this study is to evaluate the usability of a human-computer interface intend to be used in a computer-support for giving feedback environment. The study consists of three stages. In the first stage, some basic background is required from the evaluator and followed by the requirement of comments from screen capture of the system from you with semi-structured interview. In the second stage this is acquired evaluator's comment with the interface for customizing the system. In the third stage, the evaluator will be asked to generate three feedback reports and comment on the interface. That is generating a feedback report with some errors messages provided an automated analysis made by the system with extra help from the system.

About McFeSPA:

McFeSPA is a system designed to help the user learns to provide quality feedback. McFeSPA is abbreviated from 'Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship'

The starting point of this research is that most people know that giving feedback can motivate the learner to learn; however, one of the most important problems for teaching and learning is how to provide quality feedback to students. In particular, in large classes, it is difficult to provide quality feedback.

Quality feedback has been found to depend on many factors. Here are some examples of factors important for quality feedback (1) Quantity: quality feedback should include detailed content; (2) Individual: quality feedback should pay attention to the individual learner; (3) Timing: quality feedback should be delayed or immediate as appropriate; (4) A good error analysis: quality feedback should provide the correct answer following every error; (5) Positive: quality feedback should be encouraging; etc.

However, there are some problems when trying to give quality feedback, for example

1) Quantity: if the teacher gives details but no specific guidance, the feedback may not be enough to help the learners improve their learning.
2) Individual: if the teacher gives inconsistent feedback, for example, once the teacher has given feedback to a learner he may forget their performance on their previous work and then provide inconsistent feedback. This is vital because taking individual difference into account effectively is can motivate learners to learn.
3) Timing: if the teacher gives feedback too late to the learners, it may not help them learn.
4) A good error analysis: if the teacher does not explain any error, not distinguishing one bug from another, not focusing on an important error, they may not learn.
5) Positive: if the teacher gives inappropriate positive feedback or is not reasonable, it may discourage the learner.
6) Beyond this, other problems include giving unrealistic feedback, not noticing the learners' improvement etc. Such problems may lead to ineffective learning.

Even if automated marking assignment can help the teachers mark, it is important to help novice teachers and teaching assistants (TAs) who have not learnt how to give quality feedback. There is not much research, if any, in training people to give quality feedback using computer-based systems. Therefore, this brings us to the opening question. That is "How to train TAs?"

The Scaffolding approach has been applied effectively to help both adult and child learners learn. Therefore, the goal of this study is to employ a scaffolding framework to help the teaching assistants (TAs) improve their skills for giving quality feedback in the actual environment of marking programming assignments.

Section 2:

Evaluator background:

1. Could you please briefly describe your background in usability testing?

.....

.....

.....

2. How would you describe 'Usability testing'?

.....

.....

.....

Section 3

Now you will be asked to role play a TA. While you are using McFeSPA, you are able to think aloud. It would be helpful could you please read each task aloud.

Task 1: Using McFesPA's customisation

McFesPA's customisation consists of customising for 'Setting scaffold'; 'Favourite wording'; 'Favourite content'; 'Report template'; 'Manage error/weakness message'. Because the techniques involve in managing each item are quite similar when compared with each other, you will be asked to try add/update/delete your preferred data for the customisation of 'Favourite wording' as shown in Figure E.1. By double clicking the list item- '***to alter this list, double click here***' then a window for managing data, as shown in Figure E.2, will appear.

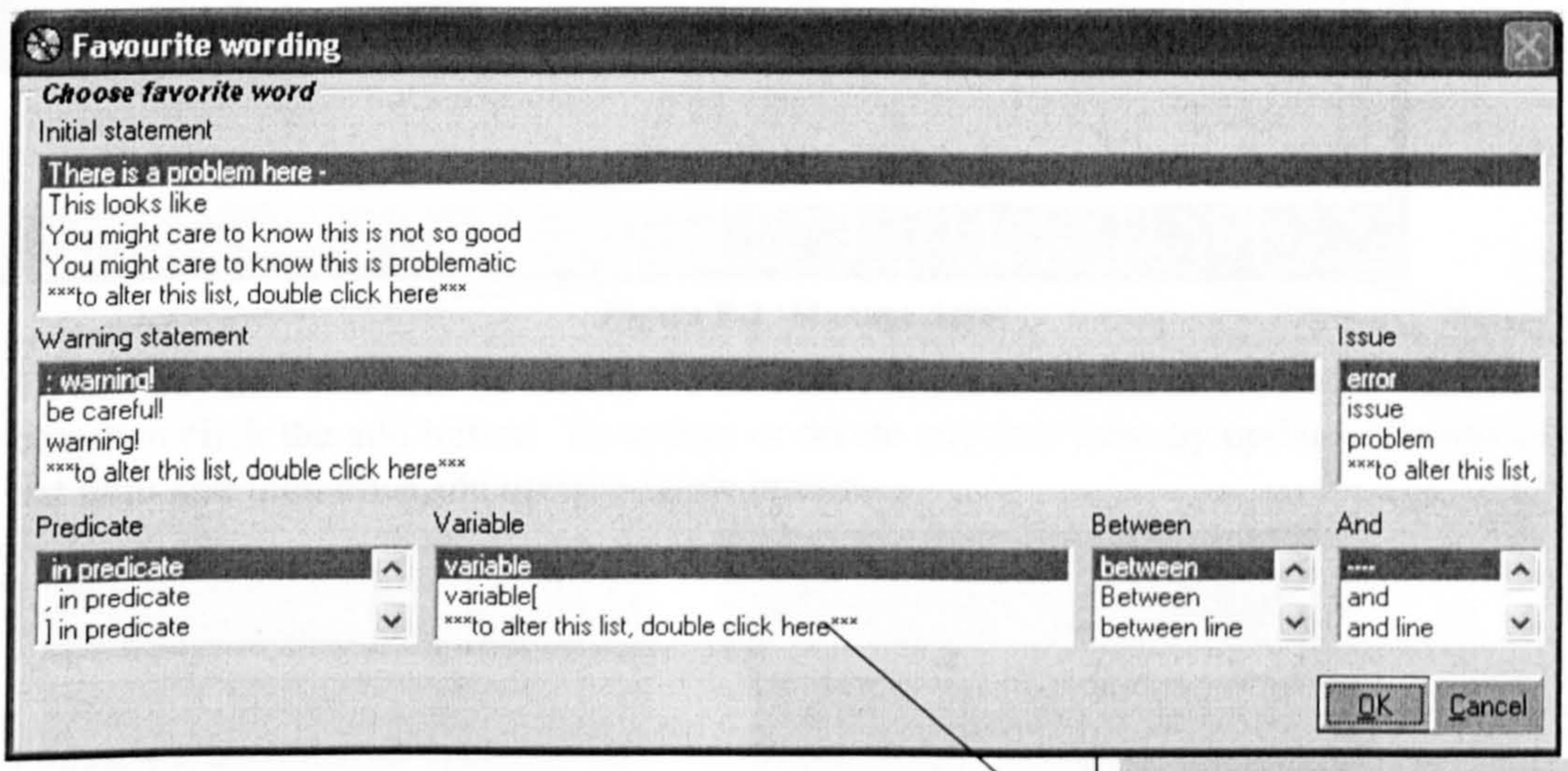


Figure E.1 'Favourite wording'

Double click the bottom list item of each list box to add/update/delete the message in each list box

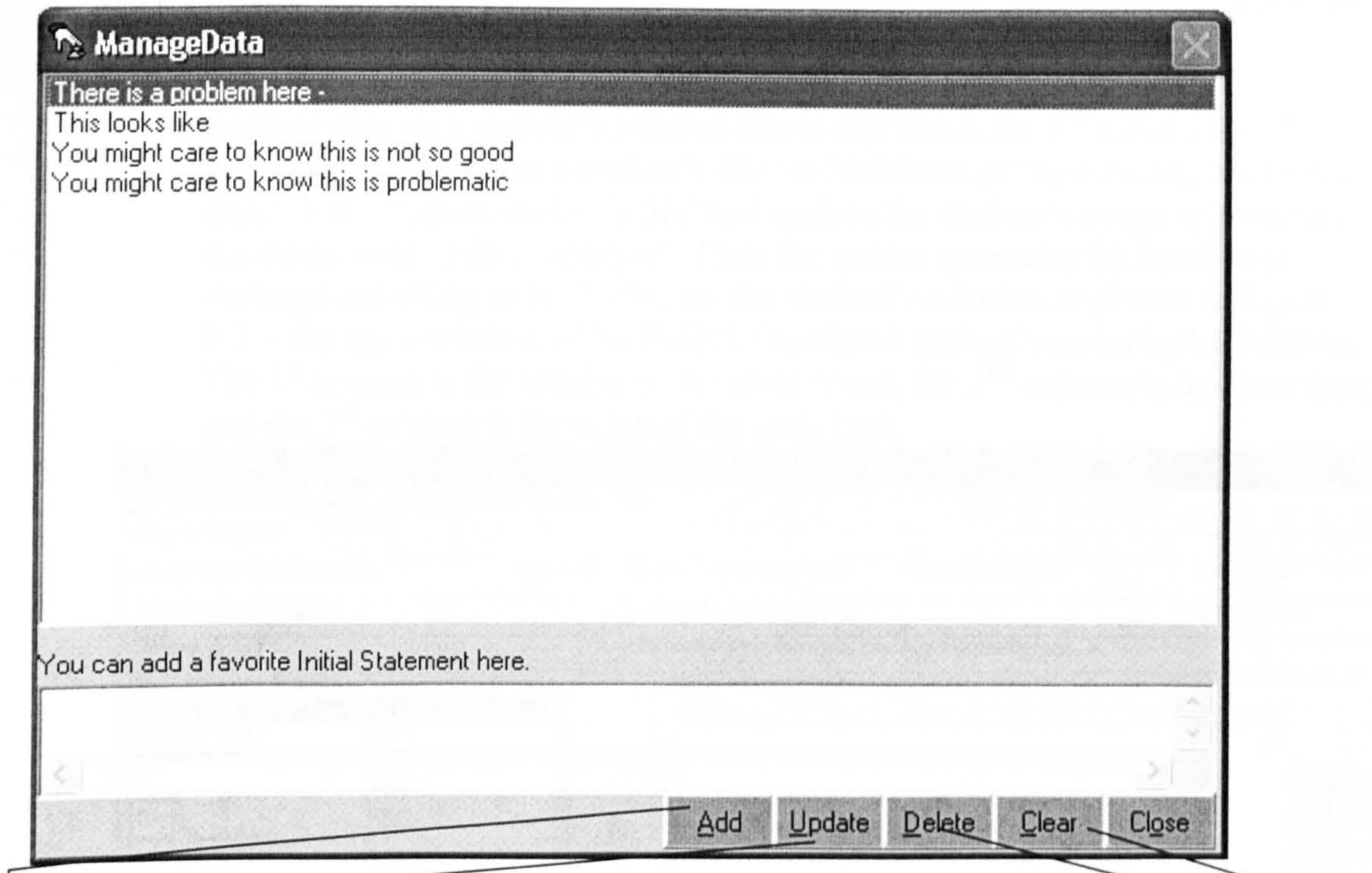


Figure E.2 'Manage data'

To add any new list item by typing the new data in the available space under the list box then click the add button. To update or delete any new item by update/deleting any list item and then click the update/delete button.

- Update the message from the text box below to the selected list box
- Delete the message in the selected list box above
- Add the message from the text box below to the list box above
- Clear the message in the text box below

You now finish the task 1, please close the 'Favourite wording' window by either clicking 'OK' button or 'Cancel' button.

Task 2: Using McFesPA to generate feedback report with scaffolding provided by McFeSPA

1. Assume that each student's solution file in this test is the 2nd submission. You are now asked to select a student's file '4120390102.pl' by selecting the menu item: 'File', 'Select student's file' and analyse the student's script by selecting the menu item: 'File', 'Analyse'. Then the system generates the brief error message according to the list below the student's solution as shown in Figure E.3 – the main window of McFeSPA (Analysed student's script by McFeSPA). The 1st column is the number of the error found, the 2nd column is the error type, and the 3rd column is the name of the error type.

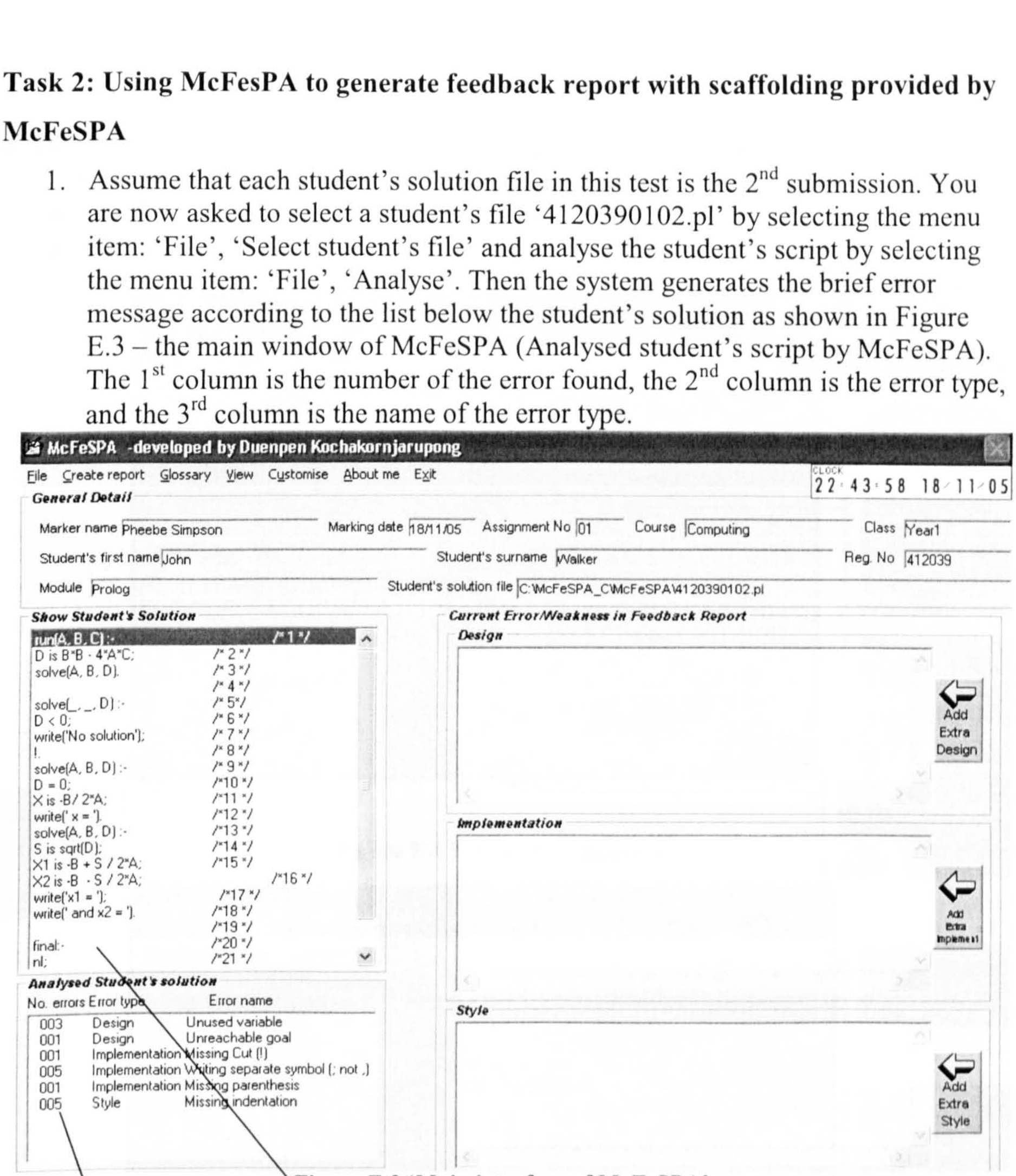


Figure E.3 'Main interface of McFeSPA'

Show the student's solution after selecting 'File' menu item → 'Select student's file' menu item

Show brief error messages generated by the system after selecting 'File' menu item → 'Analyse' menu item. The 1st column is the number of the error found (of the same error type), the 2nd column is the error type, and the 3rd column is the name of the error type.

2. To generate feedback message according to the brief messages as shown in three columns in the list box of 'Analysed Student's solution' pane.
3. Due to the fact that you are in the scaffold-on mode, from this stage the system will provide help to you for any departure from the solution path of the system. However, you can accept or refuse any help from the system.

4. Once you click on the list items in which the 1st column (number of errors) is 1, the 'choice for one error' window will be displayed as shown in Figure E.4. In other words, the 'Choice for more errors' window will be displayed for the list item in which the number of errors is more than one, as can be seen in Figure E.5.

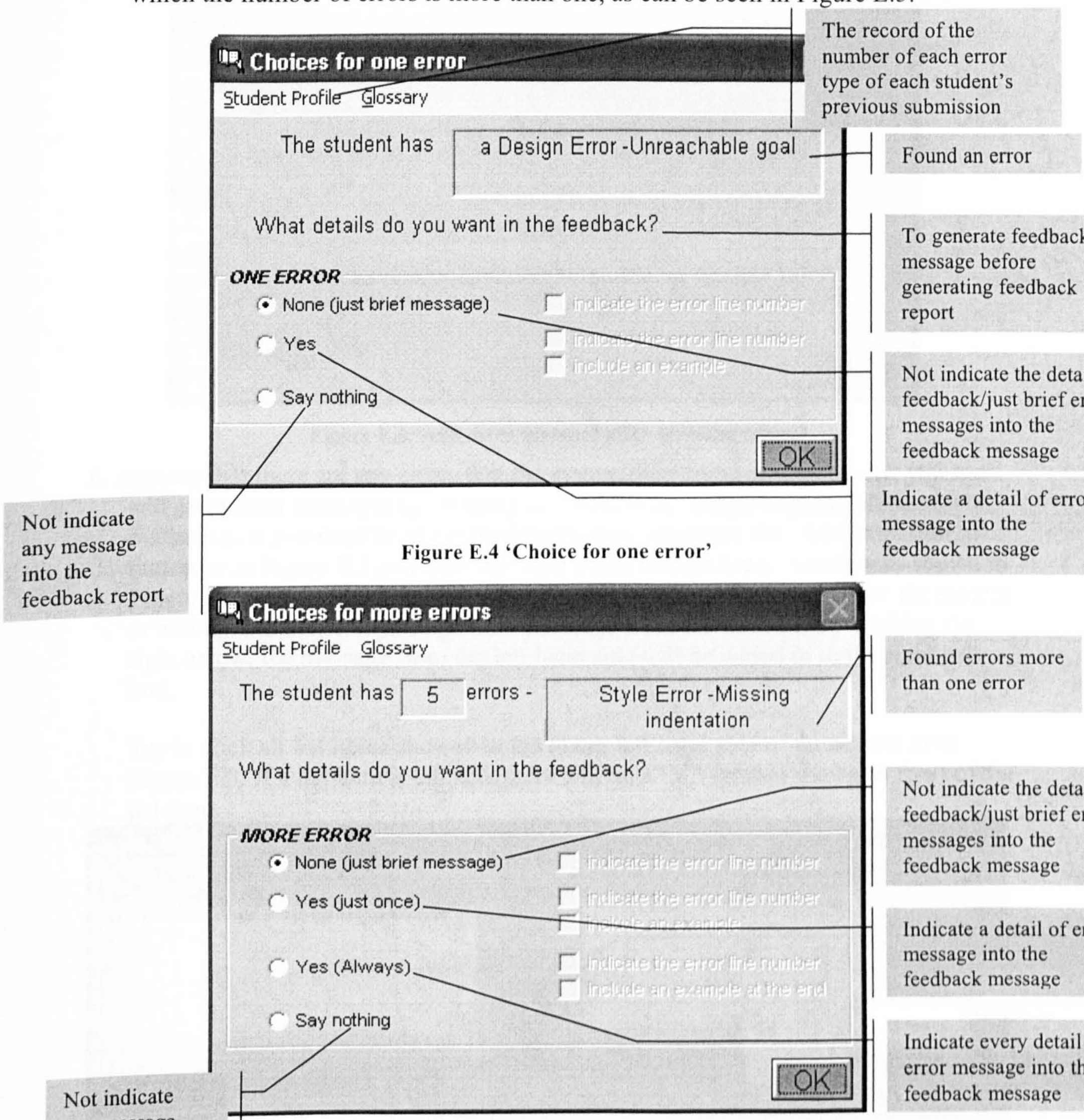


Figure E.4 'Choice for one error'

Figure E.5 'Choice for more errors'

5. Try to select the options provided in Figure E.4 and Figure E.5 to generate feedback message. Once you select the best option and press the 'OK' button. Then there will be the message box asks that you 'Do you need to take into account the history of student's error?' If you do agree to do this, 'Add extra sentence after error messages' window as shown in Figure E.6 will be displayed with five options. When you select

the best option and press 'OK' button. Selected options will be generated and the feedback messages will be displayed in design/implementation/style text box as shown in Figure E.3.

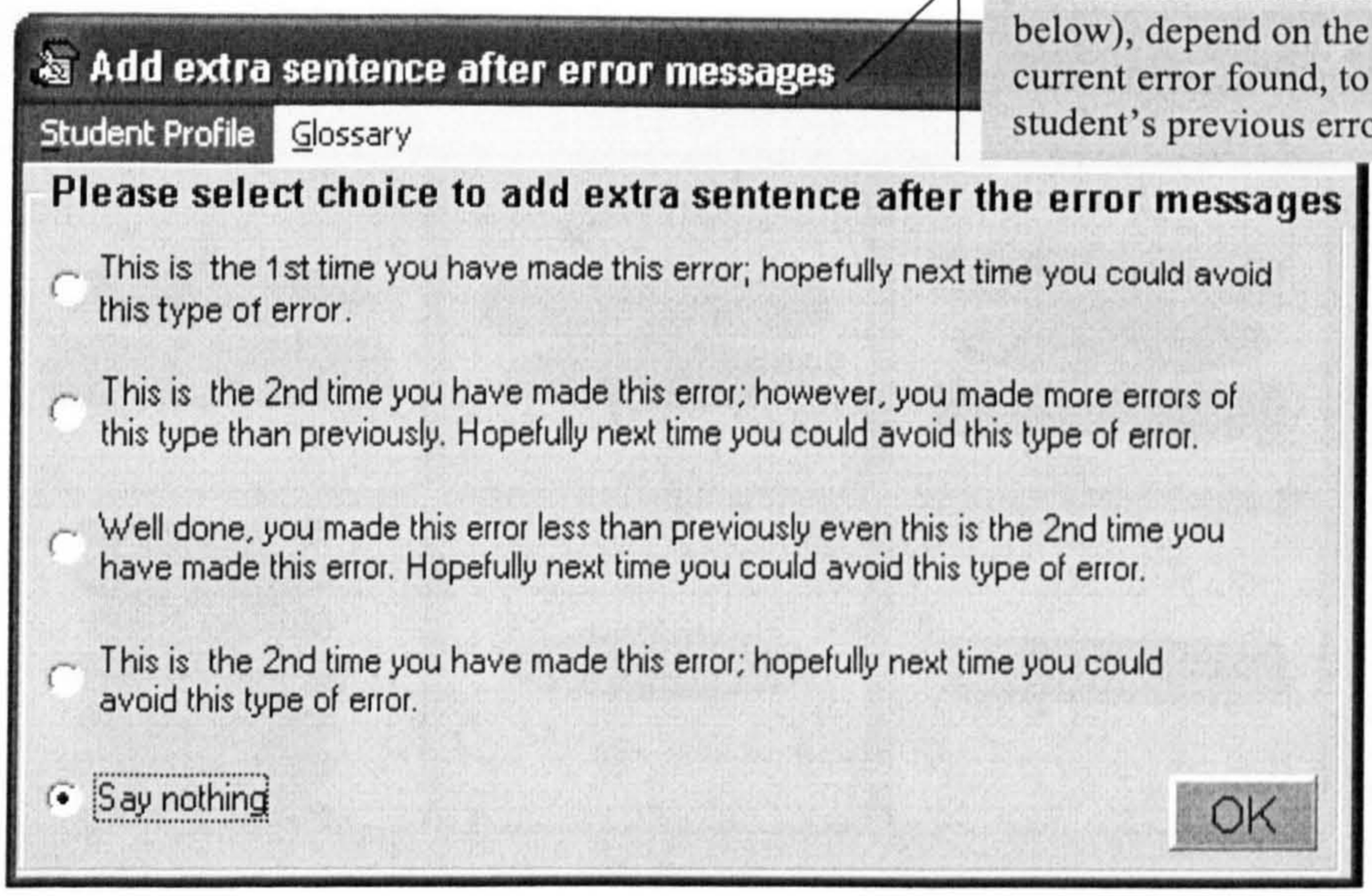


Figure E.6 'Add extra sentence after error messages'

6. Assume that there are any errors that the system didn't find automatically, you can add extra error messages by pressing the 'Add extra design/implementation/style' button e.g. if you need to add extra design error, you press the 'Add extra design' button as in Figure E.3 and then the 'Add Extra Design Error' window as shown in Figure E.7 will appear. You can select any available error message from the system or add/update/delete any design error message from the list box. By clicking the right arrow, the message from the left hand side will be added in to the right list box.

   Try to click all list items showed in the lower left hand side of the screen as in Figure E.3. If a list item is clicked, there will be a '@' symbol shown in front of the list item.

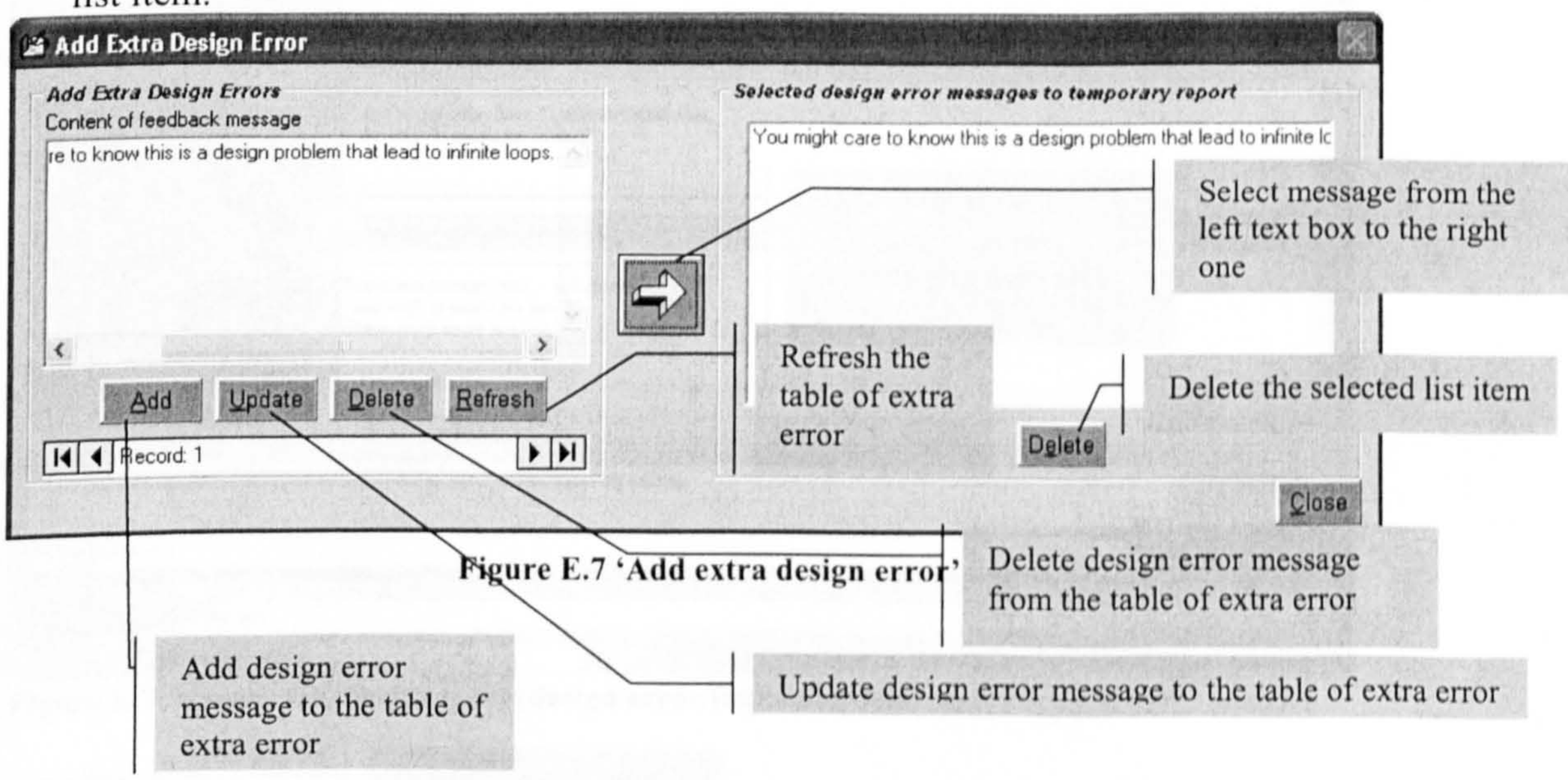


Figure E.7 'Add extra design error'

7. You will be asked to click 'Create report' menu to select a 'feedback template', as shown in Figure E.8, before generating feedback report. Please select any choice button.

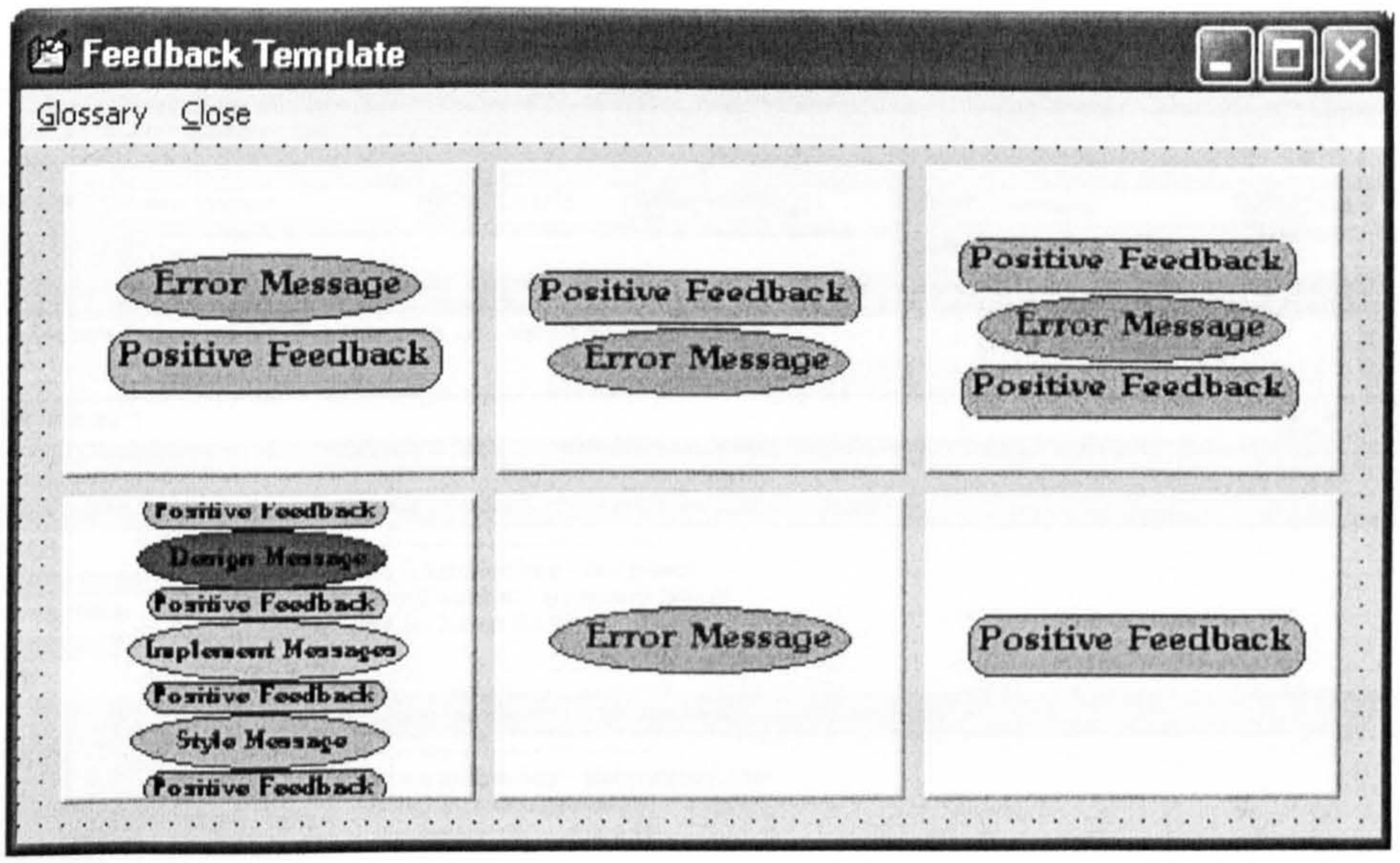


Figure E.8 'Feedback Template'

8. Assume that the left most upper template is pressed, the report template will be generated according to Figure E.9. You can order the sequence of error messages with regard to the error type (Design/Implementation/Style) according to your need. Then go to 14.

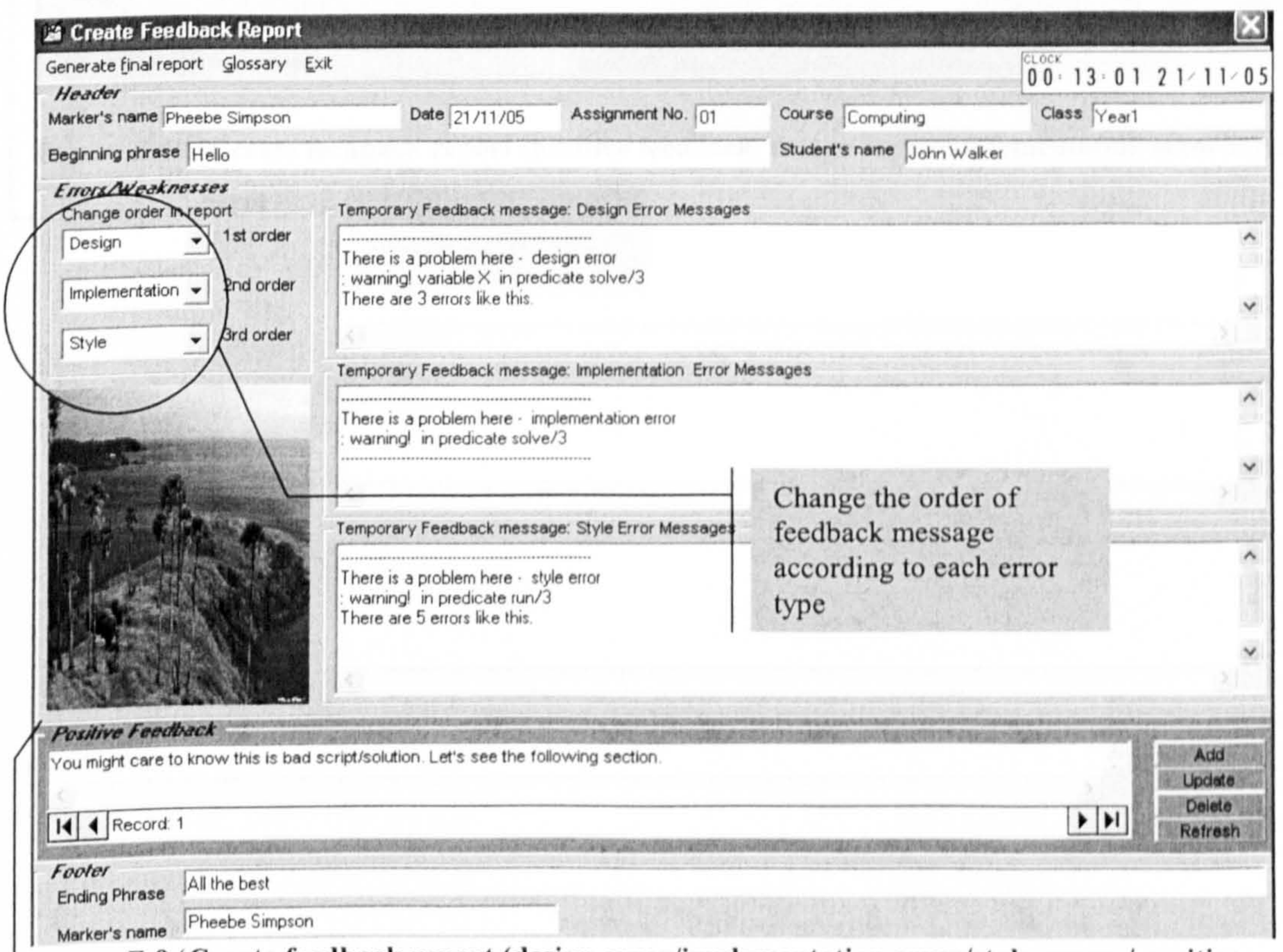


Figure E.9 'Create feedback report (design error/implementation error/style error / positive feedback)'

Choice for possible 'Positive Feedback'

9. Assume that the middle upper template is pressed, the report template will be generated according to Figure E.10. You can order the sequence of error messages with regard to the error type (Design/Implementation/Style) according to your need. Then go to 14.

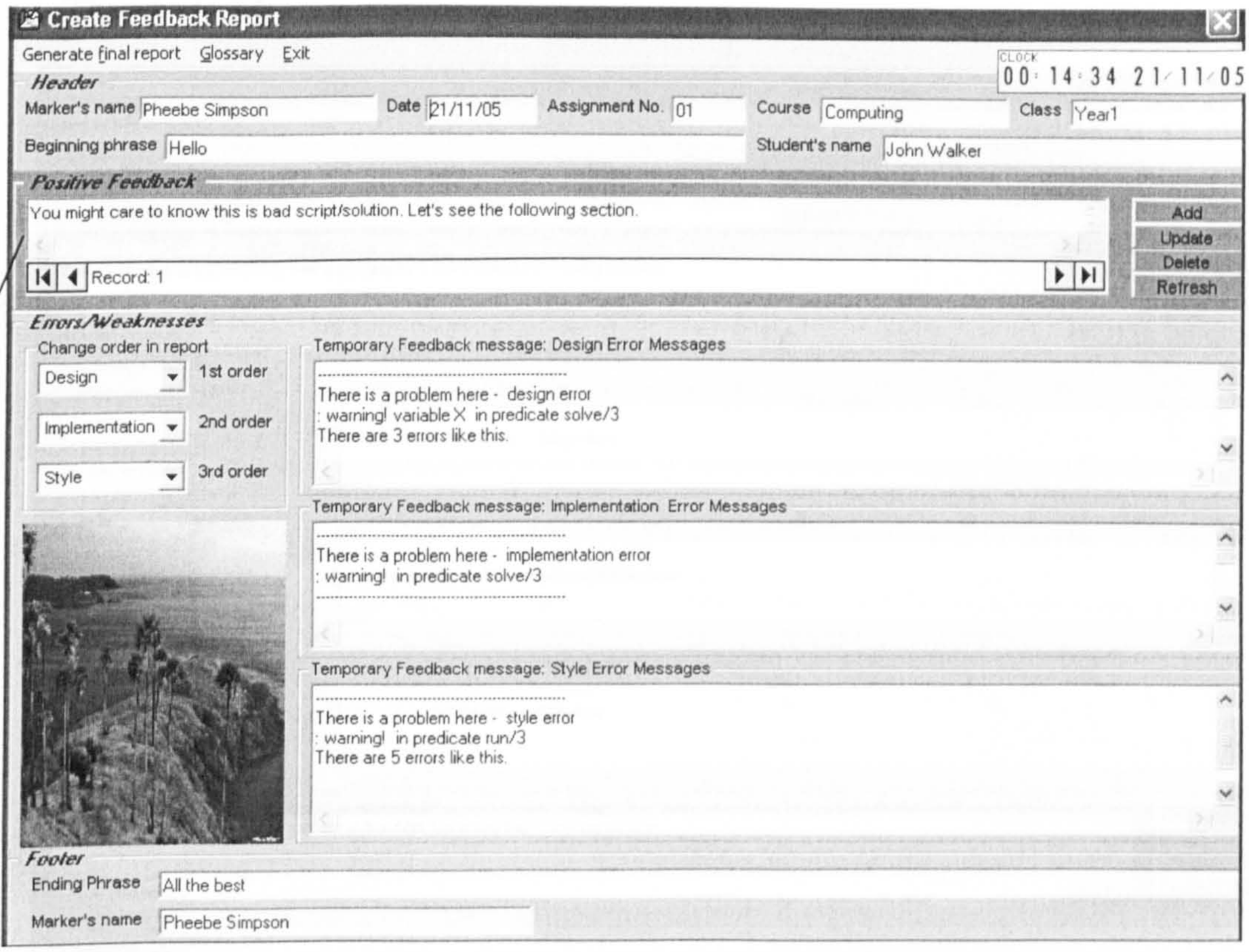


Figure E.10 'Create feedback report (positive feedback/ design error/implementation error/style error)'

Choice for possible 'Positive Feedback'

10. Assume that the right most upper template is pressed, the report template will be generated according to Figure E.11. You can order the sequence of error messages with regard to the error type (Design/Implementation/Style) according to your need. Then go to 14.

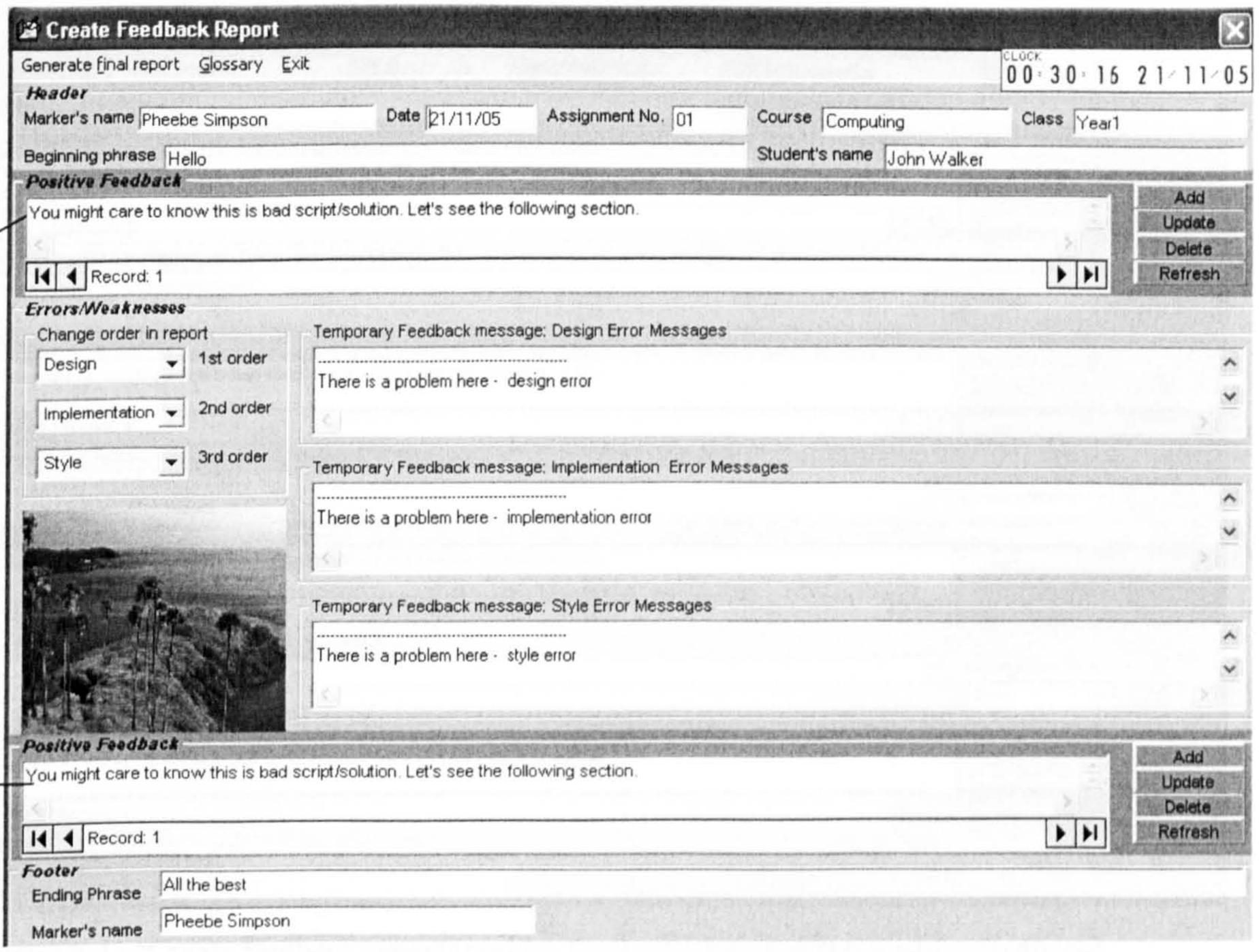


Figure E.11 'Create feedback report (positive feedback/ design error/implementation error/style error / positive feedback)'

Choice for possible 'Positive Feedback'

11. Assume that the left most lower template is pressed, the report template will be generated according to Figure E.12. You can order the sequence of error messages with regard to the error type (Design/Implementation/Style) according to your need. Then go to 14.

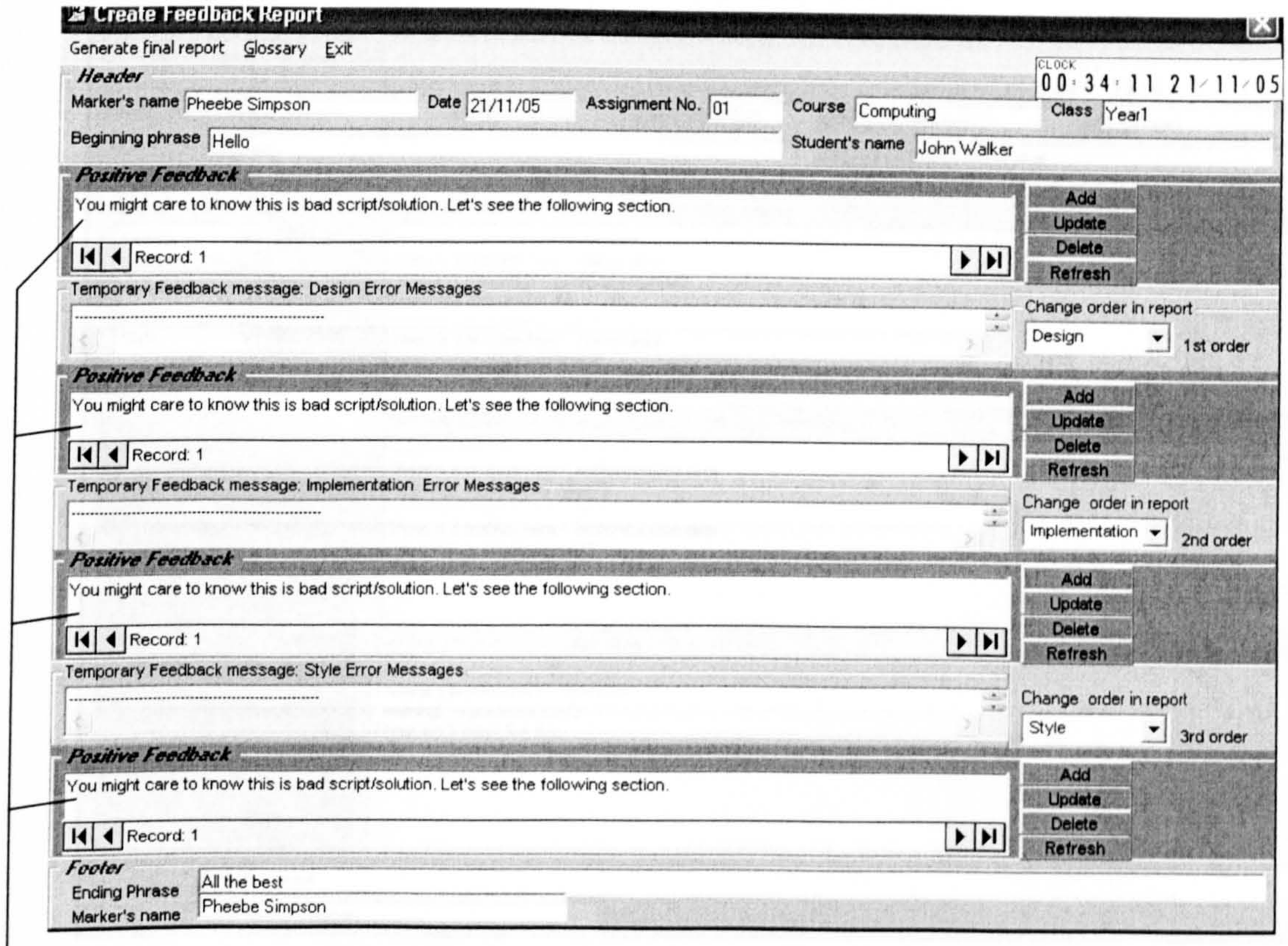


Figure E.12 'Create feedback report (positive feedback/ design error/ positive feedback/ implementation error/ positive feedback/style error/ positive feedback)'

Choice for possible 'Positive Feedback'

12. Assume that the middle lower template is pressed, the report template will be generated according to Figure E.13. You can order the sequence of error messages with regard to the error type (Design/Implementation/Style) according to your need. Then go to 14.

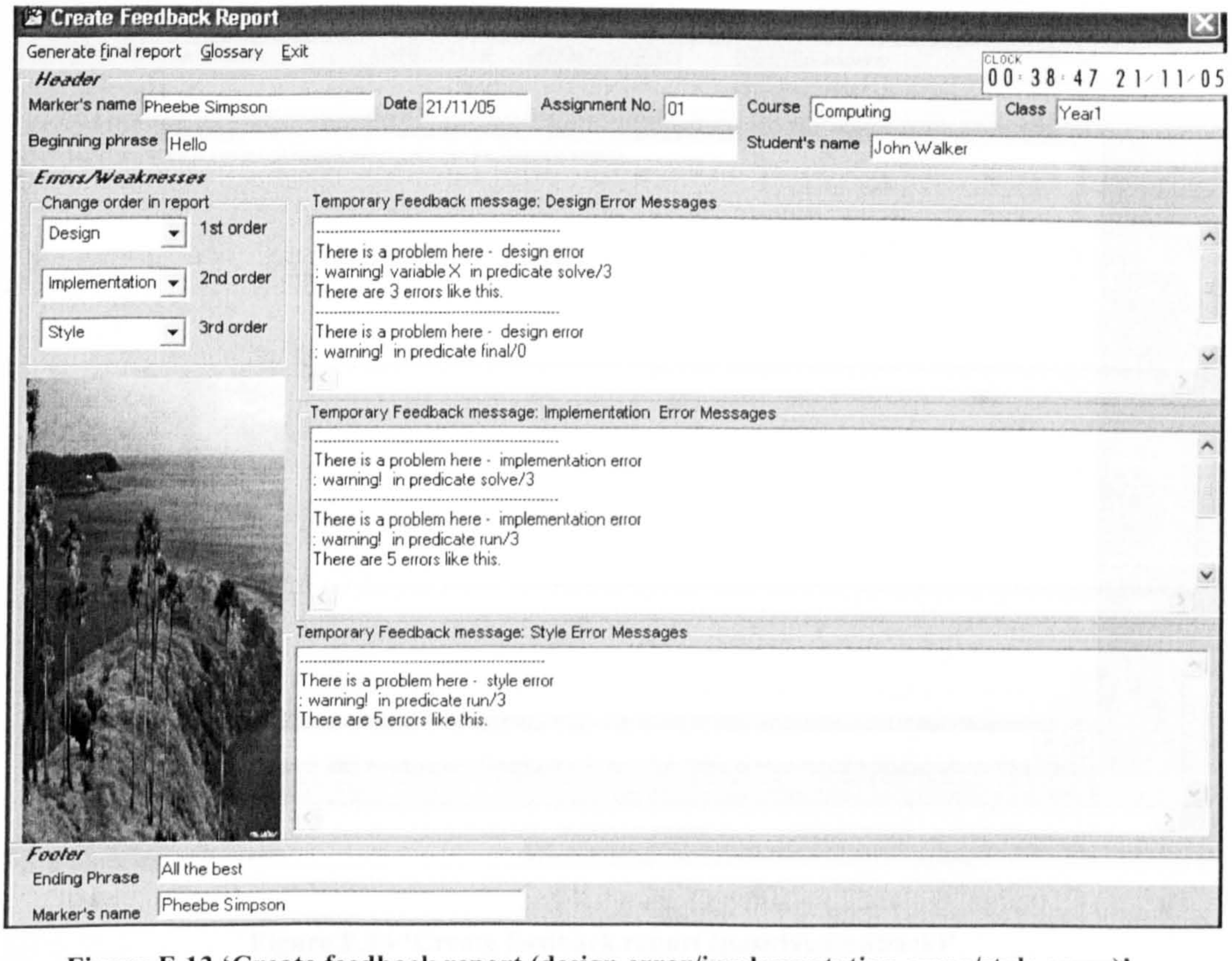


Figure E.13 'Create feedback report (design error/implementation error/style error)'

13. Assume that the right most lower template is pressed, the report template will be generated according to Figure E.14. Then go to 14.

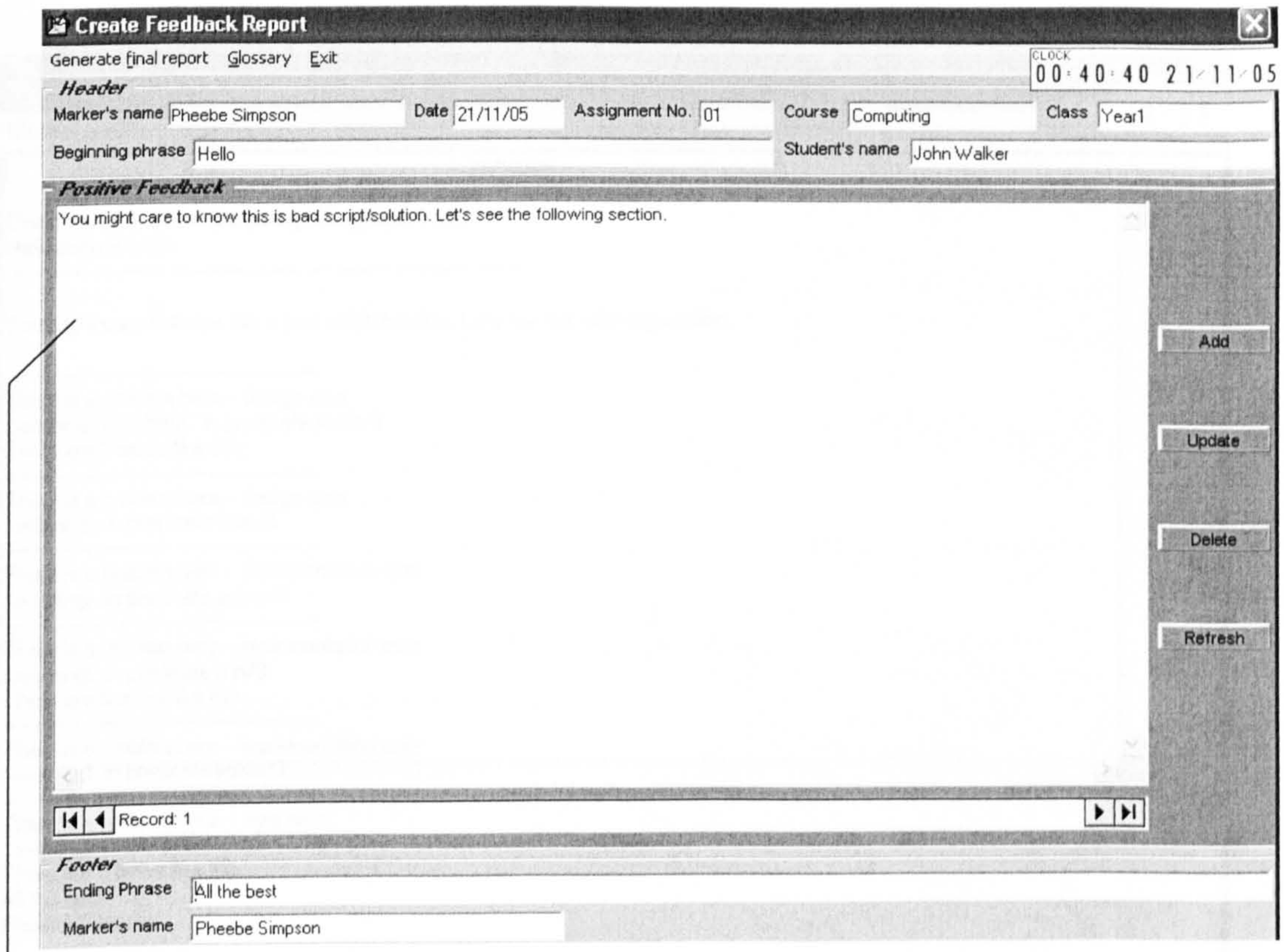


Figure E.14 'Create feedback report (positive feedback)'

Choice for possible 'Positive Feedback'

14. When you have finished organising the feedback report in 'Create feedback report' window, you will be asked to click 'Generate final report' menu to generate a draft of final report according to Figure E.15.

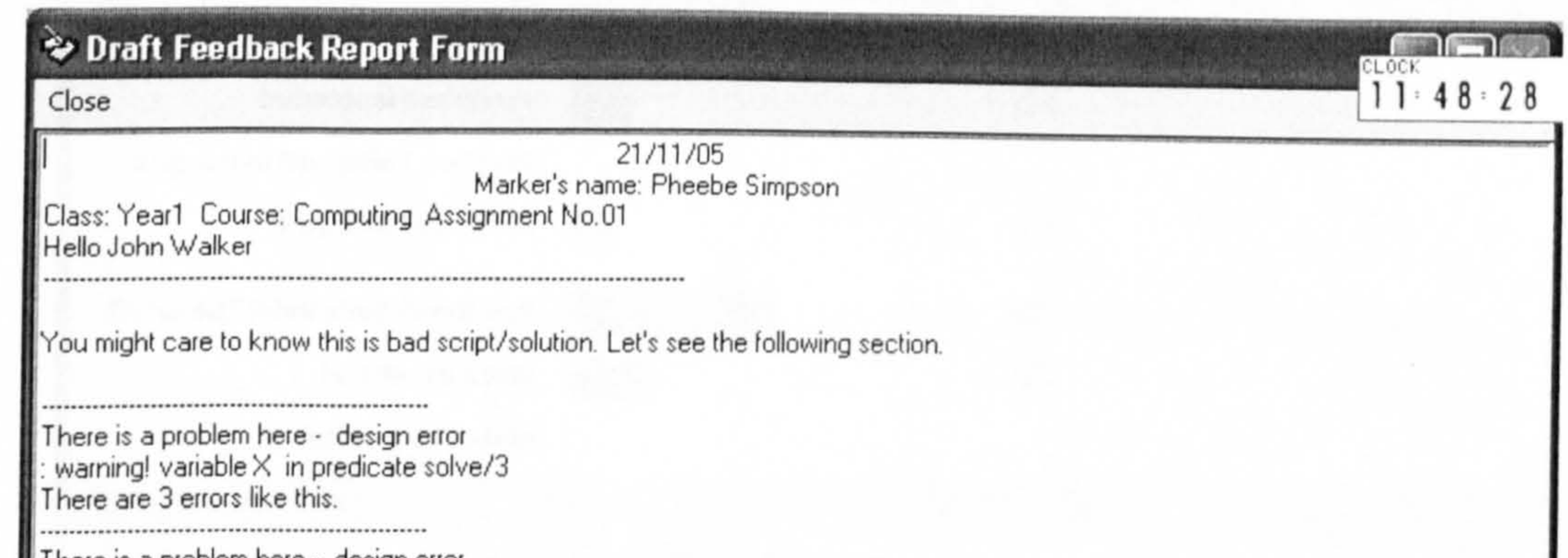


Figure E.15 'Draft feedback report form'

15. By now, you will be asked to select another two student's files: '4120330102.pl', and '4120380102.pl', and follow the instruction from 1 to 14. If you need to add error messages, you can do so.

16. To see your progress in giving feedback according to the measurement of various quality of feedback from the system, you can view the menu item: 'View', 'Skill Meter' as shown in Figure E.16.

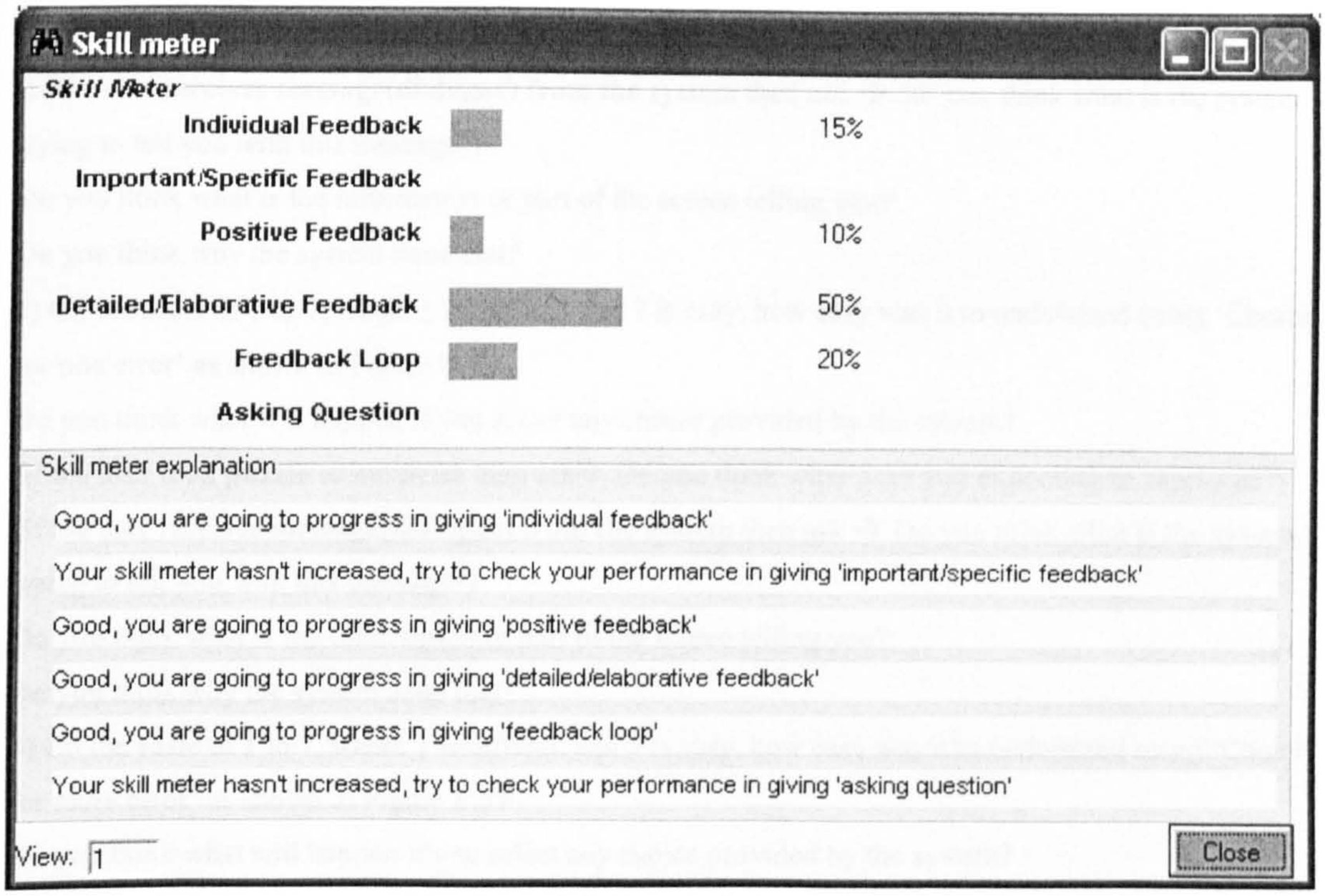


Figure E.16 'Skill meter'

E.2 Semi-structure interview sheet

1) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Favourite wording' as shown in Figure E.1?

Do you think what will happen if you double click the selected list item?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

2) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Manage data' as shown in Figure E.2?

Do you think what will happen if you change the detail of the selected list item?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

3) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Main interface of McFeSPA' as shown in Figure E.3?

Do you think what will happen if you double click the selected list item on the lower left handside?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

4) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Choice for one error' as shown in Figure E.4?

Do you think what will happen if you select any choice provided by the system?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

5) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Choice for more error' as shown in Figure E.5?

Do you think what will happen if you select any choice provided by the system?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

6) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Add extra sentence after error messages' as shown in Figure E.6?

Do you think what will happen if you select any choice provided by the system?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

7) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Add extra design error' as shown in Figure E.7?

Do you think what will happen if you click the middle arrow provided by the system?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

8) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Feedback Template' as shown in Figure E.8?

Do you think what will happen if you select any choice button provided by the system?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

9) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Create feedback report' as shown in Figure E.9-E.14?

Do you think what will happen if you click 'Generate final report' list item?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

10) On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Draft feedback report form' as shown in Figure E.15?

(If the user look **puzzle or surprise** then ask➔ Do you think what were you expecting to happened?)

(If the user receives **message(dialogue) from the system** then ask ➔ Do you think what is the system trying to tell you with this message?)

Do you think what is the information or part of the screen telling you?

Do you think why the system done that?

E.3 System checklist questionnaire

Heuristic Evaluation - A System Checklist

Please tick only one for each item and specify the reasons

1. I found the interface easy to use.
   ☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

   Comments:_____

2. I found it was very easy to select/manage the choice of feedback message, weakness message or positive feedback.
   ☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

   Comments:_____

3. I found it was very easy to manage the feedback message by edit/add/delete.
   ☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

   Comments:_____

4. I found it was very easy to organize the feedback report.
   ☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Comments:_____

5. I found it was very easy to edit the feedback report.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Comments:_____

6. I found it was very easy to generate the feedback report.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Comments:_____

7. I found it was very easy to customize McFeSPA
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Comments:_____

8. I found it was easy to read characters on the screen.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

9. Organisation of information is very clear.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

10. Sequence of screens is very clear.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

11. Use of terms throughout system is consistent.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

12. Terminology always related to task.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

13. Position of messages on screen is consistent.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

14. Error messages are helpful.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

15. McFeSPA always informs about its progress
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

16. Prompts for input is clear
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

17. Overall, I found it was simple to use McFeSPA.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree

Reason:_____

E.4 Observation sheet

1. How does the user add/update/delete data in 'Favourite Wording'/ 'Favourite Content'/ 'Report Template' menu from customise menu?
2. How does the user provide feedback message if there is no student's error analysed by the system?
3. How does the user provide extra sentences after selecting choice for error messages?
4. How does the user select a choice for one error (of the same error type) found?
5. How does the user select a choice for more errors (of the same error type) found?
6. How does the user generate feedback report from 'create feedback report' window to be a final report?
7. How does the user select feedback template?

Appendix F: Triangulation data from usability evaluation with McFeSPA

This appendix contains significant conversation between the developer and all evaluators, the summaries of evaluators' actions represented as tables, and algorithm of increasing the skill meter.

F.1 Conversation

Comment on the Customisation: 'Favourite wording' interface
Conversation#1
EV_1 "That is very easy. I'm quite happy with this predicate & variable"

Comment on the Customisation: 'ManageData' interface
Conversation#2
EV_1 said "Oh! I seem that I have lost one of them, not update the word";
EV_1 pressed the 'Update' button without selecting any list item;
Experimenter: "Actually, the 'Update' button will update the selected list";
EV_1: "OK, that is not clear because I thought it updated to my list not adding to my list";
Experimenter's comment: EV_1 made an error because EV_1 pressed the update button after changing a new word without selecting the list item that EV_1 need to update;
Experimenter: "You can see the message in this button on the 'Update' button"; EV_1: "I still think that might update on my list that I would select at that time")

Don't understand the help level 1
Conversation#3
EV_1: "I don't understand what does it mean? something in you selection doesn't work for you, because this may not be the feedback to give. I'm not sure which you help me the best feedback to give"
EV_1: "I don't understand why it doesn't work for me. I don't understand that phrase. It doesn't work for me."
Experimenter's comment: the EV_1 still comment on the help from the system while accepting the help
EV_1: "OK, the system giving me some idea how can I provide the best feedback and not explain how all different idea, not clear always"

Comment on the 'Choice for More errors' interface
Conversation#4
EV_1: "The interface is not clear that I can add my own information in there"
Experimenter: "You can add/save this information in the list of error message beyond the system."
EV_1: "But I don't have to because I want to put on information for a particular student. Can I put on other information in this that I don't want to save?"
Experimenter: "Yes, you can."
EV_1: "OK, the problem is I don't know in my solution where the errors are. When I want to look the error and I want to say something specific.
Conversation#5

EV_1: "When I click 'Yes' and what I am expecting to see, it is the 'student profile' and what am I guess is something to do with the sentence and when I tried to access the student's profile then I have to try to find the student"

Comment on the 'Main interface', the satisfaction scale = 4

Conversation# 6

EV_1: "because I don't know can I type in there (Design/Implementation/Style pane and I can't see what problems are but when I went to add any thing extra. It is difficult for me because I don't know in the solution where the problem has been found. OK."

Comment on the 'Choices for More errors' interface

Conversation# 7

EV_1: "I know when I go click on 'None' I'm going to get trouble as well"

Conversation# 8

EV_1: "I'm a bit surprise two options here 'indicate the error line number' and 'include an example' I can't understand why you need to repeat this to every option because you will waste some spaces. I don't understand why this include an example" and this 'include an example at the end'. It's not clear to me."

Experimenter's comment: 'include an example' is the option for providing an example for the error message only once and 'include an example at the end' is the option for providing one at the end of the all error messages, not after of each error messages.

Comment on the 'Create Feedback Report' interface, satisfaction scale = 3

Conversation#9

EV_1: "I think this interface here is quite poor because it's giving me some feedback. Yes, it's helpful. It's helping me but I have to keep on pressing 'Ok'. It's very frustrating."

Conversation#10

EV_1: "I have ever checked the feedback report before and now I don't want to check it again. I want to generate the report so I press 'No' but I can't see where the report is. So the problem is I press 'No', nothing happened. So I need to know that what happen when I press that 'No'. I don't know actually when the report goes to the student/ email to student."

Conversation#11

Experimenter: "On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Create Feedback Report' interface".

EV_1: "This interface is very confusing so I can give it three because the way I can change 'Positive feedback' very different from the way I can change the other thing"

Conversation#12

EV_1: "'Exit' means to me, I'm leaving the whole program"

Comment on the 'Draft feedback report' interface

Conversation#13

EV_1: "because it didn't change my name. Not very easy because I can't see the button (close)"

Conversation#14

EV_1: "I expect to turn off the interface with the cross button on the top"

Comment on the 'Add Extra Design', satisfaction scale = 3

Conversation#15

EV_1: "I don't know what is the different between the left hand side and the right hand side."

Comment on the 'Skill meter' interface
Conversation#16
EV_1: "I don't understand. It said 65% but I am not sure 65% of what, is that 65% of all opportunities that I have to give feedback or 65% of the feedback I gave, with individual feedback I gave. Not easy to work with"

Comment on whole system for using 2nd time to mark the 2nd script
Conversation #17
Experimenter: On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the system while you uses the system 2nd time to mark the 2nd script. This is the 2nd time you are using the system.
EV1: "Oh! Much easier because you explain to me at the 1st time rather than what I didn't understand. I think if I come to this without you to explain it, it would take a lot longer to learn it and because it's easier in the 2nd time rather than the 1st time. The 2nd time I gave 5. The 1st time I gave 3. I don't have any thing to worry about to compare with the 1st time.

Final comment
Conversation #18
EV_1: "Well, the important thing about this is that it is a very good and very useful system, a lot of work to program it, which is really good. The problems are small things that give a negative impression when using the system, they are not big things. There is small thing that is difficult to use, i.e. don't having the cross button at the top of the window to close, go to the different record of each student's instead of scrolling, going to the different record of positive feedback instead of scrolling, the confusing between add and update button. So it is small interface thing that make it is difficult to use. In term of it's big overall scheme, easier to use than having my feedback on this. So I have been quite critical because I am critical a small thing that made me frustrating but I think the small thing which is the simple thing to be fixed but the important thing that is further coming. The whole system as a whole will be a lot easier to use".

Comment on the 'ManageData' interface
Conversation#19
EV_2: "'Move', 'Add', 'Update', and 'Clear' button to the left hand side and 'Close' button to the right hand side. The 'Delete' button worked on the list not the text box so you should move it under the list box".
Conversation#20
EV_2: "It's not difficult to use. May be a bit confusing you can add. I understand that you can add something here but you also using the something with modify one"

Comment on the 'Favourite Wording' interface
Conversation#21
EV_2: "I understand the 'Initial statement' and 'Warning statement' but 'Predicate' I don't actually know what does it mean… you should put the example of using each word… 'Between' what, 'Between' should be 'Between line', and 'And' be 'And line'"

Comment on the 'open student' solution file' window in the 'Main' interface

Conversation#22
EV_2: "I don't know the name mean something but you should think about using different name. It's very difficult to look at the student's file. I have to look at digit by digit unless you not put the number, just put the name or something I can memorise immediately, put the name of the student, for example."
Experiment: "Student's name here mostly, they have same name, same surname."
EV_2: Laugh, "I know what you mean but it's difficult to spot which file that I have to open"

Comment on 'Analysed student's solution' pane in the 'Main' interface & 'Choices for more errors' interface
Conversation#23
EV_2: "Can I redo it again?"
Experiment: "Yes, go on"
EV_2: "No, I made a mistake. I just want to do it. I just want to close the window to make a choice."
Experiment: "I have ever made it but it is very complicated then I am going to update in the next version."
Conversation#24
EV_2: "You have to make sure that if I don't want to continue. I want to cancel. You should provide any cancel button. I can close it down without making the selection. Now I am stuck. I have to make a selection."
EV_2: "OK, I understand what it happen"

Comment on the 'Show student's solution' pane in the 'Main' interface
Conversation#25
EV_2: "I would try to put the number at the beginning because my feeling line number on the left not the right. If you think about familiarity, most people look for the line number on the left so try to think about that"

Comment on the 'Student's profile' menu item in 'Choices for one error' interface
Conversation#26
EV_2: "There is something a bit strange here"
Experimenter: "I should not offer that"
EV_2: "Oh! You should not offer because I do not suppose to look at it now, right. Ok. No problem"

Comment on the pop-up message of Help level 1
Conversation#27
EV_2: "I still don't understand why something is not working for me, why the system is telling me that. You may give me more information."
Experimenter: "To encourage you to give quality feedback"
EV_2: "Why?"
Experimenter: "When you try to do the process, it will gradually help you provide the answer to you"
Conversation#28
EV_2: "The system not let me choose the option. Why something in you selection doesn't work for me. Why that?"
Experimenter: "Because you are in the scaffolding mode. Thus the system will help you to select the appropriate option"

EV_2: "You should say something positively e.g. Yeah, Ok, but you should do better than that."
EV_2: "Don't say something in you selection doesn't work for you"

Comment on the 'Student's profile' interface
Conversation#29
EV_2: "You should have this for me one student in particular so in that case when I open the student's profile. I should expect that student in particular already selected."

Comment on the defaulted option in the 'Choices for more errors' interface
Conversation#30
EV_2: "I select this one while I know it doesn't work but I'm pretending. I want to see why.

Comment on English used in 'Choices for more errors' interface
Conversation#31
EV_2: "You need to be careful in using English. Get someone to work with you e.g. Yes (just) once, Yes (Always) I don't understand what does it mean? between two options here.

Change the title of 'Add extra sentence after error message' interface
Conversation#32
EV_2: "The sentence you are going to add are about the learner's history so, the title should be taking into account the learner's history, something like that."

Delete 'Refresh' button from 'Add extra design' interface
Conversation#33
EV_2: "What is the refresh?
Experimenter: "It is going to the 1st record."
EV_2: "I know in the database but the people who don't know the database may not know this. They don't know how different between 'Update' and 'Refresh'. I'm not sure 'Refresh' is useful in this stage but 'Refresh' may be not need it. OK, that's fine.
Experimenter: "On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'Add extra design' interface".
EV_2: "6"

Commented on 'Say nothing' option in 'Choices for one/more errors' interface
Conversation#34
EV_2: "Missing indentation" is not important thus the system should allow the user to choose the 'Say nothing' option."
Experimenter: "We have designed this and allow the user to select any options throughout the system in the case of scaffold-off if the user doesn't want to select some options several times."

'Include an example' checkbox in 'Choices for one/more errors' interface
Conversation#35
EV_2: "The example that you put here is not clear at all because it's very low why the example, the example of what. This is the whole program, doesn't show any thing."
Experimenter: "The example of the right answer of this solution"
EV_2: "Yes, but you put the whole solution, you put every thing so what you call here

that is not an example, of the right answer. If you say here 'put the solution of the right answer, not for the whole program"

Change 'Exit' menu item to 'Close' menu item
Conversation#36
EV_2: "You have to change this because exit means exit the whole program, not close this window"

Update viewing 'Temporary of feedback messages' pane in Create feedback report interface.
Conversation#37
EV_2: "You have to sort out this problem. You need to change this interface. You need to put the window here as read-only. You should not have this message. It's not good. It's confusing. You should program to prevent edition here as disable or read-only, not allow edit, avoid that message. This message is useless; very distracting. If you don't want people to edit here, then disable edition"

Comment on the message Pop-up "Good, you are going to progress in …"
Conversation#38
EV_2: "You need to change the language used e.g. 'You have made progress on giving 'positive feedback' instead of 'Good you are going to process in giving 'positive feedback'"
EV_2: "I see the 1st one. It's fine. Put only the single one, put all together in a single interface because it's shutdown the process. You can't do any thing until you click on. That is very difficult"

Commented on from 'Create feedback report' interface to 'Draft feedback report' interface
Conversation#39
EV_2: "Where is the report?
Experimenter: "The report in the file."
EV_2: "Yeah, the report. OK, Can I print it?"
Experimenter: "In this version, from the design, you can send email to the student.
EV_2: "I'm not sure what I should do about this interface"

Comment on the 'Create feedback report' interface
Conversation#40
Experimenter: "How about this interface?"
EV_2: "That is complex. It is very change interface, very overload."

Temporary feedback message in 'Create feedback report' interface
Conversation#41
EV_2: "I know it is very difficult to charge it but you have to find a way. Having a whole text here may be not a good thing because it takes space. What you should have – a button to preview small window, not all report."

Change order in report in 'Create feedback report' interface
Conversation#42
EV_2: "You should protect that possibility. The system should adapt it automatically if the user chooses the duplicate one"

Comment on the 'Skill meter' interface
Conversation#43
EV_2: "30% of individual feedback what. I don't understand what this problem is?"
Experimenter: "30% of all individual feedback"
EV_2: "Why 30% is? What is the percentage? I don't know what the 30% mean. How the learner get to 100%. The importance is the meaning of this/30%. You have to work with someone. You have to ask someone. Give the explanation and provide the key value"

Don't understand the meaning of word in 'Favourite wording' interface
Conversation#44
EV_3 "I don't know exactly what the meaning of this is?
EV_3 "The initial statement is negative, have 'problem' word"

Comment on the 'ManageData' interface
Conversation#45
Experimenter: "On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using 'ManageData' interface".
EV_3: "Oh! That's very easy".

Can't see the 'student profile' menu item
Conversation#46
Experimenter suggested the 'student profile' menu item to EV_3 and suggested EV_3 to compare the previous error with the current error.
EV_3: "OK! It's here. I didn't realize it"

Comment on the 'Temporary feedback message' in 'Create feedback report' interface
Conversation#47
EV_3: "Why don't you put a single window with all together so that I can see the report?"
Experimenter: "All right, thank you very much."

Comment on the 'Create Feedback Report' interface
Conversation#48
EV_3: "I didn't know this, not clear"
Experimenter: "Oh! This is the beginning phrase… The system allow the user to customise"
EV_3: "The meaning is not clear because I don't know the beginning phrase can include the learner's name. I don't know can I put the whole beginning phrase with student's name 'Hello John', just 'Hello'"

Comment on the message Pop-up "Good, you are going to progress in …"
Conversation#49
EV_3: "Too many messages, get annoying me"

Misunderstood the message pop-up "Double check…." in the 'Create feedback report' interface
Conversation#50
EV_3: "What they came to my mind. I didn't check the student's history account."

Experimenter: "Oh! You'd like to check student's account here"
EV_3: "I don't know the explanation of this"
Experimenter: "I'd like to inform the user to check the report before generating the final report.""
EV_3: "OK".

Comment on the 'Feedback Template' interface
Conversation#51
Experimenter: "How about using this interface?"
EV_3: "This is very easy to understand but it's difficult at first when I select the button"

Comment on the 'Skill meter' interface (after view the 'skill meter' interface)
Conversation#52
EV_3: "Too many repetition of this thing"

Comment on using whole system in the 2nd time to mark the 2nd script
Conversation#53
Experimenter: "On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the system while you uses the system 2nd time to mark the 2nd script. This is the 2nd time you are using the system."
EV_3: "Oh! Much easier"

Comment on using the 'Student's Profile' interface
Conversation#54
EV_1: "It's not clear where I should the student's history of error; Oh! I see, the name has changing, I can change the student's history of student's error
.....

F.2 Data collection from questionnaires from evaluators

F.2.1 Evaluator 1 (EV1)'s questionnaire results

Heuristic Evaluation - A System Checklist

Please tick only one for each item and specify the reasons

1. I found the interface easy to use.
   ☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
   Comments: Only with help on understanding what it meant

2. I found it was very easy to select/manage the choice of feedback message, weakness message or positive feedback.
   ☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
   Comments: Different methods

3. I found it was very easy to manage the feedback message by edit/add/delete.
   ☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☑ strongly disagree
   Comments: "Update" not clear meaning

4. I found it was very easy to organize the feedback report.
   ☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree

Comments: No: The options are too dichorial: giving options that you are not allowed!

5. I found it was very easy to edit the feedback report.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments:_____

6. I found it was very easy to generate the feedback report.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Comments: Not clear what actually happened at the end.

7. I found it was very easy to customize McFeSPA
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments:_____

8. I found it was easy to read characters on the screen.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

9. Organisation of information is very clear.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason: strange to flip through students and positive feedback rather than scroll.

10. Sequence of screens is very clear.
☐ strongly agree ☐ agree ☐ not sure ☐ disagree ☑ strongly disagree
Reason: The information about the particular student was hard to get.

11. Use of terms throughout system is consistent.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

12. Terminology always related to task.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason: Not sure what some of the dialogue boxes mean!

13. Position of messages on screen is consistent.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

14. Error messages are helpful.
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: No: often they are just frustrating. It is good to get feedback, but not so that it continuously affects the flow of the task!.

15. McFeSPA always informs about its progress
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: It was not clear what happened at the end!

16. Prompts for input is clear
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason:_____

17. Overall, I found it was simple to use McFeSPA.
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: Mainly, it was good, and a very good idea – it was, as always, the small things that gave me a negative impression. Overall, I think it is a good, worthwhile system- the small usability interface problems would be easily be fixed.

F.2.2 Evaluator 2 (EV2)'s questionnaire results

Heuristic Evaluation - A System Checklist

Please tick only one for each item and specify the reasons

18. I found the interface easy to use.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: Once you know what to do

19. I found it was very easy to select/manage the choice of feedback message, weakness message or positive feedback.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Comments: _____

20. I found it was very easy to manage the feedback message by edit/add/delete.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

21. I found it was very easy to organize the feedback report.
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Comments: _____

22. I found it was very easy to edit the feedback report.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

23. I found it was very easy to generate the feedback report.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

24. I found it was very easy to customize McFeSPA
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Comments: Not sure where customization is different from adding new template in reports

25. I found it was easy to read characters on the screen.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: _____

26. Organisation of information is very clear.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason: _____

27. Sequence of screens is very clear.
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: some are too big.

28. Use of terms throughout system is consistent.
☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: need to get English nature to rewrite interface

29. Terminology always related to task.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: _____

30. Position of messages on screen is consistent.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason: _____

31. Error messages are helpful.

☐ strongly agree ☐ agree ☐ not sure ☑ disagree ☐ strongly disagree
Reason: problem of language mostly

32. McFeSPA always informs about its progress
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: but in a very annoying way

33. Prompts for input is clear
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: _____

34. Overall, I found it was simple to use McFeSPA.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: The problem is initial explanation of tasks.

F.2.3 Evaluator 3 (EV_3)'s questionnaire results

Heuristic Evaluation - A System Checklist

Please tick only one for each item and specify the reasons

35. I found the interface easy to use.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

36. I found it was very easy to select/manage the choice of feedback message, weakness message or positive feedback.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Comments: _____

37. I found it was very easy to manage the feedback message by edit/add/delete.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

38. I found it was very easy to organize the feedback report.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

39. I found it was very easy to edit the feedback report.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

40. I found it was very easy to generate the feedback report.
☑ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

41. I found it was very easy to customize McFeSPA
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Comments: _____

42. I found it was easy to read characters on the screen.
☑ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: _____

43. Organisation of information is very clear.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason: _____

44. Sequence of screens is very clear.

☑ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

45. Use of terms throughout system is consistent.
☑ strongly agree ☐ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

46. Terminology always related to task.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

47. Position of messages on screen is consistent.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

48. Error messages are helpful.
☐ strongly agree ☐ agree ☑ not sure ☐ disagree ☐ strongly disagree
Reason: Feedback on progress was a little annoying

49. McFeSPA always informs about its progress
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

50. Prompts for input is clear
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

51. Overall, I found it was simple to use McFeSPA.
☐ strongly agree ☑ agree ☐ not sure ☐ disagree ☐ strongly disagree
Reason:_____

F.3 Summaries of evaluators' actions

No. of actions	Refuse Help (Hint#1)									Accept Help (Hint#1)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	1	0	0	0	0	0	0	0	0	4	0	0	3	0	0	2	0	0

Table F.1 Summary of the actions from all evaluators involved in Hint#1

No. of actions	Refuse Help (Hint#2)									Accept Help (Hint#2)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table F.2 Summary of the actions from all evaluators involved in Hint#2

No. of actions	Refuse Help (Hint#3)									Accept Help (Hint#3)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0

Table F.3 Summary of the actions from all evaluators involved in Hint#3

No. of actions	Refuse Help (Hint#4)									Accept Help (Hint#4)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Table F.4 Summary of the actions from all evaluators involved in Hint#4

No. of actions	Refuse Help (Hint#5)									Accept Help (Hint#5)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	5	0	0	6	1	2	0	0	0	4	0	0	0	1	0	0	0	0

TableF.5 Summary of the actions from all evaluators involved in Hint#5

No. of actions	Refuse Help (Hint#6)									Accept Help (Hint#6)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	1	0	0	0	0	0	0	2	0	4	0	0	2	0	0

TableF.6 Summary of the actions from all evaluators involved in Hint#6

No. of actions	Refuse Help (Hint#7)									Accept Help (Hint#7)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	1	1	1	1	1	0	0	0	0	1	0	0	2	0	4	0	0

Table F.7 Summary of the actions from all evaluators involved in Hint#7

No. of actions	Refuse Help (Hint#8)									Accept Help (Hint#8)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

TableF.8 Summary of the actions from all evaluators involved in Hint#8

There are no any actions from all evaluators involved in Hint#9 and Hint#10

No. of actions	Refuse Help (Hint#11)									Accept Help (Hint#11)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
								1	1									

Table F.9 Summary of the actions from all evaluators involved in Hint#11

No. of actions	Refuse Help (Hint#12)									Accept Help (Hint#12)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	4	0	0	0	0	0	0	3	0	1	3	0	0	0	0

Table F.10 Summary of the actions from all evaluators involved in Hint#12

No. of Actions	View student's profile									View skill meter									View glossary								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	3	0	2	3	1	0	9	2	1	0	0	3	4	1	1	2	1	1	1	0	0	3	0	0	0	0	0

Table F.11 Summary of the actions from all evaluators involved in viewing the student's profile, viewing the skill meter, and viewing the glossary

No. of Actions	Success									Error																	
	EV_1			EV_2			EV_3			Refuse Help									Accept Help								
										EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	6	2	0	1	1	0	3	2	0	2	0	0	1	0	0	0	0	0	0	2	0	4	1	0	2	0	0

Table F.12 Summary of the actions from all evaluators involved in the 'Choices for more errors' interface

No. of Actions	Success									Error															
	EV_1			EV_2			EV_3			Refuse Help									Accept Help							
										EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint
	5	0	4	0	1	2	3	0	4	3	0	0	2	0	2	0	0	0	3	0	0	0	0	0	0	0	0

Table F.13 Summary of the actions involved in the 'Choices for one error' interface from all evaluators

	Success									Error															
	EV_1			EV_2			EV_3			Refuse Help									Accept Help							
										EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint
	2	0	1	1	1	0	6	2	1	1	1	0	0	0	0	0	0	0	5	0	2	3	1	0	2	0	0

Table F.14 Summary of the actions involved in the 'Add extra sentence after error messages' interface from all evaluators

No. of Actions	Success									Error															
	EV_1			EV_2			EV_3			Refuse Help									Accept Help							
										EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint	1st scrint	2nd scrint	3rd scrint
	2	0	0	0	0	0	1	1	1	0	1	1	1	1	1	0	0	0	0	1	0	0	2	0	4	0	0

Table F.15 Summary of the actions involved in the 'Feedback Template' interface from all evaluators

No. of Actions	Success									Error																	
	EV_1			EV_2			EV_3			Refuse Help									Accept Help								
										EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	9	0	1	0	1	1	2	1	2	0	0	0	4	0	0	0	2	1	0	3	0	1	3	0	0	0	0

Table F.16 Summary of the actions involved in the 'Create Feedback Report' interface from all evaluators

No. of	None (just brief message)									Yes (just once)									Yes (Always)								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	2	0	0	1	0	0	0	0	0	5	2	0	2	1	0	3	2	0	0	1	0	2	1	0	1	0	0

Table F.17 Summary of the actions involved in selecting the option of the 'Choices for more errors' interface from all evaluators

No. of Actions	Say nothing									Indicate the error line number									Include an example								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	2	1	0	0	0	0	3	1	0	2	1	0	2	2	0	4	2	0	1	0	0	1	1	0

Table F.18 Summary of the actions involved in selecting the option of the 'Choices for more errors' interface from all evaluators (contd.)

	None (just brief message)									Yes									Say nothing								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	3	0	0	2	0	2	0	0	0	3	0	4	0	1	2	3	0	4	1	0	0	1	0	0	0	0	0

Table F.19 Summary of the actions involved in selecting the option of the 'Choices for one error' interface from all evaluators

No. of Actions	Indicate the error line number									Include an example								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	2	0	4	0	1	2	2	0	3	2	0	2	0	0	0	1	0	0

Table F.20 Summary of the actions involved in selecting the option of the 'Choices for one error' interface from all evaluators (contd.)

No. of actions	Managing error message by adding									Managing error message by updating								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	2	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0

Table F.21 Summary of the actions involved in 'Add Extra Design Error' interface from all evaluators

No. of actions	Managing error message by deleting									Add the error message to the feedback report								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	1	0

Table F.22 Summary of the actions involved in 'Add Extra Design Error' interface from all evaluators (contd.)

The system offer 'Add Design Error' interface to the EV_2 but he refused to add any design errors.

No. of actions	Managing error message by adding									Managing error message by updating								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table F.23 Summary of the actions involved in 'Add Extra Implement Error' interface from all evaluators

No. of actions	Managing error message by deleting									Add the error message to the feedback report								
	EV_1			EV_2			EV_3			EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table F.24 Summary of the actions involved in 'Add Extra Implement Error' interface from all evaluators (contd.)

There is only the EV_2 added the implement error to the feedback report. The system offered 'Add Design Error' interface to all evaluators but they refused to add any style errors.

No. of actions	Managing data by adding			Managing data by updating			Managing data by deleting		
	EV_1	EV_2	EV_3	EV_1	EV_2	EV_3	EV_1	EV_2	EV_3
	4	1	1	1	1	2	1	2	1

Table F.25 Summary of the actions involved in 'customise' interface ('Favourite wording', 'Report Template' interface) from all evaluators

No. of Actions	Click the list item (of 'analysed student's solution' pane in the main interface)								
	EV_1			EV_2			EV_3		
	1st script	2nd script	3rd script	1st script	2nd script	3rd script	1st script	2nd script	3rd script
	15	2	4	8	3	4	6	2	4

Table F.26 Summary of the actions from all evaluators involved in clicking the list item (of 'analysed student's solution' pane in the main interface) and clicking 'Generate final report' menu item (of the 'Create Feedback Report' interface) (contd.)

Interface / Evaluator	Favourite Wording	ManageData	Main	Choices for More Errors	Choices for One Error	Add Extra Sentence	Add Extra Design	Feedback Template	Create Feedback Report	Draft Feedback Report	Skill Meter	Overall at 1st marking	Overall at 2nd marking	Overall at 3rd marking
EV_1	5	6	4	5	6	2	3	6	3	5	3	3	5	5
EV_2	4	5	5	6	6	5	6	6	3	7	3	3	6	6
EV_3	4	7	6	5	5	6	6	6	6	7	7	5	6	6

Table F.27 Satisfied scales (On the scale of 1 to 7, where 1 is difficult and 7 is easy, how easy was it to understand using the interface) by all evaluators for each interface and overall after finishing marking each student's solution file

UNIVERSITY *of* GLASGOW

Faculty of Education

Ethics Committee For Non Clinical Research Involving Human Subjects

EAP2 NOTIFICATION OF ETHICS APPLICATION FORM APPROVAL

Application No. *(Research Office use only)*	E624
Period of Approval *(Research Office use only)*	20.06.06.06 to 29.09.06

Date: 29 June 2006

Dear Duenpen

I am writing to advise you that your EAP1 application for ethical approval, reference E624 for 'Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship' has been approved, following your fulfilment of the conditions given in the earlier notification.

You should retain this approval notification for future reference. If you have any queries please do not hesitate to contact me in the Research Office and I will refer them to the Faculty's Ethics Committee.

Regards

Terri Hume
Ethics and Research Secretary

Faculty of Education Research Office
University of Glasgow
Room 425D
St Andrew's Building, 11 Eldon Street, Glasgow, G3 6NH

Tel: 0141-330-4101
Fax: 0141-330-3005
E-mail: F.Mackinlay@admin.gla.ac.uk

Appendix G: Materials for Evaluation of McFeSPA's Learning Environment

This appendix presents useful materials for evaluation of learning environment i.e. pre-test paper (post-test paper is similar to the pre-test paper but it is in the reverse sequence), questionnaire sheet, and structured interview sheet. Observation sheet in this evaluation is same as Appendix E. Handout for evaluation of McFeSPA's learning environment with scaffolding is quite similar to the handout for evaluation of the usability of the interface (in Appendix E) so we do not present in this appendix. The handout in this evaluation does not contain part of using McFeSPA's customisation. There are two types of handout. One is for comparison group which uses the system without scaffolding. Another is for the experimental group which use the system with scaffolding and followed by use the system without scaffolding. The materials can be seen in the following.

G.1 Pre-test paper

Marking a Prolog programming assignment

Please read the problems, the scripts provided (Figure G.2), the brief error messages given by the system (Figure G.3), and the records of the students (Figure G.4). Then please do the tasks.

Problem: *Mr. John Walker is the 1st year undergraduate student in school of computing in a University of Scotland. He has submitted for the 2nd time the 1st Prolog programming assignment (his script is shown in Figure G.2) which the correct solution is shown in Figure G.1.*

```
/* 1 */  run(A, B, C) :-
/* 2 */      D is B*B - 4*A*C,
/* 3 */      solve(A, B, D),
/* 4 */      final.
/* 5 */
/* 6*/  solve(_, _, D) :-
/* 7 */      D < 0,
/* 8 */      write('No solution'),
/* 9 */      !.
/*10 */  solve(A, B, D) :-
/*11 */      D = 0,
/*12 */      X is -B/ 2*A,
/*13 */      write(' x = '),
/*14 */      write(X),
/*15 */      !.
/*16 */  solve(A, B, D) :-
/*17 */      S is sqrt(D),
/*18 */      X1 is (-B + S) / 2*A,
/*19 */      X2 is (-B  - S) / 2*A,
/*20 */      write('x1 = '),
/*21 */      write(X1),
/*22 */      write(' and x2 = '),
/*23 */      write(X2).
/*24 */
/*25 */  final:-
/*26 */      nl,
/*27 */      write('good bye').
```

Figure G.1 Correct solution

□ **4120390102.pl, [Mr. John Walker]**

```
/* 1 */      run(A, B, C) :-
/* 2 */              D is B*B - 4*A*C;
/* 3 */              solve(A, B, D).
/* 4 */
/* 5*/       solve(_, _, D) :-
/* 6 */              D < 0;
/* 7 */              write('No solution');
/* 8 */              !.
/* 9 */      solve(A, B, D) :-
/*10 */              D = 0;
/*11 */              X is -B/ 2*A;
/*12 */              write(' x = ').
/*13 */      solve(A, B, D) :-
/*14 */              S is sqrt(D);
/*15 */              X1 is -B + S / 2*A;
/*16 */              X2 is -B  - S / 2*A;
/*17 */              write('x1 = ');
/*18 */              write(' and x2 = ').
/*19 */
/*20 */      final:-
/*21 */              nl;
/*22 */              write('good bye').
```

Figure G.2 Script: *His script and the result of analysed solution are below.*

No. Errors	Error Type	Error Name
3	Design	Unused variable
1	Design	Unreachable goal
1	Implementation	Missing Cut (!)
5	Implementation	Writing separate symbol (; not,)
1	Implementation	Missing parenthesis
5	Style	Missing indentation

Figure G.3 Brief error messages provided by the system
(Analysed errors from 4120390102.pl [Mr. John Walker])

There are 3 errors of unused variable.

There are 4 errors of unreachable goal.

There is no error of missing cut (!).

There are 2 errors of writing separate symbol (; not ,)

There is no error of missing parenthesis.

There are 3 errors of missing indentation

Figure G.4 Student Record

(John Walker's history of error of the 1st submission of the 1st assignment)

Choices below is 'Taking into history of student's errors'

A. This is the 1st time you have made this error; hopefully next time you could avoid this type of error.

B. This is only the 2nd time you have made this error; however, you made more errors of this type than previously. Hopefully next time you could avoid this type of error.

C. Well done, you made this error less than previously even though this is the 2nd time you have made this error. Hopefully next time you could avoid this type of error.

D. This is the 2nd time you have made this error; hopefully next time you could avoid this type of error.

E. Say nothing

Figure G.5 Taking into history of student's errors

Task:

Assume that you are a teaching assistant (TA) for a lecturer and you are marking Mr. John Walker's assignment (as shown in Figure G.2) with the help of brief error messages (as shown in Figure G.3) provided by the programming analyser as above. Your lecturer has suggested you provide feedback such as detailed feedback, positive feedback, etc. in the student's feedback report. Please select the best choice for the feedback messages to be provided to the student.

1) There are **3 design errors of 'Unused variable'** in the current submission (2nd submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(1a)** and **(1b)**.

(1a) Which of these feedback messages do you prefer?
Please indicate your 1st preference with a "1" in the box, 2nd preference with a "2" in the box, 3rd preference with a "3" in the box, and so on.

☐ Between line 9 and line 12: warning! design error

☐ There is a problem here - design error
Between line 9 and line 12: warning! variable X in predicate solve/3
There are 3 errors like this.

☐ There is a problem here - design error
Between line 9 and line 12: warning! variable X in predicate solve/3 have not been used. It should appear more than once or be underscore(_) when no mention to any variable in that goal.
There are 3 errors like this.

☐ There is a problem here - design error
Between line 9 and line 12: warning! variable **X** in predicate solve/3 have not been used. It should appear more than once or be underscore(_) when no mention to any variable in that goal.
Between line 13 and line 18: warning! variable **X1** in predicate solve/3 have not been used. It should appear more than once or be underscore(_) when no mention to any variable in that goal.
Between line 13 and line 18: warning! variable **X2** in predicate solve/3 have not been used. It should appear more than once or be underscore(_) when no mention to any variable in that goal.

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1st choice? Please add any explanation for your choice:

(1b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

2) There is **a design error of 'Unreachable goal'** in the current submission (2nd submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(2a)** and **(2b)**.

(2a) Which of these feedback messages do you prefer?
Please indicate your 1st preference with a "1" in the box, 2nd preference with a "2" in the box, 3rd preference with a "3" in the box, and so on.

☐ Between line 20 and line 22: warning! design error

☐ There is a problem here - design error
Between line 20 and line 22: warning! in predicate final/0

☐ There is a problem here - design error
Between line 20 and line 22: warning! in predicate final/0 is not reachable.
Your program never calls it at all. I suspect that you may forget to call such goal.

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1st choice? Please add any explanation for your choice:

(2b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

3) There is **an implementation error of 'Missing Cut (!)'** in the current submission (2^{nd} submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(3a)** and **(3b)**.

(3a) Which of these feedback messages do you prefer?
Please indicate your 1^{st} preference with a "1" in the box, 2^{nd} preference with a "2" in the box, 3^{rd} preference with a "3" in the box, and so on.

☐ Between line 9 and line 12: warning! implementation error

☐ There is a problem here - implementation error
Between line 9 and line 12: warning! in predicate solve/3

☐ There is a problem here - implementation error
Between line 9 and line 12: warning! You are forgetting the cut(!) symbol in line 12 in predicate solve/3. It can cause execute in the next goal in which it may be wrong solution in some cases.

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1^{st} choice? Please add any explanation for your choice:

(3b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

4) There are **5 implementation errors of 'Writing separate symbol (; not,)'** in the current submission (2nd submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(4a)** and **(4b)**.

(4a) Which of these feedback messages do you prefer?
Please indicate your 1st preference with a "1" in the box, 2nd preference with a "2" in the box, 3rd preference with a "3" in the box, and so on.

☐ Between line 1 and line 4: warning! implementation error

☐ There is a problem here - implementation error
Between line 1 and 4: warning! in predicate run/3
There are 5 errors like this.

☐ There is a problem here - implementation error
Between line 1 and line 4: warning! You should change semicolon symbol (;) in predicate run/3 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.
There are 5 errors like this.

☐ There is a problem here - implementation error
Between line 1 and line 4: warning! You should change semicolon symbol (;) in predicate run/3 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.
Between line 5 and line 8: warning! You should change semicolon symbol (;) in predicate solve/3 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.
Between line 9 and line 12: warning! You should change semicolon symbol (;) in predicate solve/3 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.
Between line 13 and line 18: warning! You should change semicolon symbol (;) in predicate solve/3 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.
Between line 20 and line 22: warning! You should change semicolon symbol (;) in predicate final/0 in to comma symbol (,). If not, your result will be wrong in the case of the value D<0.

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1st choice? Please add any explanation for your choice:

(4b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

5) There is **an implementation error of 'Missing parentheses'** in the current submission (2nd submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(5a)** and **(5b)**.

(5a) Which of these feedback messages do you prefer?
Please indicate your 1st preference with a "1" in the box, 2nd preference with a "2" in the box, 3rd preference with a "3" in the box, and so on.

☐ Between line 13 and line 18: warning! implementation error

☐ There is a problem here - implementation error
Between line 13 and line 18: warning! in predicate solve/3

☐ There is a problem here - implementation error
Between line 13 and line 18: warning! You are missing parenthesis ( ) to achieve the right solution in predicate solve/3 If you do not put the right parenthesis, it can cause wrong solution that program will calculate from the priority of operator (do * and / before + and -)

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1st choice? Please add any explanation for your choice:

(5b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

6) There are **5 style errors of 'Missing indentation'** in the current submission (2nd submission) according to the brief error messages provided by the system (in Figure G.3). Please answer the question **(6a)** and **(6b)**.

(6a) Which of these feedback messages do you prefer?
Please indicate your 1st preference with a "1" in the box, 2nd preference with a "2" in the box, 3rd preference with a "3" in the box, and so on.

☐ Between line 1 and line 3: warning! style error

☐ There is a problem here - style error
Between line 1 and line 3: warning! in predicate run/3
There are 5 errors like this.

☐ There is a problem here - style error
Between line 1 and line 3: warning! You are missing indentation of the body of in predicate run/3
There are 5 errors like this.

☐ There is a problem here - style error
Between line 1 and line 3: warning! You are missing indentation of the body of in predicate run/3
Between line 5 and line 8: warning! You are missing indentation of the body of in predicate solve/3
Between line 9 and line 12: warning! You are missing indentation of the body of in predicate solve/3
Between line 13 and line 18: warning! You are missing indentation of the body of in predicate solve/3
Between line 20 and line 22: warning! You are missing indentation of the body of in predicate final/0

☐ Say nothing (do not put any feedback message in the feedback report)

Why do you like your 1st choice? Please add any explanation for your choice:

(6b) Use Figure G.5 to decide what additional error messages to give (if any).
Answer: ____

Please add any explanation for your selection:

7) Assume that the feedback messages from question 1-6 are error messages, which structure do you prefer to use in the student's feedback report?

A. Error messages
Well done! You worked hard on this

B. This might be made better if you think about following suggestion
Error messages

C. This might be made better if you think about following suggestion
Error messages
Well done! You worked hard on this

D. This might be made better if you think about following suggestion
Error message #1
It would be helpful. Let's see the following suggestion
Error message #2
This could help you if you think about the following suggestion
Error message #3
Well done! You worked hard on this

E. Error messages

F. Well done! You worked hard on this

Please add any explanation for your selection:

G.2 Questionnaire sheet (for experimental group)

Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship - (McFeSPA) Questionnaire

This questionnaire is designed to gather feedback about your experience in using McFeSPA

Please tick the appropriate box for each item and write down any relevant comments that you may have.

1. Initially, I needed an expert to help me use McFeSPA.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

2. The interface for creating a feedback report is easy to comprehend.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

3. The text explanation helped me understand how each function worked.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

5. It was easy to manage the feedback message by edit/add/delete.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

6. It was easy to organise the feedback report.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

7. It was easy to edit the feedback report.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

8. It was easy to generate the feedback report.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Comments:_____

9. I found the basic idea of McFeSPA helpful.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

   Reason:_____

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
    ☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

    Comments:_____

11. I found the help messages easy to understand.

☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

12. I found help messages useful.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

13. I would prefer more details with the use of the help messages.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

14. The help messages helped me improve my skill for giving feedback.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

15. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

16. The error messages were helpful.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

17. The error messages clearly told me how to fix problems
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

18. McFeSPA provided useful tools for me to learn to give feedback.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

19. I enjoyed learning with McFeSPA.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

20. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

21. I would like to use McFeSPA in giving feedback to my students.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments:_____

22. McFeSPA helped me to finish my work.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

23. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

24. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

25. I became productive by using McFeSPA.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

26. McFeSPA can also be frustrating.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

27. Overall, it was easy to learn how to use McFeSPA
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

28. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

G.3 Structured interview sheet

Structured interview sheet

Background information questions

1) What is your previous experience of giving feedback like?

☐ some training (How often? Please specify below)

☐ none ☐ some ☐ quite a lot ☐ extensive

☐ some practical experience of giving feedback (How often? Please specify below)

☐ none ☐ some ☐ quite a lot ☐ extensive

Any further comments:_____

2) How much of the time (as a percentage) did you need to learn about the system itself and its functions?

☐ % (0% = no time spent on training 100% = all time spent on training)

Any further comments:_____

Pedagogical questions: (derived from the Hypotheses)

3) In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Opinion_____

4) In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback?

Opinion_____

5) Do you agree that McFeSPA can help you to give quality feedback to your students?

Opinion_____

6) In your view, what did you learn by using McFeSPA?

Opinion_____

7) In your views, is it possible for you to learn McFeSPA without any assistance from the system?

Opinion_____

8) In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?

Any further comments:_____

9) What do you like in particular about McFeSPA?

Reason:_____

10) What do you dislike in particular about McFeSPA?

Reason:_____

11) Do you have any suggestions for improving McFeSPA?

Suggestion:_____

Specific Questions

12) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?

Any further comments:_____

13) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?

Any further comments:_____

14) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?

Any further comments:_____

15) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Any further comments:_____

16) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Any further comments:_____

17) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?

Any further comments:_____

18) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?

Any further comments:_____

19) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?

Any further comments:_____

20) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?

Any further comments:_____

21) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?

Any further comments:_____

22) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?

Any further comments:_____

23) On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?

Any further comments:_____

Appendix H –Triangulation data from Evaluation of McFeSPA's Learning Environment

This appendix we present some triangulation data (not include pre-post test data, and log-file) of all participants i.e. questionnaire, and interview data.

H.1 Questionnaire data

Metacognitive Feedback Scaffolding System for Pedagogical Apprenticeship - (McFeSPA) Questionnaire

This questionnaire is designed to gather feedback about your experience in using McFeSPA
Please tick the appropriate box for each item and write down any relevant comments that you may have.

H.1.1 Participant 1's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: **The sequences on actions were not clear and didn't understand the scaffolding.**
2. The interface for creating a feedback report is easy to comprehend.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: **Sometimes the text scrolls off the screen and in the feedback report screen the text boxes are too small.**
3. The text explanation helped me understand how each function worked.
   ☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
5. It was easy to manage the feedback message by edit/add/delete.
   ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
   Comments: **I didn't use this. I tended to just edit the feedback using the keyboard.**
6. It was easy to organise the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: **The final format of the report wasn't how I would have liked it so it might have been nicer if I could had edit the form.**
7. It was easy to edit the feedback report.
   ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
   Comments: **a bit tiddly**
8. It was easy to generate the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: **but I wasn't sure when it was actually saved**
9. I found the basic idea of McFeSPA helpful.
   ☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
   Reason: **It is a good tool for speeding the marking task and the scaffolding is useful. However, it is a bit odd being given freedom to construct feedback anyway you want but then being told this is wrong if there is only one way to do it you would expect the system to automatically generate the feedback.**

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **It's always fun to have a challenge would be better if coloured a giver as percentage.**

11. I found the help messages easy to understand.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Unclear what they referred to or how I was to act on them.**

12. I found help messages useful.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **It takes a while to realise what they are referring to my skill as a marker.**

13. I would prefer more details with the use of the help messages.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **An initial tutorial would be good.**

14. The help messages helped me improve my skill for giving feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **I wouldn't have known what was expected about them.**

15. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

16. The error messages were helpful.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **It was uncomfortable being told that I had made a mistake.**

17. The error messages clearly told me how to fix problems
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: **not clearly. Didn't always know what they expected me to do.**

18. McFeSPA provided useful tools for me to learn to give feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

19. I enjoyed learning with McFeSPA.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

20. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **I feel that I have exhausted the system in terms of learning teaching skills**

21. I would like to use McFeSPA in giving feedback to my students.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

22. McFeSPA helped me to finish my work.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

23. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

24. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

25. I became productive by using McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

26. McFeSPA can also be frustrating.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree

Reason: **especially when it tells you that you have made a mistake but not what it is.**

27. Overall, it was easy to learn how to use McFeSPA
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

28. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

H.1.2 Participant 2's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

2. The interface for creating a feedback report is easy to comprehend.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

3. The text explanation helped me understand how each function worked.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

5. It was easy to manage the feedback message by edit/add/delete.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

6. It was easy to organise the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

7. It was easy to edit the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

8. It was easy to generate the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Sometimes students name is incorrect, have to close and try again.**

9. I found the basic idea of McFeSPA helpful.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

11. I found the help messages easy to understand.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **They are quite standard, easy to learn after several attemps.**

12. I found help messages useful.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **One message could be improved 'Are you ..sure? Do you need help?' are contradicting.**

13. I would prefer more details with the use of the help messages.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments:_____

14. The help messages helped me improve my skill for giving feedback.

☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **It's quite a uniform way**

15. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

16. The error messages were helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: **It points out the error, also indicates if student has learnt from previous submission.**

17. The error messages clearly told me how to fix problems
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

18. McFeSPA provided useful tools for me to learn to give feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Especially in an automated way**

19. I enjoyed learning with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

20. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

21. I would like to use McFeSPA in giving feedback to my students.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Though more insight should be gained into looking at more severe and subtle problems in students.**

22. McFeSPA helped me to finish my work.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

23. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

24. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: **Up to a certain extent, same as 21**

25. I became productive by using McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

26. McFeSPA can also be frustrating.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: _____

27. Overall, it was easy to learn how to use McFeSPA
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

28. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: **general and simple**

H.1.3 Participant 3's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

2. The interface for creating a feedback report is easy to comprehend.

☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

3. The text explanation helped me understand how each function worked.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Tool-tips would be helpful.**

4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

5. It was easy to manage the feedback message by edit/add/delete.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

6. It was easy to organise the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

7. It was easy to edit the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

8. It was easy to generate the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

9. I found the basic idea of McFeSPA helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

11. I found the help messages easy to understand.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Not standard help messages**

12. I found help messages useful.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

13. I would prefer more details with the use of the help messages.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

14. The help messages helped me improve my skill for giving feedback.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Never used them**

15. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

16. The error messages were helpful.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: _____

17. The error messages clearly told me how to fix problems
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason:_____

18. McFeSPA provided useful tools for me to learn to give feedback.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: _____

19. I enjoyed learning with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

20. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Need more theory before this**

21. I would like to use McFeSPA in giving feedback to my students.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

22. McFeSPA helped me to finish my work.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

23. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

24. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

25. I became productive by using McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

26. McFeSPA can also be frustrating.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: **Maybe for complex programs**

27. Overall, it was easy to learn how to use McFeSPA
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

28. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

H.1.4 Participant 4's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments:_____

2. The interface for creating a feedback report is easy to comprehend.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

3. The text explanation helped me understand how each function worked.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

5. It was easy to manage the feedback message by edit/add/delete.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

6. It was easy to organise the feedback report.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

7. It was easy to edit the feedback report.

☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

8. It was easy to generate the feedback report.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

9. I found the basic idea of McFeSPA helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

11. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

12. The error messages were helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

13. The error messages clearly told me how to fix problems
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

14. McFeSPA provided useful tools for me to learn to give feedback.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

15. I enjoyed learning with McFeSPA.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments:_____

16. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

17. I would like to use McFeSPA in giving feedback to my students.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

18. McFeSPA helped me to finish my work.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

19. McFeSPA helped me to finish my work quickly.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

20. McFeSPA helped me to finish my work effectively.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

21. I became productive by using McFeSPA.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

22. McFeSPA can also be frustrating.
☐ Strongly agree ☐ Agree ☐ Disagree ☑ Strongly disagree
Reason: _____

23. Overall, it was easy to learn how to use McFeSPA
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason:_____

24. Overall, I am satisfied with McFeSPA.

☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

H.1.5 Participant 5's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
   ☐ Strongly agree ☐ Agree ☐ Disagree ☑ Strongly disagree
   Comments: _____
2. The interface for creating a feedback report is easy to comprehend.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
3. The text explanation helped me understand how each function worked.
   ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
   Comments: _____
4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
5. It was easy to manage the feedback message by edit/add/delete.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
6. It was easy to organise the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
7. It was easy to edit the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
8. It was easy to generate the feedback report.
   ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
   Comments: _____
9. I found the basic idea of McFeSPA helpful.
   ☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
   Reason: _____
10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
    ☐ Strongly agree ☐ Agree ☐ Disagree ☑ Strongly disagree
    Comments: _____
11. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
    ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
    Comments: _____
12. The error messages were helpful.
    ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
    Reason: **There was only one error message and it was obscure**.
13. The error messages clearly told me how to fix problems
    ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
    Reason: **error messages were not about problems I had.**
14. McFeSPA provided useful tools for me to learn to give feedback.
    ☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
    Comments: **I'm done a lot of marking, but it was still interesting**
15. I enjoyed learning with McFeSPA.
    ☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
    Comments: _____

16. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: _____

17. I would like to use McFeSPA in giving feedback to my students.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **I don't teach Prolog anymore& I think the hard part is finding the errors, not writing the report.**

18. McFeSPA helped me to finish my work.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **I didn't use it for "my" work & I think it should be more automated.**

19. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **It would if it was more automated**

20. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **It the history feature was automated, then it would.**

21. I became productive by using McFeSPA.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **As above.**

22. McFeSPA can also be frustrating.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: **Because many parts feel like they should be automated.**

23. Overall, it was easy to learn how to use McFeSPA
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

24. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **need to be more automated.**

H.1.6 Participant 6's questionnaire

1. Initially, I needed an expert to help me use McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **It wasn't strictly necessary, but it was helpful.**

2. The interface for creating a feedback report is easy to comprehend.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

3. The text explanation helped me understand how each function worked.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

4. It was easy to select the choice of feedback message, weakness message, and 'positive feedback' before generating the feedback report.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Easy, but needed more thought than I'd initially realised. I was assuming, I'd use the same format for all students, then realised that different would be better.**

5. It was easy to manage the feedback message by edit/add/delete.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

6. It was easy to organise the feedback report.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

7. It was easy to edit the feedback report.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree

Comments: _____

8. It was easy to generate the feedback report.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

9. I found the basic idea of McFeSPA helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: **It would greatly help when marking assignment. Even if it isn't used (e.g. assignment in another topic), it's useful to evaluate what feedback I'd give.**

10. I found the representation of my skill at giving feedback (skill meter), after using McFeSPA, made me realise I needed to improve my skills at giving feedback.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Didn't really understand this point.**

11. I found the representation of my skill at giving feedback (skill meter) helped me think more about my skills.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Comments: **Using the system helped me to think about skills, but the representation on this page didn't mean much to me.**

12. The error messages were helpful.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

13. The error messages clearly told me how to fix problems
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

14. McFeSPA provided useful tools for me to learn to give feedback.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: _____

15. I enjoyed learning with McFeSPA.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Very nice system and session. Good to have paper exercised too.**

16. I can learn more about giving quality feedback by using McFeSPA again.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Comments: **Although would have its limit.**

17. I would like to use McFeSPA in giving feedback to my students.
☑ Strongly agree ☐ Agree ☐ Disagree ☐ Strongly disagree
Comments: **If I were marking Prolog, this could greatly help. I imagine that it would miss quite a few errors though.**

18. McFeSPA helped me to finish my work.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **I've only used in experiment, so it hasn't help as yet.**

19. McFeSPA helped me to finish my work quickly.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: **If I was using it to mark, it would be useful.**

20. McFeSPA helped me to finish my work effectively.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

21. I became productive by using McFeSPA.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: _____

22. McFeSPA can also be frustrating.
☐ Strongly agree ☐ Agree ☑ Disagree ☐ Strongly disagree
Reason: _____

23. Overall, it was easy to learn how to use McFeSPA
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree

Reason:_____

24. Overall, I am satisfied with McFeSPA.
☐ Strongly agree ☑ Agree ☐ Disagree ☐ Strongly disagree
Reason: _____

H.2 Interview data

H.2.1 Participant 1's interview result

The results of Participant 1's interview are about 20 minutes.
(Experimenter: Plain text, Participant 1: *Italic*)
1)
Experimenter: Do you have any previous experience of giving feedback like some training?
Participant 1: *Yeah, I think whenever I am a teaching assistant. The school of informatics has the training skill course how to teach students in the tutorial group, how to provide feedback.*
Experimenter: So you have some training, or quite a lot, or extensive?
Participant 1: *Not extensive. Let's say 'some', 'not quite a lot'*
Experimenter: So you have some experience of giving feedback.
Participant 1: *I got quite a lot.*
Experimenter: Is it extensive?
Participant 1: *No, just quite a lot.*
2)
Experimenter: How much of the time (as a percentage) did you need to learn about the system itself and its functions? When 0% is no time spent on training and 100% is all time spent on training). If you need the system to help you to mark programming assignment to give feedback, do you think how much do you need it? How much of time?
Participant 1: *to learn how to use the system?*
Experimenter: Yeah, when 0% means no time spent on training.
Participant 1: *Am Hmm.*
Experimenter: and 0% means all time spent on training. You may need it every time spent on training.
Participant 1: *OK, well, I say just need about 10 or 15 minutes to start of it*
Experimenter: How is it, in percentage?
Participant 1: *about 5 or 10%*
Experimenter: OK, about 5 or 10%
3)
Experimenter: Now, it is the pedagogical questions. In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?
Participant 1: *Well, it's useful to see the citation about the effect of the different form of feedback and to explore the different skill that they try to teach. I think that is quite useful. Sorry, what's the question again?*
Experimenter: In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?
Participant 1: *Yeah, really just by having access to those to that information about skills.*
Experimenter: Do you mean glossary? Or the skill meter?

Participant 1: *Yeah, Am but really, it was just, I don't know I learned so much and did what it told me to do.*

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback?

Participant 1: *Am, I think it seem strange to start of with and being told you are doing wrong but then I congratulatory I can see the benefit of doing the way it told you to and you just do it. So I think it explores you to the different approaches and to get benefit about that approaches.*

Comment: The participant 1 say 'You' mean 'the participant 1'

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 1: *Yeah, I think so, probably more than writing on a piece of paper. I can structure and give some more details*

6)

Experimenter: In your view, what did you learn by using McFeSPA?

Participant 1: *I learn the way that the errors are enclosed by positive feedback motivating feedback is very important, "sandwiched".*

7)

Experimenter: In your views, is it possible for you to learn McFeSPA without any assistance from the system?

Participant 1: *I didn't see is there any help file at all. I expect that to be in the system, the help organise how I can click on. It might be missing right now because I don't know which button to press first. I don't know which act to do.*

Experimenter: Oh, right. You expect to have help button to select by your need.

Participant 1: *Yeah, because I don't know which button to press first. I don't know which act to do*

Experimenter: So, if the help button is available then you are possible to learn.

Participant 1: *I think so, yeah. You need to be able to help. This is the McFeSPA. In order to use this, you have to do this than that step. Once you were told that you be able find by yourself. I can get an orientation to the system.*

8)

Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?

Participant 1: *Umm, I guess so. I think Prolog is quite simple in the range of errors. You make over language more complicated so task' able is the main analysis of them and be able to provide the feedback might be complicated but in theory it can.*

9)

Experimenter: What do you like in particular about McFcSPA?

Participant 1: *Am, I like the automated analysis tool that automated find the error and give the location point. That is very good.*

10)

Experimenter: What do you dislike in particular about McFcSPA?

Participant 1: *I think the range of feedback is limited. It will be nice if you have a broader range of sample sentences.*

Comment: If we could have plug-in of the sample sentences, we could do in the current version. There is a research (in Sussux University?) that provides a number of samples of feedback on students' programming assignment.

11)

Experimenter: Do you have any suggestions for improving McFeSPA?

Participant 1: *Well, yeah, I think I have made some suggestion during using the system.*

Experimenter: It might be easier if you use it and give the comment during the use.

Participant 1: *OK. What would be nice when you click on this? I would see the text on here and when you click on the option so you can see automatically. What you want it is going to be and you can see what you don't want it to be. And also when you add this in this bit you can choose to do this. It caps the same window so you can see what they look like its combination. OK. That'll be quite nice.*

Comment: The participant giving comment on the 'Choices for more errors' interface.

Participant 1: *And that when I create the report this is too much here. If I lost in here, if I can find next. The window is too small so maybe if you happy to collapse it. You click on the design error it will show you, give you more space on the screen and then one final thing when you generate the final report. It will be nice if this look like a text screen so if I can go here I can change it as the performance I want. So it's quite clear, multiple line here, add the extra text, so the last final check, is it has many operation then I will save or send and just check on, go back to. Go back to this screen and stop editing here. Right now you got all this different bit and it's hard to fit it together to make the report. But when now you put them together, it will be nice jus say OK move this bit around to add it a text before you can save it.*

12)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?

Participant 1: 5

13)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?

Participant 1: 4 (handle the text)

14)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?

Participant 1: 5

15)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Participant 1: 3, actually this' quite hard because you always have to click the students' profile. It should always automatically put that on. That will be better. So I probably say 3.

Comment: In order to check the user's giving feedback loop, we need to provide a button to check whether the user taking into the history of student's errors.

16)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?
Participant 1: *I say 4. It will be nice if you have more option here*
Comment: Provide more option beyond 'Add', 'Delete', and 'Update' button.
17)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?
Participant 1: *It's easy to use. I give 5 but the pictures are different to see. If you could make the image, you could make the text more clear.*
18)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?
Participant 1: *I could say 3 because it's too much detail going on.*
19)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?
Participant 1: *Is it save here?*
Experimenter: Yes, it is.
Participant 1: *I say 3, but I'm not sure can it save because the instruction is not clear. And I need some kind of instruction say you are free to edit.*
20)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?
Participant 1: *Oh! That was good. I like that. Am I say 5. I didn't actually realise these lines explain each other of these bars. So it might be better if you put this line under each bar*
Experimenter: Right.
Participant 1: *Because I didn't realise its relationship.*
21)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?
Participant 1: *4*
22)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?
Participant 1: *6*
23)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?
Participant 1: *7*

Experimenter: Thank you very much for your help to take part in this study.

H.2.2 Participant 2's interview result

The results of Participant 2's interview are about 36 minutes.
(Experimenter: Plain text, Participant 2: *Italic*)

1)
Experimenter: What is your previous experience of giving feedback like? Some training (How often?)
Participant 2: *I have ever trained before. Am. Not formally but like for giving feedback during marking when I was teaching in Malaysia. I had to do lecture myself. I had to set a question myself and I marked myself.*
Experimenter: So you never have any train before but you got experience by yourself.
Participant 2: *Yeah, but I'm also marking here.*
Experimenter: So you are teaching assistant here.
Participant 2: *Yeah, I had a little but so I was marking for 1st year in JAVA assignment. So they give me a marking guide. They can give use a special format.*
Experimenter: So you haven't been trained before
Participant 2: *Yeah*
Experimenter: But you have experience by yourself. Do you think your experience just some, or quite a lot, or extensive
Participant 2: *Am, not extensive. I don't know what you mean by "quite a lot"*
Experimenter& Participant 2: laugh
Participant 2: *I still be a student so I have some. Maybe you can say maybe three years or four years but I think for two years and I also here helping but part time. You know, like I'm not marking all the time but I help a little bit so maybe three years. I don't know what do you think about?*
Experimenter: Now, you are the 1st year PhD.
Participant 2: *but when I am MSc, I was also helping but was rather marking with a lab. I just was helping with the lab everything but not marking but this year I was marking.*
Experimenter: All right.
Participant 2: *And 2 years in Malaysia. Like I had some exam so I had marked.*
Experimenter: That's quite a lot.
Participant 2: *But I think not quite a lot. Maybe compare to some people here. You know, it's not. You know, they were teaching but I did not here. Maybe some, I expect.*
Experimenter: OK.
2)
Experimenter: OK. How much of the time (as a percentage) did you need to learn about the system itself and its functions? I mean assume that when you use the system to mark Prolog programming assignment. Do you think do you need any train or any help from the system, any demonstration, for example?
Participant 2: *Yeah, of course.*
Experimenter: If I offer 0% means no time spent on training and 100% means all time spent on training
Participant 2: *OK.*
Experimenter: and 100% means all time spent on training. I need to learn to use or to see the demonstration.
Participant 2: *Oh, no. I think at first when you show me the demo at first. You know you have a video running. Yeah, I couldn't really. I mean I was watching it but I didn't know when you show me personally how to do it. It looked ok when I was using it the first time I didn't know whether I could use. Probably, I realise it was quite easy so it was quite quick.*
Experimenter: OK. Do you think you need any training again (as in percentage)

Participant 2: *You mean now. Ah, I wouldn't say zero but maybe 5 or 10 percent. I think I am comfortable with the system but you based on what I did. I don't know how much more functionality it was. Maybe it has more advance. I think but just what I learnt.*

Experimenter: All right. Some functionality I have tested usability of the system but don't test some in this study. I have to specific on some functionality.

Participant 2: *I don't know what is zero. What is another one?*

Experimenter: 0% means no time spent on training and 100% means all time spent on training and 100% means all time spent on training. I need to learn to use or to see the demonstration.

Participant 2: *so just 5 or 10 %*

3)

Experimenter: In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 2: *I think it more systematic and more structure. Maybe because the problem is Am it make zero like when you have the problem to solve and that was the solution. Ah, you know don't see the bigger picture. OK here are the problems like you say. This is semi colon or unreachable goal. It's just organises the structure. It's better than I did. Stop me form... because sometime I look for the related problem like... the students didn't do this and I need really to do something else... and that make my marking more difficult than I realise because I marked manually a lot. And always I try to type why student do this and more psychological but sometime I think it's easy and quick and maybe that's all the student's need for now. <u>That's what I realise. I realise that maybe I have been thinking that I give you more structure and organise the way of marking.</u>*

Experimenter: OK, I am going repeat the question and show you an example. How does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 2: *Yeah.*

Experimenter: If you answer wrong, the system will offer help but as I observed you. You didn't answer wrong. You didn't get any help from this because you always answer right. Do you remember?

Participant 2: *Yeah. Yeah. Yeah.*

Experimenter: For example. This one you never answer wrong. I think you are an experience teacher.

Participant 2: *Laugh, yeah. So in term of knowledge, maybe not but it is a good way of presenting of the feedback because it structure and organise but I don't know about.*

Experimenter: OK. You never use the system before when you use it do you gain any knowledge in giving feedback by using the system.

Participant 2: *Ah, I think it's better because it taught me how to be simple and straight forward about my feedback.*

Experimenter: All right.

Participant 2: *And not analyse too much sometime. It's better to just say ok. Yeah, it's a problem and I think for student's learning. For example, learning to use program, maybe this is other thing that they need to know rather than I don't know more complicate than I think. Once I can get rid of the simple problem when they solved more difficult problem. It not so*

such a big issue any more. Yeah, because we can be quite implicit when we are trying to give feedback we don't realise. We just have to count them, OK missing cut. You know, just tell them that you are very implicit. We are trying to say. Oh, this is not a good kind of feedback maybe doesn't highlight. Maybe all they need to know is no cut. In that sense, the knowledge I learned just to be simple.

Experimenter: All right. Looks like it give more explanation and help you to give feedback

Participant 2: *because you had like design/implement/style. You know you had that but I wouldn't have that. Yeah, it's systematic. I think.*

Experimenter: All right.

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback?

Participant 2: *Am. I think generally. I'm very positive anyway. Am. when I marked Am. but not like I don't I think the student had done giving a very good answer that I never seen before something special than I will say well done or I will give a very excellence but I wouldn't do it normally. I wouldn't just say excellence or well done. If they have done a normal mark so that sense I think it more positive than I would be normally. Oh, sorry, how is the question?*

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback?

Participant 2: *Yeah, I think that is may most of what I told you the first part. I think what I like about its automatic. You know I like make the real of marking because it is so much quicker. And I think it is very good.*

Experimenter: Do you think the skill meter help you to rethink you in giving feedback when you have a look this?

Participant 2: *Possibly, yeah, yeah, some of them but I already got some help from the system to help me like... do that feedback. So I already have a good meter skill because the system helping me.*

Experimenter: In term of the help message. Does it help you to reflect or rethink about this rather than this?

Participant 2: *Yeah, ok, the messages themselves Ah, yes, they have. I think it's good because it gave the line number. It gave problems. Yeah, it could make me think more an automatic way then because it automatically gave me three errors like this. You know I normally wouldn't say I would just say yeah, there were more errors that I would point and say same as. I would just say this two errors same as... You know but this one gave a report so umm I think it would help me more clear and precise about my feedback like, yeah, and break down every error you know.*

Experimenter: All right.

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 2: *Yeah, I think quality up to a curtain extent because quality can be a lot of things I think. A quality in term of time definitely because marking manually take a long time and this one is very quick because it does the analyse I think it's great. You just have a code and you can analyse*

and it comes with that. I think for me that quality because I don't do it myself then quality because of time. I think I save a lot of times. Umm in term of the messenger, I think, yes, for simple and quick messages like syntax error so you say missing cut and all that. Yes, definitely but beyond that. Let's say maybe the students actually has some other problems that deep or more subtle that you can't just get from just saying or these are five missing parenthesis but when you use the system You don't realise you have to look at the solution and look for more the bigger problem but that you have to do your self. I think so that I couldn't do with the system.

Experimenter: Am Hmm

Participant 2: *Not say for basic but for quick and that I said. You know, it's very good I think for like tutorial or like exercise for 1st year students when you want to answer you want to say just that's the basic skill but beyond that if they are going to like intermediate or advance level, I think, or you want to see a certain pattern. Maybe, for example, the parenthesis, you know the one that they are missing the parenthesis. That is like basic mathematic. It's not to do with the programming so they are mathematic skill. They are like mathematic skill. So as a tutor, let's say, do something else. So, you can relate and say oh! This showed that you know you really need to learn your algebra. You really need to ....which is something you learn in school but I think the system doesn't necessary help to do those kind of analysis.*

Experimenter: but in the kind of implementation issue. The system will not process if lack of another parenthesis. This type of error is classified in the system as implementation issue.

Participant 2: *Yeah, I think missing the parenthesis is just a syntax error or sorry maybe just actually allows do that but what I am saying like it's a subtle error where is actually it not answering the question but syntactically collect. You know what I mean.*

Experimenter: Like a logical error.

Participant 2: *Yeah, yeah, yeah, that's right. So I think the system doesn't help to solve this kind of error but then it's up to you. So it has to use your thinking and also use the system to get a really good feedback and excellence feedback. Yeah I think the system is good because I never use anything like this. I always just use hand marking. So I don't know this is my first system. I'm using which is good so it's fantastic.*

Comment: Participant 2 thinks quality feedback is that McFeSPA give automatic analyse code quickly. PT 2 think in term of saving time (is the quality feedback)

6)

Experimenter: In your view, what did you learn by using McFeSPA? I think you already answered.

Participant 2: *laugh, Yeah like the automated way of this thing. I think they are fantastic and I think they should be used because I don't know many people you will speak to but for me it's very quick umm for example. It depends how much you want to when you marking umm some time when they marking assignment they contribute only 1 or 2 percent to*

the student over all mark so you don't have to worry if you are giving them very personalise like sometime because I have been doing manual. So what happen when I do marking the student's thing I can see the way they program. The way they actually are and whether they are weakness or strength are. In a very like umm maybe in mathematic is not able to do something. You know like student's characteristic but think all these things are not necessary. Let's say for assignment that contributes to 1 or 2 percent. It takes too much time because you are thinking and you're marking. You are not just looking at, OK, here it's right, here it's wrong. You know so I think this is excellence for assignment and you know you just get them quickly rather than a very big assessment where it's 50 percent you know something is very big. You're definitely to look and analyse their code much more in that and look for more subtle error. Some more like pattern and the way they program. I think this system is excellence for marking but I am sure this is very difficult. It just analyse the error OK and just say ok here are five errors. I think that's good.

Experimenter: It summarize of the errors.

Participant 2: *Yeah, yeah. It's very good. And also as a human, we can make a mistake, maybe we don't realise umm sometime we don't realise we are marking, we forget. Sometime we look at the code. It's correct but actually it's wrong so I think using the automated way accurate at least catch the errors. You know sometime we don't see it.*

7)

Experimenter: All right, in your views, is it possible for you to learn McFeSPA without any assistance from the system?

Participant 2: *Yeah, it's ok. That's because I have used this with.*

Experimenter: Scaffold-on mode.

Participant 2: *Yeah, it's possible*

Experimenter: You are familiar with.

Participant 2: *Yeah. It's ok.*

8)

Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?

Participant 2: *Yeah, I think. Well, I don't know, depend on how the system analyse this.*

Experimenter: The knowledge to apply to other programming language

Participant 2: *Yeah, Yeah, Yeah, generally. It's just like I told you. It's difference. In the structure, automation, organisation which is the same in all programming language.*

9)

Experimenter: What do you like in particular about McFcSPA?

Participant 2: *Laugh, it's very quick and also it's easy to use in general, very few bug so far. I think it's good. I think it's very simple and umm as well every time it pops up, come up, almost always like the same one are two kinds of messages. You know "Are you sure? ..." so it was very easy to use. You don't read different message all the times and I think the uniformity it's easy to use.*

Experimenter: OK.

10)

Experimenter: What do you dislike in particular about McFcSPA?

Participant 2: *Am, I don't know. I didn't read the ... when I say generate whenever the report. I didn't actually read that thing because I belief that it generates the errors properly. Ok, I have to think about this more.*

Experimenter: You can use the system again, and see what do you dislike it?

Participant 2: *Yeah. Actually I didn't realise. Yeah, now I realise. I didn't like when you go create report and then ask you to choose the template of what kind of template you want it to be because I suppose it to select I see the point. It's ok but I don't know why it not the right answer even though I see the rational. OK how you present it. Sometime you want me to say something else and then error and then say something else. I don't know I didn't like it the first time I say it because it looks complicated you know at this style especially when the scaffolding was on. That the system tried to tell me that I didn't choosing the right template but I didn't know what the right template? Suppose to be so that was a little bit Am, yeah.*

11)

Experimenter: Do you have any suggestions for improving McFeSPA?

Participant 2: *Yeah. I think for now it's a good assessment because it highlight the error but I think umm this probably just got a problem in that it can try e.g. it said ok, number of error this time five, and then previous time two but it's solving the same problem. Let's say it was a different program that the student's wrote but let's say it's the same student they did one exercise and this is the 2nd exercise they are doing or is it the same exercise?*

Experimenter: Yeah.

Participant 2: *Oh, ok, so then it's not so bad. Let's say if they are the 1st exercise then the 2nd exercise now, which is different problem. What am I saying is, those numbers don't reflect anything. The numbers of mistakes maybe doing something like percentage. You know something more proportional rather than number of mistakes because like let's see you make five mistakes last time and now you make three. I didn't like that message that "Well done" I wouldn't say "Well done". I don't know I think they are doing the same mistake that's the whole point. It doesn't reflect like sometime they didn't learn anything. I don't think analyse the previous and current error in the more realistic way. Maybe not just the number of errors, maybe they are confusing but this might be good when you use the system with the 1st simple assignment to the students. I think this is quite very useful. It's very quick. I think in term of time.*

Experimenter: Right.

12)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?

Participant 2: *Am , 6.*

13)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?

Participant 2: *Oh, yeah, very easy. 6 yeah.*

14)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?

Participant 2: *Yeah 6 as well.*

15)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Participant 2: *Yeah 6 as well.*

16)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Participant 2: *I didn't use it Am. I would say 4 but I don't think it is difficult. Actually even I didn't use it yet.*

17)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?

Participant 2: *Laugh, this interface is very easy to use, 7.*

18)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?

Participant 2: *This one maybe 5. Just that, it has a lot of thing to do. Actually it's not bad at all.*

19)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?

Participant 2: *Yeah, that was easy, 6.*

20)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?

Participant 2: *I think the explanation should be there (under each bar). I could say 6.*

21)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the second script?

Participant 2: *5 maybe.*

22)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?

Participant 2: *6, after that.*

23)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the second script?

Participant 2: *Yeah, I think the 3rd one also 6 yeah.*

Experimenter: Thank you very much for your attention in this study.

H.2.3 Participant 3's interview result

The results of Participant 3's interview are about 21 minutes.

(Experimenter: Plain text, Participant 3: *Italic*)
1)
Experimenter: Do you have any previous experience of giving feedback like some training?
Participant 3: *Some training*
Experimenter: Right, you have some training before you are teaching assistant.
Participant 3: *Yeah, half day or something.*
Experimenter: Do they give you any train? What did they teach you?
Participant 3: *Basically, yeah, they told us the basic principle, something in feedback. How to be supportive, how to critical? Something likes that. That's quite pretty enough, general knowledge.*
Experimenter: Right, do you have practical experience of giving feedback. I think you have ever marked before.
Participant 3: *Yeah*
Experimenter: Do you think how often?
Participant 3: *I marked expert system lab 20 sheets a year.*
Experimenter: 20 sheets a year.
Participant 3: *20 sheets, 20 students*
Experimenter: Only once a year.
Participant 3: *Yeah, only once, only once a year.*
Experimenter: Not repeat? 1st lab, 2nd lab
Participant 3: *No, they gave me all the sheets at one time. This is a multiple test, small C program and be accepted to got to the next step.*
Experimenter: So you marked only once a year, and you think you have some experience.
Participant 3: *Yeah*
2)
Experimenter: How much of the time (as a percentage) did you need to learn about the system itself and its functions? When 0% is no time spent on training and100% is all time spent on training). If you need the system to help you to mark programming assignment to give feedback, do you think how much do you need it? How much of time?
Participant 3: *For the system?*
Experimenter: Yeah
Participant 3: *10%*
Experimenter: Do you have any comments about this?
Participant 3: *No, please straight forward*
Experimenter: Right, next.
3)
Experimenter: Now, it is the pedagogical questions. In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?
Participant 3: *Basically, it said in the meter a word some asking question or something. Usually they do not allow write comments and ask questions in my marking. It just finds the mistake than I mark so they not allow me to write the comments. The suppose it will be bad to marking comment or something*
Experimenter: I mean, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 3: *Well, this one said contradicting with before time talk. Well, this bit so speed which confuse.*

Experimenter: Oh, you confuse.

Participant 3: *And I learn a whole as a better way.*

Experimenter: Because you never provide feedback like this.

Participant 3: *Maybe bit difference. What, something you teaching, maybe for English or something you made, be more critical or something but ...just give this a mistake, this is a mistake.*

Experimenter: Right, in your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback? We don't point out at the domain knowledge of programming but we point out at the domain knowledge of giving feedback.

Participant 3: *not much.*

Experimenter: because?

Participant 3: *because I am used to giving feedback in the normal way, in the standard way so it's hard to change in an hour. It happens like this for two years. I am quite used to giving feedback in the same way.*

Experimenter: Right, you are familiar with.

Participant 3: *Yeah, I'm familiar.*

Experimenter: with pointing out this wrong and this right, not familiar with extending messages, umm.

Participant 3: *Yeah.*

Experimenter: for two years?

Participant 3: *Yeah, two years.*

Experimenter: And ok, next question.

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback? You can use the system again if you'd to use.

Participant 3: *Am, ok what is the question?*

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback?

Participant 3: *I think it's more structure.*

Experimenter: Am Hm

Participant 3: *You see the mistake than you see what feedback you want to give. It's like an assembly change, when I was doing it manually it tended to be quite random, but this system provides more structure. You can't skip step but manually, you see the sheet and you just cite the single comments of it but here you have to go the whole loop.*

Comment: PT 3 doesn't know that the system doesn't control/force the use to sequence giving error messages. The user can select any error type (don't be necessary to sequence)

Experimenter: OK, next.

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 3: *Yeah, I think so because it allows you to add more errors something else. It can help, yeah.*

Experimenter: OK, next.

6)

Experimenter: In your view, what did you learn by using McFeSPA?

Participant 3: *What I learn?*
Experimenter: Am, right what did you learn?
Participant 3: *Oh, I think I learnt different way of giving feedback because normally you don't see the work of student. You just see them at one point but in McFeSPA maybe the previous mistakes of student can also be used in giving feedback. I never used that kind of thing.*
Experimenter: You never used that kind of thing, right.
Participant 3: *Usually they don't give me the matriculation number. They gave me very random. A set of sheets, we don't know who's the students. Whose sheets they are marking?*
Experimenter: Right, next.
7)
Experimenter: In your views, is it possible for you to learn McFeSPA without any assistance from the system?
Participant 3: *To learn?*
Experimenter: Yeah.
Participant 3: *I think you need someone to help you to use it.*
Experimenter: so, it's possible, can't use without.
Participant 3: *I don't think so. You need some form of training to use it properly.*
Experimenter: Ok, anything else, Right.
8)
Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?
Participant 3: *I think I can be more structure that I'm marking. I can take thing one by one rather than know how.*
Experimenter: when you mark the other programming language.
Participant 3: *Yeah, yeah, yeah, you know this is the same thing. You have these kinds of sense. You have artificial skill.*
Experimenter: and you can apply this
Participant 3: *Yeah.*
Experimenter: and next
9)
Experimenter: What do you like in particular about McFeSPA?
Participant 3: ***I like that I think. You can extend the errors.***
Experimenter: Right, you like it because you can extend the errors
Participant 3: *Yeah, I can extend my own knowledge. I can get the message you can see.*
Experimenter: Right, next.
10)
Experimenter: What do you dislike in particular about McFeSPA? You can use it again.
Participant 3: *It's ok. It's quite pedagogic. It's ok for the first time but every time it keeps asking. "Do you want to close?" (laugh), that's a bit annoyed.*
Experimenter: Do you want to close?
Participant 3: *All the things "Do you really want to close?"*
Experimenter: Ok, anything else, okay, next.
11)
Experimenter: Do you have any suggestions for improving McFeSPA?
Participant 3: *No, not really.*
Experimenter: You can suggest in term of usability.

Participant 3: *That's ok. I'm ok with this.*
Experimenter: How about the suggestion, not to add the student's name to the feedback report during marking.
Participant 3: *No, it depends. If this for the small marking, you need to know the students to the personal feedback but in the exam, you don't want to know the student.*
Experimenter: So, in term of practising, you need to know. Okay.
Participant 3: *Well, kinds of thing, it depends. So you can argue with this.*
Experimenter: Right, do you have any suggestion to improve this?
Participant 3 shook head
Experimenter: OK, no, next.
12)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?
Participant 3: *7 is easy, 5*
Experimenter: Do you have any comment about this?
Participant 3 shook head
Experimenter: OK, next.
13)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?
Participant 3: *6*
Experimenter: Any comments?
Participant 3: *Please straight forward.*
Experimenter: and next.
14)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?
Participant 3: *6*
Experimenter: Any comments? OK.
15)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?
Participant 3: *6*
Experimenter: Any comments? OK.
16)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?
Participant 3: *4 because the database you presented is awesome. The people who don't know computer properly may confuse how you present record 1 or record 2.*
Experimenter: Any further suggestion about this? OK.
17)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?
Participant 3: *This should be 5 because it's pretty blur.*
Experimenter: Anything else? OK.
18)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?
Participant 3: *4 because you know, again what the image doing, the record presents.*
Comments: PT 3 gave the comments similar to the question number 16.
Experimenter: Ok, anything else about this interface?
Participant 3 shook head.
Experimenter: No, next.
19)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?
Participant 3: *I think 6.*
Experimenter: Do you have any suggestion? Ok, next.
20)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?
Participant 3: *I think 6.*
Experimenter: Do you have any comments about this interface? Right
21)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?
Participant 3: *4.*
22)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?
Participant 3: *a bit easy, 6.*
23)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?
Participant 3: *yes, 6.*
Experimenter: So, same, right. Thank you very much for your help to take part in this study.

H.2.4 Participant 4's interview result

The results of Participant 4's interview are about 12 minutes.
(Experimenter: Plain text, Participant 4: *Italic*)
1)
Experimenter: Do you have any previous experience of giving feedback like some training?
Participant 4: *No, but I do it by myself.*
Experimenter: In your perspective, do you think do you have experience in giving feedback some or quite a lot or extensive.
Participant 4: *Yes, some.*

2)
Experimenter: How much of the time (as a percentage) did you need to learn about the system itself and its functions? When 0% is no time spent on training

and100% is all time spent on training). If you need the system to help you to mark programming assignment to give feedback, do you think how much do you need it? How much of time?

Participant 4: 5-10 %

3)

Experimenter: Now, it is the pedagogical questions. In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 4: I have more example of feedback.

Experimenter: Right.

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback? You can use the system again if you'd to use.

Participant 4: Give me more idea that the feedback should have more one kind. However, I don't know exactly how to give feedback in each situation.

Comments: PT4 don't know how to give feedback in each situation because PT4 used the system without scaffolding.

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 4: Yes, I agree.

6)

Experimenter: In your view, what did you learn by using McFeSPA?

Participant 4: I learn to provide more kinds of feedback

7)

Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?

Participant 4: Yes, of course.

8)

Experimenter: What do you like in particular about McFeSPA?

Participant 4: It automatically generates report and can adaptable feedback message.

9)

Experimenter: What do you dislike in particular about McFeSPA? You can use it again.

Participant 4: Sometime, it showed many dialogue. For example, when I do the same thing, it has to do the same process.

10)

Experimenter: Do you have any suggestions for improving McFeSPA?

Participant 4: Am, no.

11)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?

Participant 4: 6

12)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?

Participant 4: 7

13)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?

Participant 4: 7

14)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Participant 4: 7, I can understand from this figure but I haven't used it before. I think it is easy to understand.

15)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Participant 4: Not sure about this one because it has more buttons and I didn't get use to it so I can't tell.

16)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?

Participant 4: 7, easy

17)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?

Participant 4: I like it, 7

18)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?

Participant 4: The same, 7

19)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?

Participant 4: 7, oh, it is easy to understand.

20)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?

Participant 4: Am, quite easy, 5.

21)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?

Participant 4: 7.

22)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?

Participant 4:7.

Experimenter: Right. Thank you very much for your help.

H.2.5 Participant 5's interview result

The results of Participant 5's interview (about 31 minutes)

(Experimenter: Plain text, Participant 5: *Italic*)

1)

Experimenter: Do you have any previous experience of giving feedback like some training?

Participant 5: *I have done a lot of teaching and lot of assignments in Prolog. I was the teaching system automated reasoning course where the students had to write a large prolog program so I don't know I can't remember how do I need to train in it but I done a lot of tutoring and also marking assignment in Prolog and in functional programming language as well.*

Experimenter: How long have you been marking or tutoring?

Participant 5: *I were tutoring for, I guess 2 years and teaching assistant for 2 years. This is overlapping about one year. Something likes that. So the 1st year was tutoring and teaching assistant, 2nd year was teaching assistant, 3rd year teaching like that. I can't remember exactly.*

Experimenter: Am Hm Thank you very much. Then do you think do you have some practical experience of giving feedback?

Participant 5: *Yeah, I have done giving feedback to student. So I guess a lot. Not written but I have done quite a bit. I mean when I was teaching I have done quite a lot.*

Experimenter: In your perspective, do you think do you have experience in giving feedback some or quite a lot or extensive.

Participant 5: *I think I have quite a lot.*

Experimenter: In giving feedback

Participant 5: *In giving feedback.*

Experimenter: OK, just find your perspective.

Participant 5: *This' my own perspective. I'm not talking relative to lecturers like many lecturers who have done for 10 or 15 years and I have done for 2. For tutors, a lot of their mark so I was 2nd PhD student then done very little.*

Experimenter: So, in your perspective, you have quite a lot.

Participant 5: *Yeah.*

Experimenter: OK.

2)

Experimenter: How much of the time (as a percentage) did you need to learn about the system itself and its functions? When 0% is no time spent on training and100% is all time spent on training). If you need the system to help you to mark programming assignment to give feedback, do you think how much do you need it? How much of time?

Participant 5: *I think what was needed would that maybe umm 2% or something. The system is very easy to use and I think I don't need any training at all umm but it did help to see it is being use two or three option in the menu. It is pretty obvious what I have to do.*

Experimenter: Right.

3)

Experimenter: Now, it is the pedagogical questions. In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 5: *Umm, I didn't realise it help me to learn to give feedback. It's a kind of interesting to just think about what the process is and for those reasons so task of considering what feedback to give is the kind of*

interesting. *Umm but I don't think those mention more about this than that.*

Experimenter: All right, next.

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback? You can use the system again if you'd to use.

Participant 5: *Well, if you really mean how than umm do I did it mostly was by giving the report to at the end or umm the assessment of what you did umm how well I did like the measuring thing like positive feedback, it presumably was measuring on how many time actually I give positive feedback rather than umm the actually feedback that I did. So umm in this sense, measurement is done very good job, but it was nonetheless* ***interesting to see that and good to see that.***

Experimenter: OK.

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 5: *No, I think. The most important thing, giving feedback is to actually understand what they are writing down, what the errors are umm and you give a hardwire that a hard part interpreting Prolog program, figuring out what it does, Laugh, and whether even produce the right answer where it's wrong and why it's wrong. This is the most difficult. Once you do that then it's quite easy to give sensible feedback.*

Experimenter: OK, next.

6)

Experimenter: In your view, what did you learn by using McFeSPA?

Participant 5: *Umm, so* ***I like the idea of separating of the different sections of design, implementation, style,*** *but it often feedback giving in the mixed ways. People, yeah, just like on the actual prolog program design different point of style, different point of the other thing. So* ***they give a summary of that. That's quite nice. I like that.*** *umm, they made me realise that positive feedback umm it's not about writing things.* ***It sounds nice but it is about when the students had done well in one of those sections, design...Implementation but they had done nicely than the good set...make me realise that notion of positive feedback with McFeSPA is clearly not what I want but I didn't really think about that before, so that sounds good.***

Experimenter: Thank you very much, and next.

7)

Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?

Participant 5: *Yeah, I think umm well, I mean I don't know about what I learn from using McFeSPA but I get so, the idea, I think, would work for the other programming language.*

8)

Experimenter: What do you like in particular about McFeSPA?

Participant 5: ***I like the idea. The idea of having a tutor that help student marking and the system that help tutor marking. I think that's great. This is nice to be compared between the students. If that more automated, it will be great. At the moment it's not almost automatic...It's not for***

the version I use...If it automated and I can use the preference with McFeSPA beforehand then I think it will be nice to use.

Experimenter: You haven't used the preference beforehand.

Participant 5: *Pardon, no, I didn't see the preference inside the menu so I don't aware you can figure the preference.*

Experimenter: I offer it in the customise.

Participant 5: *Yeah , in the customise.*

9)

Experimenter: What do you dislike in particular about McFeSPA? You can use it again.

Participant 5: *The lack of automation. Umm, the large of number of questions that ask you, so there are five or four different places that you can add in some sources of the description to do the design. You can add in at the very beginning. You can add it in after you click on the design thing. You can ask once you click on the design thing whether you wan t to add some which you can do...In the final report and in the creation of the report, so there are a lot of places that you can give this feedback on design, implementation, style. Each of those category the feedback in many places, it seems a bit redundant. I think it could be much simple. Is that make sense?*

Experimenter: Could you summarise that you said again, please.

Participant 5: *Laugh, umm what I found frustrating is that it repeated ask me many question. umm.*

Experimenter: For example.

Participant 5: *For example, would I like to give design feedback. It would be a box. I could do it many places. It sounds wrong could be able to add in design feedback so many places. I seems it should be automated generated the report, no clicking and then you can add it yourself. That would be much simple and much good I think... you just set it in the preference and you just do it. You load in a file and you just give this.*

Experimenter: This one, I have design and it's already in my design for the tutor who have experience in giving feedback but the system is to help people to give feedback that should not allow automated at this time.

Participant 5: *Yeah, I can see the system a bit... making for experiment not for being a real tutor and some sense that's difference. I don't think it will be much more difficult for someone who tries to learn. Actually, I don't think provide a lot of questions make it easier to learn. I think UMM having some experience and working with someone else usually or see other people's answer. That will make it help to learn...*

Experimenter: So, do you think learning with people is better than learning with the system.

Participant 5: *That's not necessary, but having UMM someone else to generate the report and then say that looks this is I think a good report, and then compare your report with that report. That'll help you to improve your ability to give feedback on a student. Having a system ask me would I like to do a history thing and then made me choose them, doesn't help me to give feedback. In fact, it ask me a lot of question, make me frustrating with it and I don't want. I want to go through as quick as possible, so I would learn to click the button very quickly*

without reading any thing...because I had to guess so what is going to be then I should suspect that would leave two errors something to correct feedback.

Experimenter: OK, anything else about dislike it. You can use it again.

Participant 5: *When you analyse it. When you load a file, I don't see why it helpful to UMM block to do the analysis so* ***when you load the file, you can always do the analysis, right. And then it said no errors. I don't need to here. When it said I don't find any design error. Do you want to add some? I can add some later why it asking now.***

Experimenter: UMM.

Participant 5: *...and* ***this sound be shown beforehand how do you like design, implementation, style error messages to appear. So I shouldn't have*** *to type any of this.* ***Do you want to add history...? Again this should be automated, should be something to be chosen beforehand...a lot*** *of giving of history thing but you don't have this error is good, right. If the student made a lot of this error last time, it's good. You don't make any error this time. That's a kind of positive feedback...I think the create feedback report should be pre-show I think...so the other comments that I had which I told you at the beginning is that it feel* ***like it should be a visit style with a little 'next' button and 'back' button*** *because you feeling in a load form, so I fell like you just be from to other form rather than this kind of sometime you click on the menu. Sometime you click on close.*

Experimenter: All right.

10)

Experimenter: Do you have any suggestions for improving McFcSPA?

Participant 5: *Am, I already gave you a lot*

Experimenter: Anything else about improving McFeSPA.

Participant 5: *Am, syntax colouring on code made difference on how easy to read. How quickly can be read? So probably you don't need to do this because it can be quite a lot of work but if you can syntax colour the code then you can often fix a lot of issue that you way out.*

Experimenter: All right

Participant 5: *It made you quickly to read*

Experimenter: Ok.

11)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?

Participant 5: *6, very easy*

12)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?

Participant 5: *5 or 6*

Experimenter: Any comments?

13)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?

Participant 5: *I found that's all the same. That's all pretty obvious 5 or 6.*

14)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Participant 5: *Am, a little bit... maybe 2 or 3 because you have to click on student's profile and then you have to compare the number of error here against to the number of errors here really you should have already selected this automatically. This no need to have this thing...*

Experimenter: Am, I'd like to check the user's giving 'feedback loop'

Participant 5: *Yeah, you want to check it.*

Experimenter: Then I can't do it automatically in learning to use the system

Participant 5: *Yeah.*

Experimenter: Once, perhaps people who would like to use the system should have separate of this.

Participant 5: ***Yeah, I don't know if you learn anything by. I don't think you lean anything by having these choices.***

Experimenter: All right

Participant 5: ***I think just saying "Is it interesting to include history?" that maybe teach you something but I don't think compare two numbers teach you something.***

Experimenter: All right, thank you very much.

15)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Participant 5: *Well, this interface I didn't understand actually. Well, I kind of understood it but... the interface a bit messy. You have a horizontal scroll in box here. You have to scroll left and right in order to read you line. I think this is a bit messy. I don't think it's very good.*

Experimenter: Any suggestion about this interface?

Participant 5: *...it should be a drop down menu really so I can see at the moment in front of me and I can select them...*

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Participant 5: *2 or 3, 2*

16)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?

Participant 5: *6, well, I mean its very easy to use... but it's still have close button here that unclear why you don't have back or forward thing.*

Experimenter: OK, next.

17)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?

Participant 5: *I guess 5 because that easy to understand*

18)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?

Participant 5: *That's standard textbox. That's very easy to understand as well. It's a bit unclear what happed this. I mean you got close button but presumably what your really want me to do it with save or send to student. I guess 6 because unclear about the button.*

19)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?
Participant 5: *I don't know what feedback loop and what asking question was? So in those things so I think 5*
Experimenter: Any suggestions about this interface?
Participant 5: *Am, I am not completely sure what it is trying to do or what the assessment. So say 3 times is a bit unclear. Should be percentage of the times, I don't really know why time you have done something is the right until and measure, in that way, a bit confusing to me.*
Experimenter: Previously I have presented this as percentage but the evaluators ask me, percentage of what then it made me think again. It made me confuse myself. Then I change it to times. Do you prefer percentages?
Participant 5: *Well, it depend what we are trying to measure and why we try to measure it...e.g. 3% of the time I give individual feedback and I did individual feedback for 100 times.*
Experimenter: Ok, thank you very much.
Participant 5: *You're welcome.*
20)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?
Participant 5: *Am, 6.*
21)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?
Participant 5: *That's the same, laugh, I did get much easier.*
22)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?
Participant 5: *Yeah, the third one is the same*
Experimenter: All right, thank you very much.

H.2.6 Participant 6's interview result

The results of Participant 6's interview (about 20 minutes)
(Experimenter: Plain text, Participant 4: *Italic*)
1)
Experimenter: Do you have any previous experience of giving feedback like some training?
Participant 6: *In giving feedback?*
Experimenter: Yeah.
Participant 6: *Yeah, Am, training to give feedback?*
Experimenter: Yeah,
Participant 6: *Well, I did PGCE which qualify to teach. Umm, so I have a reasonable in training.*
Experimenter: In your perspective, do you think do you have experience in giving feedback some or quite a lot or extensive.
Participant 6: *I think quite a lot. Do you mean in general or Prolog*

Experimenter: In general.
Participant 6: *Yeah, quite a lot.*
Experimenter: Do you have any comments about this?
Participant 6: *I think you get to know if you get to know the student personally it will help because they can. You may help initially various sources of problems. Some student response to how the feedback they got. Some students wouldn't realise that. Also some error might be taught might not be taught. And it is the time issue. You may, haven't got time to give detailed feedback. Problem...in paper.*

2)

Experimenter: How much of the time (as a percentage) did you need to learn about the system itself and its functions? When 0% is no time spent on training and100% is all time spent on training). If you need the system to help you to mark programming assignment to give feedback, do you think how much do you need it? How much of time?

Participant 6: 5-10 %

3)

Experimenter: Now, it is the pedagogical questions. In your view, how does McFeSPA help you increase your knowledge/understanding in the issue of learning to give feedback?

Participant 6: *Um, made me think about what mistake the students previously made and whether if they would make the errors. You want more detailed feedback or assuming have more detailed feedback at first time and then give less detailed feedback...if they made same errors you may give them detailed feedback. So this make me consider their previous report more than I have, I think.*

4)

Experimenter: In your view, how does McFeSPA help you reflect/rethink your skills in giving feedback? You can use the system again if you'd to use.

Participant 6: *Yeah, thinking emphasizing, well, the important thing is the history of students and also what types of errors they have made because I think it start up by saying I have to give feedback as much as possible every time and then stylistic error like missing indentation. You don't need to go into that every time and then if you penalise them, they will be upset. We want to give this as much as information we need but not more. Think about which feedback and which type.*

5)

Experimenter: Do you agree that McFeSPA can help you to give quality feedback to your students?

Participant 6: *Yeah, you mean actually using it as marking system as Prolog assignment*

Experimenter: Yeah.
Participant 6: *Yeah.*

6)

Experimenter: In your view, what did you learn by using McFeSPA?

Participant 6: *Yeah, I can just thinking about which even you got very few highlighting every error to the students. You might not want to do that. You might select what you emphasising. Am, just thinking about what to say? how much details you need to say?*

7)

Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?
Participant 6: *Yeah, oh, this is not every programming, I think*
Experimenter: Do you have any comments about this?
Participant 6: *Well, think the kind of general thing*
Experimenter: In you view, can you apply the knowledge you obtain by using the system to mark any programming assignment (not only Prolog)?
Participant 6: *Yeah, definitely*
8)
Experimenter: What do you like in particular about McFeSPA?
Participant 6: *Very simple to use, easy to learn how to use it. Am, and it helps the markers to think about the way in which they mark.*
Experimenter: Do you have any comments about this?
Participant 6: *Am, the page on defining the skill. I didn't find any thing useful. Am, because I didn't really know what each line mean, to be honest...*
9)
Experimenter: What do you dislike in particular about McFeSPA? You can use it again.
Participant 6: *Yeah, that page I mentioned.*
Experimenter: Skill meter?
Participant 6: *Yeah*
Experimenter: Any further comments about this?
Participant 6: *That's fine.*
10)
Experimenter: Do you have any suggestions for improving McFeSPA?
Participant 6: *Am, well, that page could be improve by explaining, maybe it should may, I'm not sure but explaining what about. Am, I think kind sort of half way to use it. What point is...?*
Comments: PT6 don't know that the glossary can help PT6 to know the explaining of each feedback.
11)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Main' interface?
Participant 6: *6 because I don't know three 'Add extra...' buttons.*
12)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for More Errors' interface?
Participant 6: *Yeah, I think 7.*
Comments: PT6 understand that 'None (just brief message)' mean not give any message but in fact it is brief feedback message.
13)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Choices for One Error' interface?
Participant 6: *Am, let's say 6.*

14)
Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Taking into history of student's errors' interface?

Participant 6: *5, you should have student's profile up to somewhere out, not pop up in the same areas. That will be easy to use.*

15)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Add Extra Design Error' interface?

Participant 6: *6.*

16)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Feedback Template' interface?

Participant 6: *Yeah, that pretty obvious, 7.*

17)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Create Feedback Report' interface?

Participant 6: *7, that's easy*

18)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Draft Feedback Report' interface?

Participant 6: *Yeah, 7.*

19)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand the 'Skill Meter' interface?

Participant 6: *2, I understand that it was feedback for me but I don't know what ground of judging, you know.*

Experimenter: Yeah, it is in the glossary

Participant 6: *Yeah, I should see it in the glossary.*

20)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the first time to mark the first script?

Participant 6: *Probably say 5.*

21)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the second time to mark the second script?

Participant 6: *7.*

22)

Experimenter: On the scale of 1 to 7, where 1 is very difficult and 7 is very easy, how easy was it to understand McFeSPA when you use the system the third time to mark the third script?

Participant 6: *7.*

Experimenter: Right. Thank you very much for your help.

Appendix I -- Suggestion to improve McFeSPA

Table I.1 is for justify the evaluators' suggestion (in this chapter) and some participants' suggestion (PT1 – PT3, PT5, and PT6 from next chapter) to improve McFeSPA. The '√' symbols from all users are the suggestions to improve the interface while the '√' symbol of results is the suggestions are done. The 'N' symbol means the interface will be updated in the next version. The 'P' symbol means the interface was partly updated in the current version. The 'X' symbol is that the suggestion will not be taken into account.

Suggestions from users	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
EV_1	√	√	√	√	√	√	√	-	-	-	√	-	-	-	√	-	√	-	-	-	-
EV_2	-	-	-	√	√	√	√	√	√	√	√	√	√	√	-	√	√	√	√	√	√
EV_3	-	-	-	√	√	-	√	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PT_1	-	-	-	-	-	-	-	-	-	-	-	-	-	√	√	-	-	-	-	-	-
PT_2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PT_3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PT_5	-	-	-	-	-	-	-	-	-	-	-	√	-	-	-	-	-	-	-	-	-
PT_6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	√	-	-	-	-	-	-
Results	√	√	√	√	√	√	√	√	√	√	√	X	√	N	X	N	√	√	√	√	√

Table I.1 Evaluators' and participants' suggestion to improve McFeSPA and the results

Suggestions from users	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
EV_1	√	-	-	√	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EV_2	√	√	√	√	√	√	√	-	-	√	√	-	-	-	-	-	-	-	-	-	-
EV_3	-	-	√	-	-	-	-	√	√	-	-	-	-	-	-	-	-	-	-	-	-
PT_1	-	-	-	-	-	-	-	-	-	-	-	√	√	√	√	√	√	√	√	√	-
PT_2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PT_3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PT_5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	√
PT_6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Results	√	N	P, N	N	√	√	√	X	√	N	√	N	N	N	N	N	N	N	N	N	N

Table I.1 Evaluators' and participants' suggestion to improve McFeSPA and the results (contd. 1)

Suggestions from users	43	44	45	46	47	48	49	50	51
EV_1	-	-	-	-	-	-	-	-	-
EV_2	-	-	-	-	-	-	-	-	-
EV_3	-	-	-	-	-	-	-	-	-
PT_1	-	-	√	-	-	√	-	√	√
PT_2	-	-	-	-	-	-	-	-	√
PT_3	-	-	-	-	-	-	-	√	-
PT_5	√	√	-	√	-	-	√	-	-
PT_6	-	-	-	-	√	-	-	-	-
Results	N	N	N	N	N	N	N	N	N

Table I.1 Evaluators' and participants' suggestion to improve McFeSPA and the results (contd. 2)

According to the Table 1 the number 1-51 represented all suggestions from all evaluators and participants from evaluation of McFeSPA (in next chapter), the explanation of each number can be seen below.

I.1 Suggestion in which the interface was updated in the current version:

1) Clear feedback message in the 'Design/Implementation/Style' pane of the main interface before reading new student's solution file.

2) The marker's name should appear after any help message pop up rather the default of user name from McFeSPA. (This has been updated by changing the default marker name to be the current marker name in customisation menu/item.)

3) Change the pop up message before McFeSPA opens 'Add extra sentence after the error messages' interface from 'Do you need to take into account history of student's error?' to be 'Do you need to add extra sentence beyond this by taking into account history of student's error?' (see Conversation#32 in Appendix F).

4) Display the particular student's profile (not all) of the student who was marked and need to be taking into account the history of student's errors. All evaluators commented on this interface and we do agree with this suggestion (see Conversation#29 and #54 in Appendix F).

5) Move the pop up message of 'Good, you are going to progress in giving .....' to the other part of the interface (not the middle of the interface). All evaluators commented on this interface and we do agree with this suggestion (see Conversation#9, #38, and #49 in

Appendix F). (This has been updated in the current version by reducing the encouraging messages from McFeSPA. The other pop-up messages are still be in McFeSPA because the main purpose of scaffolding system is to offer help to assist the user to go the right path.)

6) Change the pop up message "Do you double check the feedback report! Otherwise, you may miss some useful feedback" to be "Remember to double check the feedback report! Otherwise, you may miss some useful feedback", and changing yes/no buttons to be ok button. EV1 and EV2 commented on this interface and we do agree with this suggestion (see Conversation#10 in Appendix F).

7) Explain the criteria of the skill meter in the 'Skill meter' interface (see Conversation#16, #43, and #52 in Appendix F); We do agree with this because there are also some problems in passing variable this in this interface leads to the repeating results of the explanation of the skill meter; thus, the interface was updated and we also allow the user to view criteria of the skill meter from the 'Glossary' interface.

8) Change the word in 'Favourite wording' interface from 'between' to be 'between line'; and from 'and' to be 'and line' (see Conversation#21 in Appendix F).

9) Change the order of two list boxes in 'Favourite wording' interface i.e. The order of 'variable' list box come first then followed by the 'predicate' list box.

10) Provide the example of using each word in 'Favourite wording' interface (see Conversation#21 in Appendix F). This could help the user understand using this word; thus, the interface was updated in the current version.

11) Move 'Add', 'Update' and 'Clear' button in to the left bottom of the in 'ManageData' interface and move the 'Delete' button to under the list box but above the text box in the interface. (EV_1 said that the using the 'update' button made the EV_1 confuse (see Conversation#2 in Appendix F), EV_2 said that the position of some buttons in the 'ManageData' interface are not clear (see Conversation#19 in Appendix F)).

13) In the main interface, present the line number in the student's solution file in front of the student's script in each line, in the left most of the pane (see Conversation#25 in Appendix F).

17) Change the message 'Something in your selection doesn't work for you...' of the help level 1 to be any sentence should be discussed with the English native (see Conversation#3, #28 in Appendix F).

18) There should not have the 'Student Profile' menu item in the 'Choices for more errors' and 'Choices for one error' interface (see Conversation#26 in Appendix F).

19) Glossary menu in 'Choices for more errors' should open the glossary interface.

20) Change the title from 'Add extra sentence after the error messages' to be 'Taking into history of student's error'. The new title could help the user understand rather than the previous one (EV_1 also suggested change the pop up message before McFeSPA opens 'Add extra sentence after the error messages' interface from 'Do you need to take into account history of student's error?' to be 'Do you need to add extra sentence beyond this by taking into account history of student's error?' see Conversation# 5, in Appendix F).

21) There should not have the 'Refresh' button in 'Add extra design error' interface (see Conversation#33 in Appendix F).

22) Change the 'Exit' menu item of 'Create Feedback Report' interface and 'Draft Feedback Report' interface to be 'close' menu item (see Conversation# 12, #36 in Appendix F).

26) Improve the language used in skill meter explanation. (This was updated in the current version by discussion with native English speakers).

27) There should have 'glossary' button/menu item in the 'Skill meter' interface. We do agree with this; thus, this interface was updated in the current version.

28) The skill meter of 'asking question' should not tell that 1 word mean 10% but should state the number of word in the skill meter instead of percentage.

30) Change the 'Student Profile' menu item to be 'Student Profile' button at the left bottom of the 'Add extra sentence after error messages' interface (This is true because the EV_3 tried to find 'Student Profile' menu item but the EV_3 could not see it; thus the EV_3 suggest to change it into the button).

32) Provide a cancel button in the 'Choice for more error' and 'Choice for one error'

(see Conversation# 24, in Appendix F).

I.2 Suggestion in which the interface will be updated in the next version:

12) Represent the student's file, in the main interface, in term of student's name (see Conversation#22 in Appendix F). This is not the primary problem to improve McFeSPA in the next version.

14) In the main interface, when double selecting the list of analysed student's solution, the previous error message/feedback message should be clear (deleted) (Experimenter: McFeSPA does not clear the previous error messages/feedback messages because McFeSPA allows the users to delete the previous error message/feedback message by themselves). This is a good suggestion but not the main aim of system to improve McFeSPA in the current version.

15) Group the same check box in the 'Choices for more errors' and 'Choices for one error' interface. (see Conversation#8 in Appendix F). This is a good suggestion but not the main aim to improve McFeSPA in the current version and any checkboxes that not belong to any options will not be active and not make the user confuse.

16) In the 'Choices for more errors' and 'Choices for one error' interface, the option for including an example should not be shown in whole solution, it should be some part of the solution to be an example (see Conversation#35 in Appendix F). This is a good suggestion but not the main aim of system to improve McFeSPA in the current version.

23) Not allow the user to select the wrong item in 'Change order in report' area of the 'Create feedback report' interface. McFeSPA should protect this automatically (see Conversation#42 in Appendix F). This is a good suggestion but not the main aim to improve McFeSPA in the current version.

24) Should present three panes of 'Temporary Feedback message' in a button and click such button to preview all error messages/feedback messages (not allow to type) and take the warning message with regard to this interface off. (EV_2 said that these panes are useless if McFeSPA inform warning message and the EV_2 can type any message in each pane (see Conversation#37, #41 in Appendix F); EV_3 said that the temporary of

each feedback messages (design/implementation/style) are not helpful when they are shown each pane. They should be shown all in the same pane. If they are reordered, it is easy to view the changed order (see Conversation#47 in Appendix F);

Experimenter's comment: this was partly updated:- take the warning message off and not allow the user to type any thing in three panes of error messages/feedback messages. We could not update all templates of 'Create feedback report' interface according to the EV_2's requirement (Should present three panes of 'Temporary Feedback message' in a button and click such button to preview all error messages/feedback messages) because this will affect to the 'Create Feedback Report' for the (left most lower) template (from 'Feedback Template' interface). This suggestion is important to take into account to update the system in the next version.

25) McFeSPA should allow print the report and send e-mail to the student in 'Draft Feedback Report' interface (see Conversation#10, #39 in Appendix F). This is a good suggestion but not the main aim to improve McFeSPA in the current version and will be taken into account to update the system in the next version.

29) Expand the 'Show student's solution' pane to be bigger than the previous one. This is not primary aim to improve McFeSPA in the current version because it is a low level interface not relate to the goal. (It is less important to update.)

31) Allow the user to undo the previous action. (see Conversation# 23 in Appendix F). Even though this is a good suggestion and also undo is a good mechanism in the authoring process (Kim, Whang, Lee, & Kim, 1999), the other evaluators did not comment on this. Regarding sufficient implementation for pilot testing, "Undo" mechanism is judged not to implement in the current system because there is very few continuous action for each step in which the user can click cancel or exit the process and start each step easily again. (It is important to update.)

33) PT1 need a help file from the system ("*I didn't see is there any help file at all. I expect that to be in the system, the help organise how I can click on...*") while using the system without scaffolding. In fact, glossary in the system is like a help file. However, these should have an instruction file provided in the system similar to handout in the experiment so that PT1 could use the system alone without any support from the system.

This could be supported by providing the handout of using the system as a help file into the system. (It is important and easy to update.)

34) PT1 thought the range of feedback is limited (*"It will be nice if you have a broader range of sample sentences").* However, this is not the main issue of learnability. It is the usability issue. **If we achieve examples of feedback files, we could plug in all comments to the system so that all TAs will have several examples of feedback messages to select and adapt to use it.** (It is less important to update.)

35) Allow user generate feedback report without clicking all lists of errors which are analysed by the system because PT1 think there are some errors PT1 thought that they are not important to inform the students. However, in the current version, the system does not allow the user generate feedback report if the user hasn't clicked all lists of errors which are analysed by the system. (It is interesting to update.)

36) Should plug Prolog module near the student's solution in order to run the student's script from compiler again. (It is interesting to update.)

37) There should have feedback message displayed the result of selecting options from both 'Choices for one error' interface and 'Choices' for more errors' interface to let the user know how their selecting going on. (It is important easy to update.)

38) The system should pop up only the corresponded feedback detail with the history of student's errors. (It is important to update.)

39) The system should give student's code back into the feedback report to write comment to any line number. (It is important and easy to update.)

40) The system should not display the student's name during marking. It may not be fair to some students. We argue that this is not the exam so it is necessary to know the student's name and individual student's history of their errors. (It is less important but easy to update.)

41) After closing 'Draft feedback report' interface, the 'Create feedback report' interface should be closed too without asking the user to confirm to close it again and the system should tell the user free to edit the message in the 'Draft feedback report' interface (It is not important but easy to update.)

42) The system should analyse student's file immediately after file was loaded. Not to click anayse button again. (It is less important but easy to update.)

43) The system should not ask every time when the system could find any error after anallyse it because the user can add it later if the user need. (It is less important but easy to update.)

44) The system should generate the additional message automatically in comparison the current student's errors with the previous one. (It is less important but easy to update.)

45) The system should display the student's profile automatically. However, in order to check the user's giving feedback loop, we need to provide a button to check whether the user taking into the history of student's errors. (It is less important but easy to update.)

46) The system should provide 'next' and 'back' button (It is important to update.)

47) The system should have syntax colouring on code to help the marker read the code. (It is less important to update.)

48) The system should provide more option beyond 'Add', 'Delete', 'Update' button in the 'Add design errors' interface. However, we implemented the evaluation version enough to test our hypothesis. Thus, some features were ignored in the experiment version. (It is less important to update.)

49) Adjust the error messages in the 'Add design errors' interface to be displayed fit in the text box in the horizontal scroll.

50) 'Feedback template' interface, pretty blur, should be clear.

51) Each explanation to the quality feedback in the 'skill meter' interface should be explained under each bar of skill meter

In order to facilitate the developer in re-implementing the system in the next version, we have sequence the suggestions of the evaluators and the participants from high to low importance which relate to improving McFeSPA in the next version. We have groups such suggestions into six groups. The first group is high importance to take into account and easy to improve. These are the suggestion #37, #39. The second group is, also high important to consider, the suggestion #14, #16, #24, #25, #31, #33, #45. The third group is, important to concern and also easy to improve, the suggestion # 43. The fourth group is, moderate important, the suggestion #35, #36. The fifth group is less important to take

into account but easy to update. These are the suggestion #12, 40, 41, 42, 44. The last group is, less important to consider, the suggestion # 15, 23, 34, 46.

Appendix J -Plan and Timetable of the Research

Year 1 (2003)

1. Preparation Study and Literature Review	January 2003	- March 2003
2. Initial Project Approval	April 2003	
3. Learning and checking the potential methods	March 2003	- November 2003
4. Further Literature Review	May 2003	- December 2003
5. Exploring & playing with previous system	October 2003	- December 2003

Year 2 (2004)

1. User requirement & Specification	November 2003	- January 2004
2. Design & Implementation preliminary scaffolding system	December 2003	- March 2004
3. Further Literature Review	March 2004	- June 2004
4. Mid Point Progression Report Preparation	April 2004	- June 2004
5. Mid Point Review	July 2004	
6. Redesign & Further literature review	June 2004	- March 2005
7. Usability prototype design	June 2004	- August 2004
8. Interactive prototype design	September 2004	- October 2004

Year 3 (2005)

1. Functionality prototype design	October 2004	- December 2004
2. Decision approach design	December 2004	- January 2005
3. Design analysis	January 2005	- March 2005
4. Refinement (Redesign & Reimplementation)	April 2005	- June 2005
5. Evaluation Design & Further literature review	June 2005	- July 2005
6. Usability prototype evaluation	August 2005	- February 2006

Year 4 (2006)

1. Refinement	September 2005	- December 2005
2. Usability evaluation	January 2006	- March 2006
3. Refinement	April 2006	- May 2006
4. Final evaluation	June 2006	- July 2006

5. Further literature review — August 2006

Year 5 (2007)

1. Writing up the Thesis — October 2004 - January 2007
2. Thesis Submission — February 2007

N.B. - Writing of thesis is an ongoing process throughout the project.

- Methods in this plan are the methodology in Table 1.1 in Chapter 1.