Nelson Mandela Metropolitan University

for tomorrow

Department of Computer Science and Information Systems

An Intelligent User Interface Model for Contact Centre Operations

---

Akash Singh

Supervisor: Prof J.L. Wesson

December 2007

Submitted in partial fulfilment of the requirements for the degree of
Magister Scientiae in Computer Science and Information Systems
at the Nelson Mandela Metropolitan University

Acknowledgements

The author wishes to acknowledge the following individuals and groups for their continued assistance and support throughout this research:

- My supervisor, Prof. Janet Wesson, for her invaluable support, guidance and encouragement. Her dedication in her role as a supervisor has inspired and motivated me to perform to the best of my capabilities at all times. I would especially like to thank her for the long hours she spent reading the various versions of this dissertation and for the constructive feedback that she gave to improve its content and structure;
- The Nelson Mandela Metropolitan University ICT Service Desk for their valued support and assistance;
- The staff at the Department of Computer Science and Information Systems at the Nelson Mandela Metropolitan University for their valued feedback and suggestions; and
- The Telkom Centre of Excellence programme for the necessary funding and technology for making this research possible.

Table of Contents

Chapter 1: Introduction 1

1.1 Background...1

1.2 Research Rationale...2

1.2.1 Contact Centres 2

1.2.2 Intelligent User Interfaces..5

1.3 Research Outline 6

1.3.1 Problem Statement..6

1.3.2 Scope and Constraints..6

1.3.3 Research Objectives 6

1.3.4 Research Questions 7

1.3.5 Research Design 9

1.3.5.1 Methods...9

1.4 Dissertation Structure 10

Chapter 2: Contact Centres 13

2.1 Introduction 13

2.2 Overview 13

2.3 Types of Contact Centres 14

2.3.1 Inbound Contact Centres 14

2.3.2 Outbound Contact Centres... 15

2.3.3 Help Desks 15

2.3.3.1 Types of Help Desks 15

2.3.4 Services Desks.. 16

2.3.4.1 Types of Service Desks 18

2.4 Existing Software 20
2.4.1 Axio Systems–Assyst 22
2.4.2 Livetime Help Desk 23
2.4.3 SiteHelpDesk-IT 24
2.4.4 FrontRange Solutions-Heat 24
2.4.5 Serio Service Desk 27
2.4.6 Polar Help Desk 28
2.4.7 Software Comparison 29
2.5 Challenges 30
2.5.1 Customer Satisfaction 31
2.5.2 Cost Reductions 32
2.5.3 Employee Satisfaction 33
2.5.4 Information Challenges 33
2.6 Call Resolution Process 34
2.7 Conclusions 37
Chapter 3: Intelligent User Interfaces 39
3.1 Introduction 39
3.2 Background 39
3.3 What is an IUI? 40
3.4 Intelligent Interfaces vs. Traditional Interfaces 41
3.4.1 Interface Intelligence 42
3.4.2 Accommodating Change 42
3.4.3 Change Protocols 43
3.4.4 Inclusion of Artificial Intelligence 43
3.5 Types of IUIs 45

3.5.1 Direct Manipulation Interfaces 45

3.5.2 Intermediary Interfaces 45

3.5.3 Agent Interfaces 46

3.6 IUI Components 46

3.6.1 The User Model 47

3.6.2 Multimodal Communication 50

3.6.3 Plan Recognition 52

3.6.4 Dynamic Presentation 54

3.6.5 Natural Language Processing 54

3.6.6 Intelligent Help 55

3.6.7 Interface Adaptability 56

3.6.8 Intelligent Agents 58

3.7 Conclusion 60

Chapter 4: Model-Based Design of IUIs 62

4.1 Introduction 62

4.2 What is a Model? 62

4.3 Characteristics and Quality of a Model 63

4.4 Types of Models 64

4.5 Elements of a Model 64

4.6 Stages of a Model 65

4.7 Existing IUI Models 66

4.8 Conclusion 73

Chapter 5: Model Design 75

5.1 Introduction 75

5.2 Current Limitations and Modifications 75

5.2.1 Limitations 75

5.2.2 Modifications 76

5.3 Proposed Model 77

5.3.1 Architecture 78

5.3.1.1 Input/Output Manager 79

5.3.1.2 Plan Manager 79

5.3.1.3 Agent Manager 80

5.3.1.4 Knowledge Base 81

5.3.1.5 Presentation Manager 81

5.3.2 Component-Level Design 82

5.3.2.1 Task Model 82

5.3.2.2 User Model 83

5.3.2.3 Solution Model 85

5.3.3 Interface Design 86

5.4 Conclusion 88

Chapter 6: Implementation 90

6.1 Introduction 90

6.2 Architecture 90

6.2.1 Plan Manager 92

6.2.2 Agent Manager 93

6.2.3 Knowledge Base 97

6.2.3.1 The Task Model 99

6.2.3.2 The User Model 100

6.2.3.3 The Solution Model 100

6.3 The User Interface 100

6.3.1 The Query Diagnosis Window 101
6.3.1.1 Task Status Section 101
6.3.1.2 User Input Section 103
6.3.1.3 System Feedback Section 104
6.3.2 Query Resolution Window 106
6.3.3 Customer Query Scenario 108
6.3.3.1 Customer Identification 109
6.3.3.2 Query Diagnosis 112
6.3.3.3 Query Resolution 115
6.4 Conclusion 119
Chapter 7: Evaluation 121
7.1 Introduction 121
7.2 Existing Methods and Instruments 121
7.3 Model Evaluation 122
7.4 User Testing 130
7.4.1 Evaluation Methodology 131
7.4.1.1 Goals 132
7.4.1.2 Measurements and Metrics 132
7.4.1.2.1 Usability Measurements 135
7.4.1.2.2 Eye-Tracking Metrics 135
7.4.1.3 Instruments 136
7.4.1.4 Procedure 137
7.4.2 Participant Selection 138
7.4.3 Results 142
7.4.3.1 Satisfaction Results 142

7.4.3.1.1 Quantitative Analysis 142
7.4.3.1.2 Qualitative Analysis 149
7.4.3.1.2.1 Negative Comments 149
7.4.3.1.2.2 Positive Comments 150
7.4.3.2 Effectiveness Results 151
7.5 Implications 160
7.5.1 Model Evaluation Suggestions 160
7.5.2 User Testing Suggestions 160
7.6 Conclusions 161
Chapter 8: Conclusions and Recommendations 163
8.1 Introduction 163
8.2 Research Objectives Revisited 163
8.3 Research Achievements 164
8.3.1 Theoretical Achievements 164
8.3.2 Practical Achievements 165
8.4 Research Contributions 166
8.4.1 Theoretical Contributions 166
8.4.2 Practical Contributions 168
8.5 Recommendations 169
8.5.1 Recommendations for Theory 169
8.5.2 Recommendations for Practice 170
8.5.3 Recommendations for Future Work 170
8.6 Summary 171
References 172
Appendix A: Field Study Questionnaire 181

Appendix B: Usability Evaluation Test Plan 190

Appendix C: Post-Test Questionnaire.. 198

Appendix D: Biographical Profile of Test Participants 201

Appendix E: User Satisfaction Results 202

Appendix F: Task Completion Rates 205

Appendix G: Eye-Tracking Data 206

List of Figures

Figure 1.1: Current Model of CC Operations....4
Figure 1.2: Extended Model of CC Operations using an IUI....5
Figure 1.3: Dissertation Structure.... 10
Figure 2.1: Contact Centre Types and Structures.... 14
Figure 2.2: Proactive Service Desk (ITIL 2006).... 19
Figure 2.3: Assyst Mobile Interface (Axio Systems 2006) 23
Figure 2.4: Call Management Screen from SiteHelpDesk-IT (SiteHelpDesk.com 2006) 25
Figure 2.5: HEAT Call Log Screen.... 26
Figure 2.6: Serio Service Desk Feature Overview (Serio 2006).... 27
Figure 2.7: Polar Help Desk Track Incidents Screen (Polar Software 2006).... 28
Figure 2.8: CC Management Balancing Act (Mehrota and Fama 2003).... 30
Figure 2.9: Current Call Resolution Process 35
Figure 2.10: Call Resolution Process using an IUI 36
Figure 3.1: Current Interface Practice and its Relation to IUIs (Maybury 1998:4) 41
Figure 3.2: Contribution of AI to Interfaces (Maybury 1998) 44
Figure 3.3: Generic Multimodal I\O Architecture (Trompf 2001).... 51
Figure 3.4: Human - Agent Collaboration (Lesh *et al.* 2001) 52
Figure 3.5: Collagen e-mail with and without plan recognition (Lesh *et al.* 2001).... 53
Figure 3.6: General Processing Schema for Adaptive Systems (Jameson 2003).... 56
Figure 3.7: Structure of an IA Application (Lieberman 1997).... 59
Figure 3.8: User Model Processes (de Vrieze *et al.* 2004) 60
Figure 4.1 : Traditional Interface Architecture (Maybury 2001) 66
Figure 4.2: Intelligent Interface Architecture (Tyler *et al.* 1991).... 69
Figure 4.3: Focus Architecture (Edmonds 1993) 70
Figure 4.4: Adaptive Intelligent Interface Architecture (Hefley *et al.* 1993).... 71
Figure 4.5: L - CID Model of a Self Adaptive Intelligent Interface (Puerta 1993).... 72
Figure 5.1: Modified IUI Architecture.... 76
Figure 5.2: Proposed IUI Model for CCs.... 78

Figure 5.3: Architecture of the Proposed IUI Model 78
Figure 5.4: Extract Task Model Schema 83
Figure 5.5: Extract of User Model Schema 84
Figure 5.6: Extract of Solution Model Schema 85
Figure 5.7: Low-Fidelity IUI Design Template 87
Figure 6.1: Activity Diagram of the IUI Prototype 91
Figure 6.2: Plan Manager Class 92
Figure 6.3: Agent Manager Class 94
Figure 6.4: Validation Class 95
Figure 6.5: Extract of Validation Class (isAlpha() Method) 96
Figure 6.6: Knowledge Base Class 98
Figure 6.7: Task Model Methods used by the Plan Manager 99
Figure 6.8: Query Diagnosis Window 101
Figure 6.9: Task Status Information Section 102
Figure 6.10: Visual Step Indicator Colour Chart 102
Figure 6.11: User Input Section 103
Figure 6.12: System Feedback Section 104
Figure 6.13: Standard Trie (Goodrich and Tamassia 2002) 105
Figure 6.14: Query Resolution Window 107
Figure 6.15: Identification of Customer Details (Step 1) 110
Figure 6.16: Selection of Customer Details (Step 1) 111
Figure 6.17: Diagnosis of Customer Query (Step 2) 113
Figure 6.18: Dynamic Query Categorisation (Step 2) 114
Figure 6.19: Resolution of Customer's Query (Step 3) 116
Figure 6.20: Query Resolution Window (Step 3) 117
Figure 6.21: Resolution of Customer's Query (Step 3) 118
Figure 7.1: ISD – Query Diagnosis Window 123
Figure 7.2: ISD - Error Detection 126
Figure 7.3: ISD - Providing Guidance 127
Figure 7.4: ISD - Rapid Task Completion 129
Figure 7.5: Eye-Tracking AOIs 133

Figure 7.6: Gender Profile of Test Participants (n=10) 139
Figure 7.7: Occupational Profile of Test Participants (n=10)... 140
Figure 7.8: Age Profile of Test Participants (n=10)... 140
Figure 7.9: Computing Experience Profile of Test Participants (n=10) 141
Figure 7.10: User Satisfaction Results (n=10).. 143
Figure 7.11: Overall User Reaction Results (n=10).. 144
Figure 7.12: Interface Design Satisfaction Results (n=10).. 145
Figure 7.13: Terminology and System Information Satisfaction Results (n=10) 146
Figure 7.14: System Feedback Satisfaction Results (n=10) 147
Figure 7.15: Task Support Satisfaction Results (n=10) 148
Figure 7.16: Tutorial Task Completion Rates Results (n=10).. 152
Figure 7.17: Evaluation Task Completion Rates Results (n=10).................................... 153
Figure 7.18: Tutorial Heatmap – Total Fixation Time (n=10) 154
Figure 7.19: Tutorial Fixation Duration (n=10) 155
Figure 7.20: Tutorial Observation Count (n=10) 156
Figure 7.21: Evaluation Heatmap – Total Fixation Time (n=10).................................... 157
Figure 7.22: Evaluation Fixation Duration (n=10)... 158
Figure 7.23: Evaluation Observation Count (n=10)... 159
Figure 8.1 : IUI Model for CCs 166
Figure B.1: IUI Prototype – Query Diagnosis Window... 191
Figure B.2: IUI Prototype – Query Resolution Window 193

List of Tables

Table 1.1: Research Questions and Methodology..9
Table 2.1: Advantages and Disadvantages of Help Desk Structures (Sanderson 2003)... 17
Table 2.2: Advantages and Disadvantages of Service Desk Structures............................ 21
Table 2.3: Service Desk Software Comparison.. 29
Table 3.1: Six Questions for an Intelligent Interface (Rich *et al. 2001*)........................... 42
Table 4.1: Typical Characteristics of a Model (Olivier 2006)... 63
Table 4.2: Quality Assurance Factors of Model Design (Aagedal and Mohagheghi 2007) 63
Table 4.3: IUI Architecture / Model Evaluation Criteria.. 68
Table 7.1: Proposed Methodology (Pretorius *et al.* 2005)... 132
Table D.1: Biographical Data of Test Participants (n=10)... 201
Table E.1:Overall User Satisfaction - Summary Analysis (n=10).................................. 202
Table E.2: Overall User Reaction - Detailed Analysis (n=10)... 202
Table E.3: Interface Design - Detailed Analysis (n=10).. 203
Table E.4: Terminology & System Information - Detailed Analysis (n=10).................. 203
Table E.5: System Feedback - Detailed Analysis (n=10)... 203
Table E.6: Task Support - Detailed Analysis (n=10).. 204
Table F.1: Tutorial and Evaluation Task Completion Rates (n=10)................................ 205
Table G.1: Tutorial Fixed Duration and Observation Count Data (n=10)....................... 206
Table G.2: Evaluation Fixed Duration and Observation Count Data (n=10).................. 207

List of Abbreviations

ACD	Automatic Call Distributor
AI	Artificial Intelligence
AOI(s)	Area(s) of Interest
API	Application Program Interface
ASP	Active Server Pages
CC(s)	Contact Centre(s)
CCA(s)	Contact Centre Agent(s)
CRR	Call Resolution Rates
CS&IS	Computer Science and Information Systems
CSR(s)	Customer Service Representative(s)
CTI	Computer-Telephony Integration
FCR	First Call Resolution
HCI	Human-Computer Interaction
IA(s)	Intelligent Agent(s)
ICT	Information and Communications Technology
IDE	Integrated Development Environment
ISD	Intelligent Service Desk
IT	Information Technology
ITIL	IT Infrastructure Library
IUI(s)	Intelligent User Interface(s)
IVR	Interactive Voice Response
J2EE	Java 2 Enterprise Edition
NMMU	Nelson Mandela Metropolitan University
QUIS	Questionnaire for User Interaction Satisfaction
SLA	Service Level Agreements
SLM	Service Level Management
SMS	Short Messaging Service
SQL	Structured Query Language

UI(s)	User Interface(s)
UML	Unified Modelling Language
VoIP	Voice over Internet Protocol
XML	Extensible Markup Language

Summary

Contact Centres (CCs) are at the forefront of interaction between an organisation and its customers. Currently, 17% of all inbound calls are not resolved on the first call by the first agent attending to that call. This is due to the inability of the contact centre agents (CCAs) to diagnose customer queries and find adequate solutions in an effective and efficient manner.

The aim of this research is to develop an intelligent user interface (IUI) model to support and improve CC operations. A literature review of existing IUI architectures, model-based design and existing CC software together with a field study of CCs has resulted in the design of an IUI model for CCs. The proposed IUI model is described in terms of its architecture, component-level design and interface design.

An IUI prototype has been developed as a proof of concept of the proposed IUI model. The IUI prototype was evaluated in order to determine to what extent it supports problem identification and query resolution. User testing, incorporating the use of eye tracking and a post-test questionnaire, was used in order to determine the usability and usefulness of the prototype. The results of this evaluation show that the users were highly satisfied with the task support and query resolution assistance provided by the IUI prototype.

This research resulted in the design of an IUI model for the domain of CCs. This model can be used to assist the development of CC applications incorporating IUIs. Use of the proposed IUI model is expected to support and enhance the effectiveness and efficiency of CC operations. Further research is needed to conduct a longitudinal study to determine the impact of IUIs in the CC domain.

Keywords: Contact centres, intelligent user interfaces, model-based design, task models, user models, user interfaces

Chapter 1: Introduction

1.1 Background

A traditional call centre can be defined as an operation which assists customers and delivers a value-added service through the use of telephone and various computer technologies (Australian National Audit Office 1996; Gans, Koole and Mandelbaum 2003; Parbhoo 2002). In addition to resolving customer queries, call centres can also be regarded as an intermediary for the communication of information between a company and its customers (Australian National Audit Office 1996). Initially unpopular due to their high operational costs, call centres can improve customer service by providing increased access for customers (Siebel System 2005). This allows for increased efficiency in the completion rate of customer queries or transactions. The end result is improved customer satisfaction and a more effective and efficient means of delivering a customer service (Australian National Audit Office 1996).

The term contact centre (CC) refers to a call centre that provides a value-added service to customers via more than one electronic medium (Avaya, Inc. 2006; ECA 2006). Customers seeking to utilise this service are typically assisted by contact centre agents (CCAs), through the use of telephones. A typical example of a CC would be a call centre that handles calls via telephone, e-mail, facsimile and interaction via the Internet (ECA 2006).

Interaction within a CC can occur between customers and the CCAs, between the CCAs and the computer and between the customer and the computer (Steel 2003). Two critical issues relating to CC operations are the reduction in call durations and increased customer satisfaction (Steel 2003). CCs require integrated software solutions (applications) in order to reduce time spent handling calls and to increase customer satisfaction. To a large extent, this integration may be achieved through the inclusion of multimedia elements in CC applications (Aspect Communications Corporation 2004).

Intelligent user interfaces (IUIs) are one of the ways in which to incorporate multimedia elements in order to produce applications that are more efficient, effective and natural (Maybury 1999). IUIs can be viewed as a means to amplify the interaction and the rate of information flow between humans and computers (Maybury 1993). Interfaces and users can benefit from IUIs by:

- Reducing the complexity of application, through guidance from the application itself (Koelle 2004);
- Using multimodal communication through IUIs to increase the accessibility of computers (Koelle 2004); and
- Providing rapid task completion with less effort than with traditional interfaces (Maybury 1999).

This section has provided a brief overview into the domain of call centres, CCs and the field of IUIs. The next section discusses the rationale for this research by highlighting the problems currently experienced by CCs, and how IUIs could possibly address these problems.

1.2 Research Rationale

1.2.1 Contact Centres

Merchant SA's benchmarking report reveals that only 71% of customer queries are resolved on the first call by the first agent who answers the call, and that 10% of all calls answered are escalated to second- and third-level CCAs (Dimension Data 2006). The report also reveals that 17% of all customer queries are not resolved on the first call. The inability to solve these queries and the necessity to escalate the problems reside in the effectiveness of a CCA to diagnose the problem and find an adequate solution to meet a customer's needs.

A field study of three South African CCs was conducted in May 2006 to determine the cause of the problems experienced by CCAs. This study consisted of an observation of the CCAs in their working environment followed by an interview with the CCAs after a call was answered. The observation of the CCAs was done in a discrete manner in order to reduce the risk of interpreting incorrect actions and drawing incorrect conclusions as to the nature and severity of the perceived problems. A questionnaire (Appendix A) was issued to the CCAs to obtain information relating to:

- The role of the CCA at a particular CC;
- The use of information systems and how these systems influence their everyday activities;
- The types of information that the CC deals with on a daily basis;
- Problems experienced and how these are resolved; and
- What additions they would like to see in CC software that could improve CC operations.

The observations and the interviews conducted revealed the need for an integrated CC solution. Presently, CCAs find themselves switching between third-party calling software and in-house knowledge-base software. Figure 1.1 contains a model of current CC operations and illustrates that one of the factors influencing call resolution rates (CRR) is the number of applications with which a CCA needs to interact in order to complete a task.

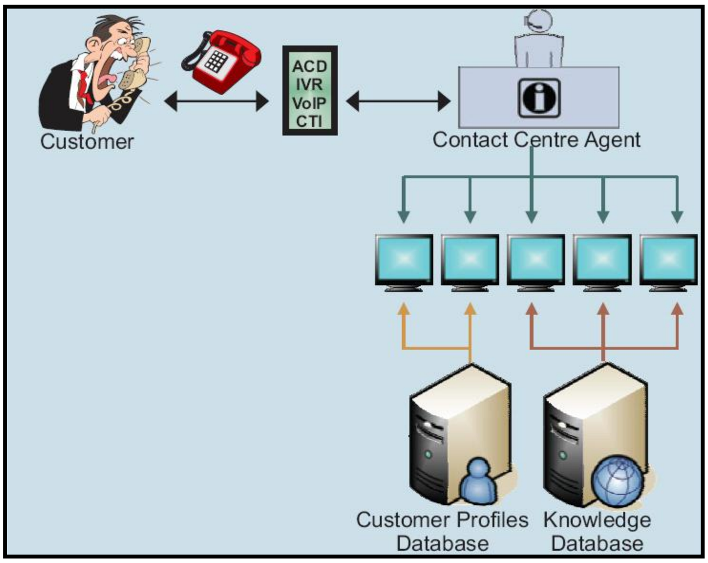


Figure 1.1: Current Model of CC Operations

Interviews conducted with the CCAs revealed the use of up to 13 different applications at one time. The purpose of these different applications is to log and verify customer details as well as to find relevant information required to solve the customer's query. On average, customer verification is done twice, using two separate systems with the same data. Knowledge retrieval is typically done in applications that are different from those containing the customer's profile. The time spent switching between these applications and the waiting period involved for verifying and retrieving the relevant information contributes to the delay and/or failure to resolve customer queries on the first call.

The above discussion highlights the need for an integrated application that allows CCAs to:

- Identify the customer;
- Describe the customer's query; and
- Obtain resolution information based on the description of the query.

Applying IUIs to the domain of CCs could improve the efficiency and effectiveness of call resolution.

1.2.2 Intelligent User Interfaces

IUIs can be considered as the next wave of interfaces (Maybury 1998). They are human-computer interfaces with the aim of enhancing the efficiency, effectiveness and naturalness of interaction between users and computers (Maybury 1999). IUIs have been designed to answer the following questions, which have emerged from the field of human-computer interaction (HCI):

- How can HCI be simplistic and at the same time, efficient and effective?
- In what way can interfaces provide a greater level of support for users' tasks, plans and goals?

In order for IUIs to address the call resolution problem currently experienced by CCs, they would need to provide an integrated interface with which CCAs can interact. Figure 1.2 illustrates that although the interaction mode between the CCA and the customer remains the same; the manner in which information is queried, validated and retrieved is done through an integrated IUI as opposed to several separate applications.

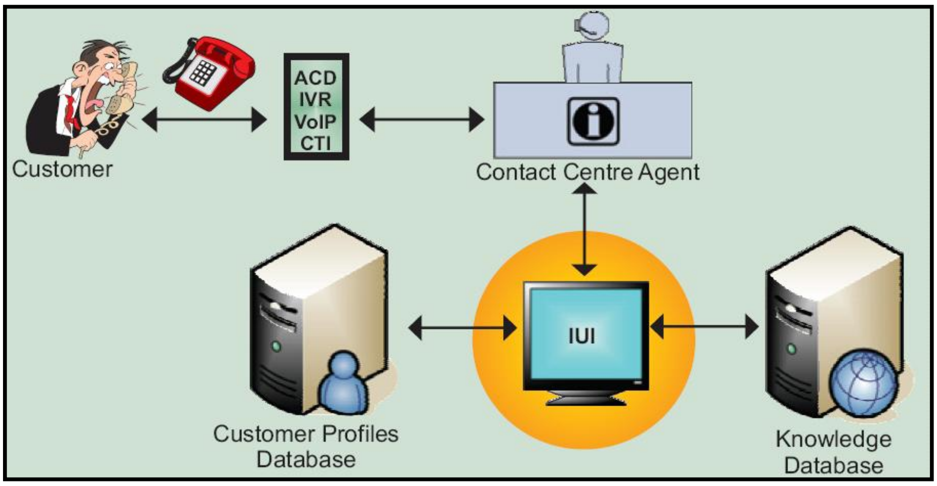


Figure 1.2: Extended Model of CC Operations Using an IUI

IUIs possess the ability to accept and generate multimodal forms of communication (Maybury 1998). This ability to process and generate multimodal input and output allows for the diagnosing of customer queries to be explained in a naturalistic manner, which is understood by the CCA, but is also comprehensible by the application.

Delivering an efficient and effective interface, for CCs, for information retrieval and presentation can be achieved through the application of IUIs. From a theoretical standpoint, it is evident that IUIs have significant benefits to offer to the domain of CC operations. The focus of this research, therefore, is to develop an IUI model to support and improve the efficiency and effectiveness of CC operations.

1.3 Research Outline

1.3.1 Problem Statement

The main objective of this research is to develop an IUI model and an IUI prototype for CCs. The IUI model would be used to improve CC operations by potentially improving CRR. The model needs to show how CRR can be improved by means of enhancing the query diagnosis and resolution process at the first-level of support.

1.3.2 Scope and Constraints

The scope of this research is limited to designing an IUI model for CCs, and specifically, to diagnosing and resolving of customer queries and generating information in order to resolve these queries.

1.3.3 Research Objectives

For the purpose of this research, the following research objectives have been identified:

- Identification of the factors influencing CRR and the current process of logging and resolving a customer's query (Chapter 2);
- Determining how IUIs could be used to address these problems (Chapter 3);
- Evaluating existing IUI models and how they could be specialised for the domain of CCs (Chapter 4);

- Specialising an existing IUI model for the domain of CCs (Chapter 5);
- Determining how the proposed IUI model could be implemented (Chapter 6) and evaluated (Chapter 7); and
- Highlighting the achievements and contributions made by this research and outlining any recommendations that could lead to future work (Chapter 8).

1.3.4 Research Questions

The main research question to be addressed is: *How can an IUI model for CCs be designed to potentially improve CRR?*

The research objectives, specified in Section 1.3.3, are broken up into research questions. These research questions are tabulated in Table 1.1 along with the methods that will be used to answer these questions. The chapter(s) in which each of these research questions will be addressed is also specified.

Research Question	Research Methods	Chapter
1. What are the factors influencing CRR? • What are CCs? • What types of CCs exist? • Are there any specific structures for each type of CC? • What are the advantages and disadvantages of these structures? • What software is employed within CCs? • Does current CC software contain IUIs and are they extensible and flexible? • What challenges do CC operations face? • What is the current process when logging and resolving a customer's query?	*Literature Study, Field Study*	2
2. How can an IUI be utilised to address the problems experienced by CCs?	*Literature Study*	3

• What is an IUI? • How does an IUI differ from a traditional user interface? • Are there different types of IUIs? • What are the components of an IUI? • Which of these components could be used to address the problems experienced by CCs?		
3. Are there any existing IUI models that could be specialised for the domain of CCs? • What is a model? • What are the characteristics of a model? • Are there different types of models? • Of what elements should a model comprise? • What are the different stages of model-based design? • Which is the most generalised IUI model that could be specialised for CCs?	*Literature Study*	4
4. How can the selected IUI model be specialised for CCs? • What are the current limitations of the selected model and how can these limitations be addressed? • What are the elements of the proposed IUI model? • Which elements of the proposed IUI model need to be specialised?	*Iterative Design*	5
5. How should the proposed model be implemented? • Which elements of the model should be implemented? • How should these elements be implemented?	*Implementation* *Evaluation*	6
6. How useful is the proposed IUI model? • What type of evaluation should be performed? • What metrics and measurements should be taken into consideration?	*Model Evaluation, User Testing, Evaluation*	7

• What evaluation instruments should be used? • Who should be selected to perform the evaluation? • What are the results of the evaluation? • How do these results impact on the proposed IUI model and the IUI prototype?		
7. What recommendation and conclusions can be made? • What are the objectives of the research? • Do the outcomes of this research meet the initial research objectives? • What achievements and contributions has this research made to theory and practice? • What recommendations can be made for theory, practice and future work?	*Analysis*	8

Table 1.1: Research Questions and Methodology

1.3.5 Research Design

This research will focus on the development of an IUI model for CC operations. The primary method is the design of a model and this will be supported by several secondary methods (Olivier 2006). These secondary methods include a literature survey, field study, prototype construction and an evaluation of the prototype (Olivier 2006). Instruments that used to aid this research are questionnaires and a test plan.

1.3.5.1 Methods

A literature survey of CCs, IUIs and model-based design of IUIs will be performed. The purpose of this literature survey is to gain an in-depth understanding of the field of CCs, IUIs and model-based design of IUIs. A field study will be conducted to identify the problems currently influencing CRR.

The design of an IUI model for CCs will be based on the knowledge acquired during the literature survey for IUIs and model-based design of IUIs. Evaluation of the model will be done through the implementation of a prototype (Olivier 2006).

This prototype will be further evaluated by means of a model evaluation and user testing. The aim of these evaluations is to determine the extent to which the model supports CC operations by being able to potentially reduce CRR; and whether the prototype meets it objectives of being able to log and resolve customer queries.

1.4 Dissertation Structure

This dissertation consists of eight chapters. Each chapter attempts to meet a research objective by answering the research questions in Table 1.1. The methods applied in this research will assist in determining the structure of this dissertation (Figure 1.3).

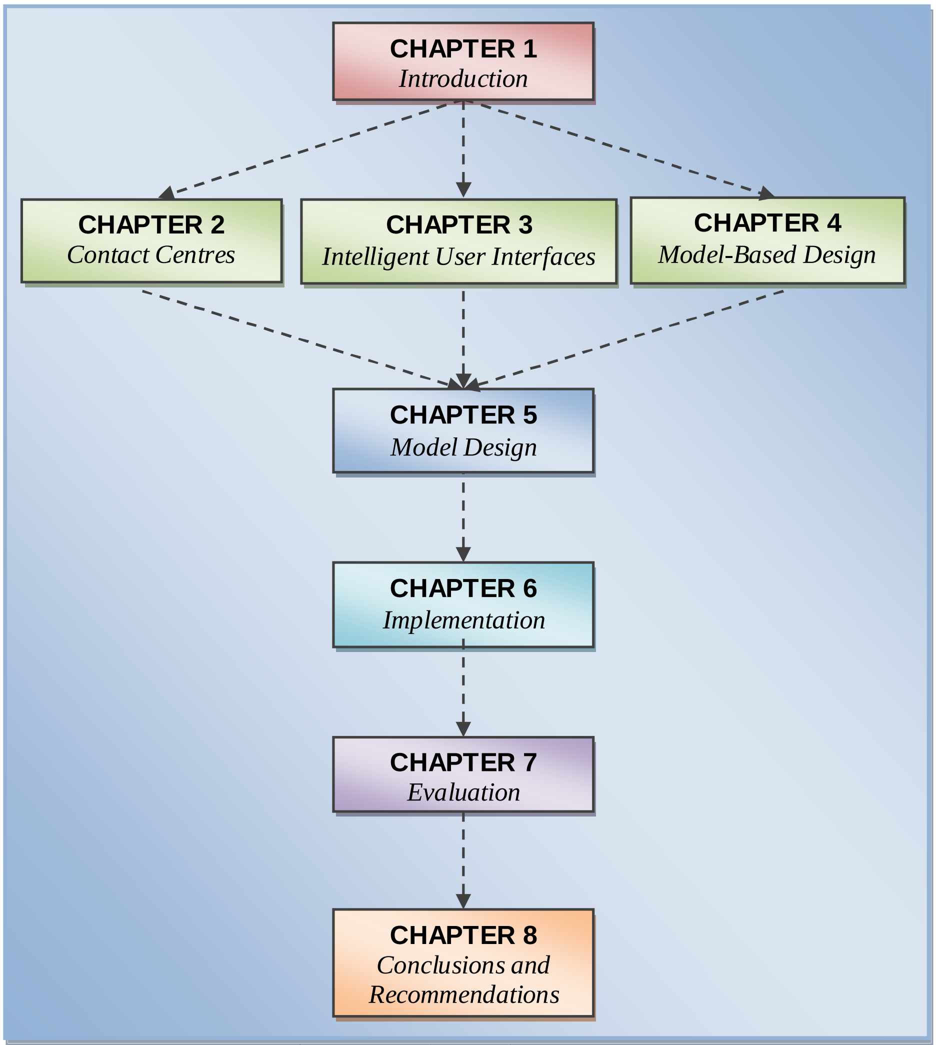


Figure 1.3: Dissertation Structure

Chapter 1 (Introduction) presents a short discussion on the domain of CCs and the field of IUIs. The preliminary literature study highlights some of the concerns and problems that are currently experienced by CCs (Section 1.2.1). This is followed by a short discussion on IUIs (Section 1.2.2) showing that their application to the domain of CCs could, theoretically, present some benefits. The scope and constraints of this research, are presented in Section 1.3.2. Research objectives are presented in Section 1.3.3, with the design that this research will follow presented in Section 1.3.5.

Chapter 2 (Contact Centres) comprises the first literature study. It presents a detailed literature study into the domain of CCs. The chapter focus on identifying the types of CCs and their various structures. Existing CC software is analysed in order to identify the extent to which intelligence is currently supported. The challenges facing CCs are discussed by highlighting the problems identified by both the literature and a field study. The process of logging and resolving a query (identified through the field study) is also discussed.

Chapter 3 (Intelligent User Interfaces) presents a detailed literature study into the field of IUIs. This chapter aims to define what IUIs are, how they differ from traditional interfaces and identifies the various types of IUIs that exist. A discussion on the various IUI components is presented in order to determine which of them is most suitable to address the problems experienced by CCs.

Chapter 4 (Model-Based Design of IUIs) focuses on model design theory. This chapter discusses the theory of model-based design and evaluates of several existing IUI models. The goal of this evaluation is to identify the most generalised IUI model that could be specialised for the domain of CCs.

*Chapter 5 (*Model Design) focuses on the specialisation of the model, selected in Chapter 4, as a solution to the research problem. The proposed IUI model is discussed in terms of its architecture, component-level design and interface design.

Chapter 6 (Implementation) discusses the implementation of an IUI prototype based on the IUI model proposed in Chapter 5. A scenario is used to demonstrate how the IUI prototype meets its functional objectives. The implementation of the proposed IUI model serves as a proof of concept that the model can be used to develop IUIs for CC applications.

Chapter 7 (Evaluation) discusses the evaluation of the proposed IUI model and the IUI prototype in order to determine its usefulness. The evaluation strategy is discussed and a presentation of the results follows.

Chapter 8 (Conclusions and Recommendations) presents the conclusions of this dissertation. It examines whether the research objectives set out in this chapter have been met. A review of the research conducted and presented in this dissertation is also included. Achievements, contributions made and recommendations for future work are outlined in this chapter which are followed by a list of references and several appendices.

Chapter 2: Contact Centres

2.1 Introduction

The aim of this chapter is to provide a detailed discussion of the domain of CCs. This is necessary in order to understand what CCs are, how they are structured, the software they use and the different types of challenges that impact on the operations of a CC. A discussion on existing CC software is presented to determine the extent to which intelligence is present within the interface as well as a discussion on the various challenges impacting on CCs to determine what factors impact on CRR. Lastly, the results of the field study are discussed. These results aim to identify the current process used when solving a customer's query and how this process can be improved through the use of an IUI.

2.2 Overview

CCs are not restricted to a particular business domain and can be implemented by any corporation which is willing to provide additional support and marketing services to its customers. The benefits provided by CCs are the increased accessibility of the company to the customer and the ability to complete queries and transactions in a more efficient and focused manner (Australian National Audit Office 1996).

Despite these benefits, CCs were initially dismissed by the business fraternity due to their high operational costs and labour intensive nature (Almskog and Frydman Communications 1996). Telecommunication companies have now realised that excellent customer service not only serves as a campaign to attract new customers but also retains existing customers (Kaish 2004).

A CC employs staff known as CCAs, whose responsibility it is to handle calls placed or received (Contact Centre University 2000). CCAs can also be referred to as customer service representatives (CSRs) and act as the bridge between the customer and the organisation by resolving customer queries (ECA 2006).

2.3 Types of Contact Centres

Classification of CCs is done in terms of their functionality and areas of operation. There are two main types of CCs, those being inbound and outbound. This classification can be broken down further, based on the area of operations, as either being a help desk or a service desk. Figure 2.1 shows the hierarchy of CCs and their various types. It further illustrates that CCs are a type of call centre and, therefore, still maintain the characteristics of either being inbound or outbound.

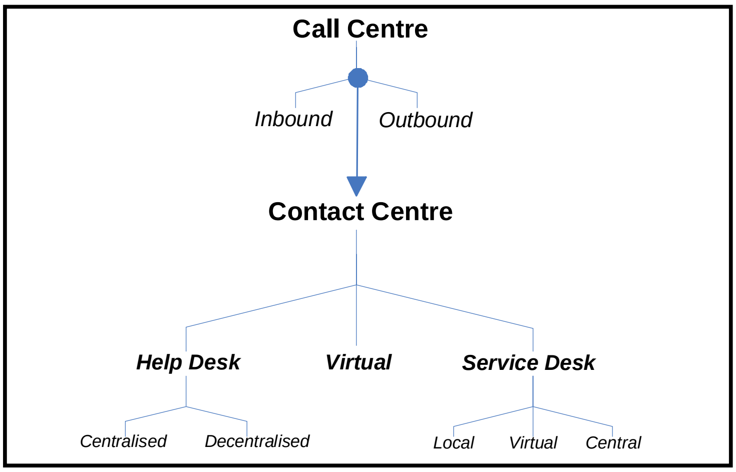


Figure 2.1: Contact Centre Types and Structures

2.3.1 Inbound Contact Centres

Inbound CCs are responsible for receiving calls in order to provide customer service and support (Robertson 2005; Gans *et al*. 2003; ECA 2006). Inbound CCs can further be differentiated as either being transactional or query-based (Robertson 2005). A query-based inbound CC focuses on handling queries from customers and relies on a knowledge base in order to resolve these queries (Robertson 2005). In contrast, a transactional inbound CC primarily deals with the processing of transactions for customers. In order to perform these processes, transactional CCs rely on a set of procedures that need to be followed in order for a transaction to be completed correctly and efficiently (Robertson 2005). Query and transactional-based CCs do not have to be mutually exclusive, as both of these types of inbound CCs can be integrated and require complex hardware and software (Robertson 2005).

2.3.2 Outbound Contact Centres

Outbound CCs are responsible for the placing of calls from within the CC (Robertson 2005; Gans *et al.* 2003; ECA 2006). The purpose of placing a call is to sell a product or service (Robertson 2005). A recent trend within the CC industry is to generate an outbound call to highly valued customers who have abandoned their inbound call before being served by a CCA (Gans *et al.* 2003).

2.3.3 Help Desks

Help desks can be defined as a single point of contact which deals with the resolution of customer queries and requests (Frantz 2006; Davis and Maxwell 2004). The support provided by help desks either deals with users who are having trouble remembering how to accomplish a task or users who are experiencing problems with the information technology (IT) system and require support to solve this problem (ACM Diagnostics 2005; Contact Centre University 2000).

2.3.3.1 Types of Help Desks

A help desk can either be centralised or decentralised. The role of a centralised help desk is to act as a single location within an organisation to provide support to all its users (Sanderson 2003). Centralised help desks are often located within the IT department and assist with software or hardware support (Sanderson 2003). Decentralised help desks are distributed across multiple sites located throughout the organisation. Maintaining consistent information between decentralised help desks is one of the major concerns. Performance benchmarking is also difficult if common methods and techniques for capturing information are not used (Sanderson 2003).

An advantage of a centralised helpdesk is that the severity of a problem (indicated by the number of affected users) is immediately known (Sanderson 2003). A disadvantage of a centralised help desk is the fact that a single staff member may not always understand the needs of each business unit within an organisation and that an offsite division may require support which cannot be supplied due to its remote location (Sanderson 2003).

Decentralised help desks solve the problem that centralised help desks have with attending to offsite queries and provide support to a variety of departments within a company or organisation (Sanderson 2003).

Table 2.1 contains a summary of the advantages and disadvantages of centralised and decentralised help desks.

2.3.4 Services Desks

Service desks, according to the IT Infrastructure Library (ITIL), are defined as the central hubs of contact between the customer and all IT-related areas, more specifically IT services (ITIL 2006). This definition is supported by Microsoft (2002) as service desks are defined as a single point of contact for customers and technicians. Service desks aim to provide communication, information and resources to customers with IT infrastructure problems (Microsoft 2002).

Benefits of a service desk (ITIL 2006) include:

- Improved knowledge retrieval due to single point contact;
- IT infrastructure components are controlled and managed appropriately;
- Identification of customer training requirements for better utilisation of services;
- Improved sense of collaboration within the service organisation;
- Increased customer satisfaction; and
- Reduction in service costs.

.

	Advantages	**Disadvantages**
Centralised	• Easily located by users. • Better communication among specialists. • Easier to enforce standards. • Better use of resources. • Exposure of specialists to a broad range of issues.	• Inability to provide on-site support to remote locations. • Difficulty in understanding business needs. • Challenge of providing support in multiple time zones.
Decentralised	• Ability to address local and remote site needs. • Availability of on-site services.	• Difficulty in providing standardised information. • Need for common mission and goals. • Challenge of measuring performance.

Table 2.1: Advantages and Disadvantages of Help Desk Structures (Sanderson 2003)

The service desk should be seen as a function and not a process of an organisation (ITIL 2006). The primary role of the service desk is to coordinate information and act as the central facilitator between an organisation's customers, the internal service organisations and the external providers for the organisation (ITIL 2006). Figure 2.2 provides an overview of how ITIL describes a proactive service desk.

The distinguishing factor between help desks and service desks is the multiple channels which can be used by service desk clients to report problems and receive assistance (Frantz 2006).

2.3.4.1 Types of Service Desks

There are three types of service desk structures: namely, a local structure, a centralised structure and a virtual structure.

The local service desk structure presents a service desk that is localised to a department or business unit (ITIL 2006). Knowledge used by the service desk staff is very specialised in order to deal with IT-related queries that are specific to a particular department or business unit. This, however, could become costly for an organisation that houses multiple departments under one roof.

A centralised service desk means that a single centralised service desk is responsible for the support provided to an entire organisation (ITIL 2006; Microsoft 2002). Centralised structures also imply a shared set of resources by all the departments within an organisation. Time allocation and sharing of technicians could then become problematic without proper management.

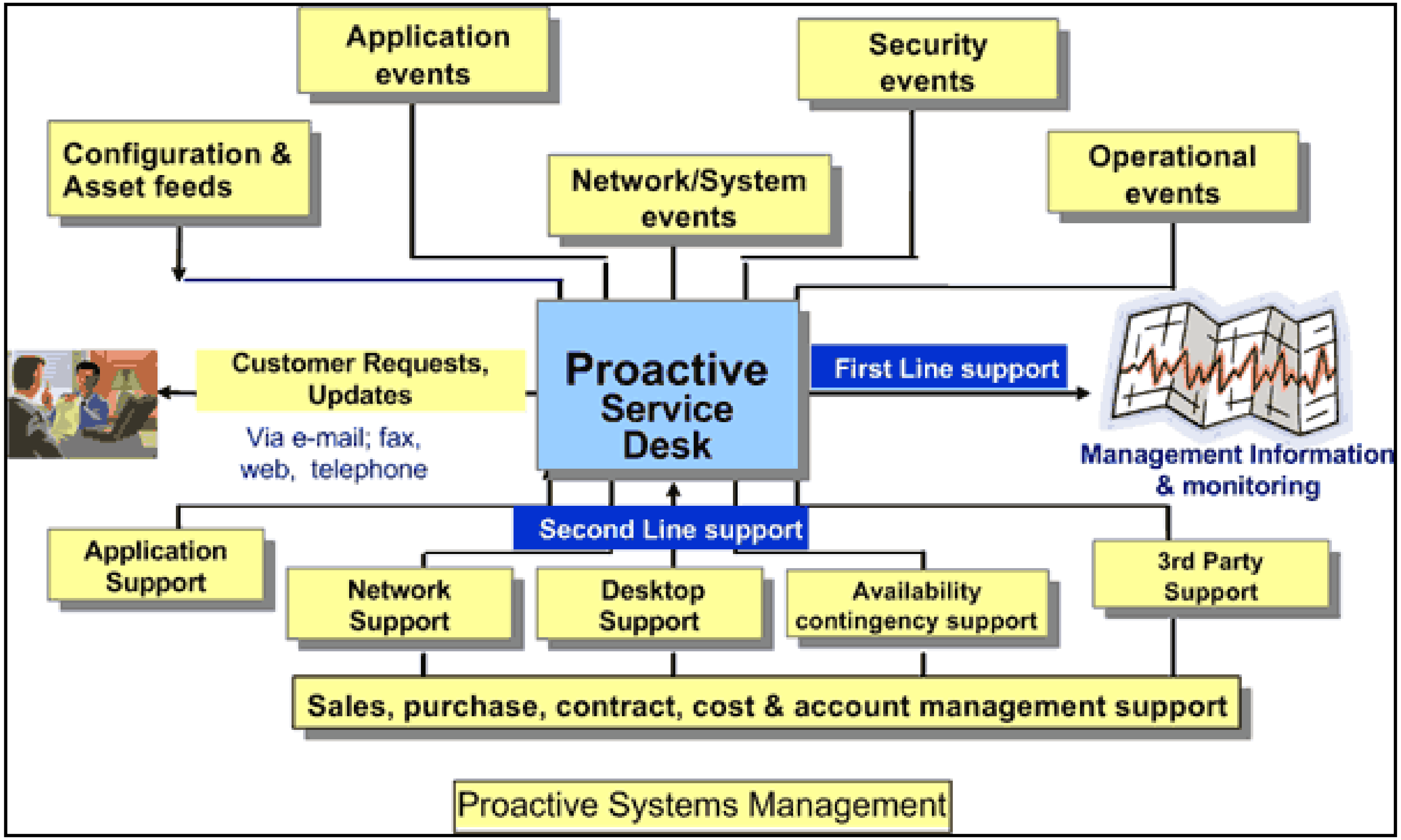


Figure 2.2: Proactive Service Desk (ITIL 2006)

A virtual service desk is a combination of both a local and a centralised service desk. Virtual service desks allow for information to be centrally stored but globally accessible. This type of structure provides support for a 24-hour work day, whereby a "follow the sun" approach is employed (Microsoft 2002).

Table 2.2 presents the advantages and disadvantages of these service desk structures (ITIL 2006; Microsoft 2002).

2.4 Existing Software

Software is used by CCs as a means to:

- Retrieve customer information through the use of interactive voice response (IVR);
- Control the routing of calls through an automatic call distributor (ACD);
- Integrate operations performed on the computer and calls received on the telephone through computer-telephony integration (CTI);
- Provide customer profiles (customer-relationship management); and
- Support the retrieval and storage of information in a knowledge base.

Usually ACD, IVR and CTI are combined into a single application when sold. Customer-relationship management (CRM) software and knowledge-base software are often sold independently of each other.

A problem occurs when these applications are sold individually. This can result in CCAs working in multiple applications in order to identify the customer and to resolve the customer's query. On average, CCAs are presented with seven different systems on their desktop, and this number can escalate to as high as 15 (Kaish 2004). This situation causes an increase in cost and time factors.

	Advantage	**Disadvantage**
Local	• Support provided is focused on IT infrastructure relating to a specific department.	• Multiple service desks for multiple departments within a single organisation can be costly and difficult to manage.
Centralised	• Users have a single number to call. • Fewer staff required which implies reduced training times and facilitates lower costs.	• The knowledge of the centralised service desk staff is comprehensive but not as specific and detailed as the knowledge of local service desk staff. • Shared technical resources, especially support staff, can be a time-critical factor that requires constant management.
Virtual	• Uses a "follow the sun" approach. • Support can be provided 24 hours a day. • Combines advantages of both local and central service desks.	• Accessing of information can be slow depending on geographical location, network traffic, line speed and various other physical and network factors.

Table 2.2: Advantages and Disadvantages of Service Desk Structures

Recent CC software allows for the integration of CC telephony technology and CRM. The more expensive solutions offer integrated knowledge-base services, CRM capabilities and CC call-management technology.

Hardware that supports ACD, IVR and Voice over Internet Protocol (VoIP) is not essential for a CC to operate but can be implemented to provide increased levels of efficiency in terms of service delivery. The traditional setup for a CC workstation is a telephone to place and receive calls, as well as a computer to log the customer's request and possibly resolve the problem. CC applications provide these facilities through a distributed range of applications. A traditional CC application would have all of these features integrated as a complete solution but at an escalated price.

2.4.1 Axio Systems–Assyst

Axio Systems service desk application is called Assyst (Axio Systems 2006). Assyst presents the following features and benefits (Axio Systems 2006):

- Original solution to adopt and present ITIL framework;
- Provides web-based self-service;
- Supports handheld devices;
- Extensible and configurable;
- Comprehensive knowledge base; and
- Automatic linking and associations to incidents, problems, errors, tasks and changes.

No support for intelligence within the interface is provided. Figure 2.3 is an example of the Assyst interface on a mobile device.

The mobile module allows for updating of business-critical services via a mobile device (Axio Systems 2006).

Figure 2.3: Assyst Mobile Interface (Axio Systems 2006)

2.4.2 LiveTime Help Desk

LiveTime Help Desk is a help desk or service desk application that supports all major operating systems, databases and Internet browsers (LiveTime 2006). This application boasts an IUI with the benefit of being able to resolve a call within 10 seconds and the fact that no training or a manual is required in order to use the product (LiveTime 2006). LiveTime Help Desk solutions can run on any Java 2 Enterprise Edition (J2EE) server. It is also the industry's first J2EE customer service and support solution. LiveTime Help Desk is fully customisable, ITIL compliant and allows for web service integration (LiveTime 2006).

2.4.3 SiteHelpDesk-IT

SiteHelpDesk-IT is a help desk application that is compliant with all ITIL standards and specifications (SiteHelpDesk.com 2006). It is web-based and requires no client installations. Accessiblity from handheld devices is supported. SiteHelpDesk-IT is built on Active Server Pages (ASP) technologies by Microsoft. One of the unique features about this application, in terms of extensibility and customisation, is the fact that the source code for the application is included in the purchase price (SiteHelpDesk.com 2006). Figure 2.4 is an illustration of the web-based SiteHelpDesk-IT call management screen. SiteHelpDesk-IT does not possess an IUI nor does it support the addition of one.

2.4.4 FrontRange Solutions-Heat

The Nelson Mandela Metropolitan University (NMMU) Information and Communications Technology (ICT) Service Desk uses the HEAT Product Suite as its service desk application. Developed by FrontRange Solutions, HEAT provides support for (FrontRange Solutions 2006):

- Automatic ticket generation;
- Remote and secure access to clients' desktops to provide remote assistance;
- Single source of administration;
- ITIL standard-based modules;
- Self-service facility for clients to solve smaller problems before they contact the service desk; and
- Knowledge support to provide accessibility to answers or solutions on demand.

Figure 2.5 is an illustration of the HEAT call log screen. HEAT is not web based and presents users with a traditional application interface.

.

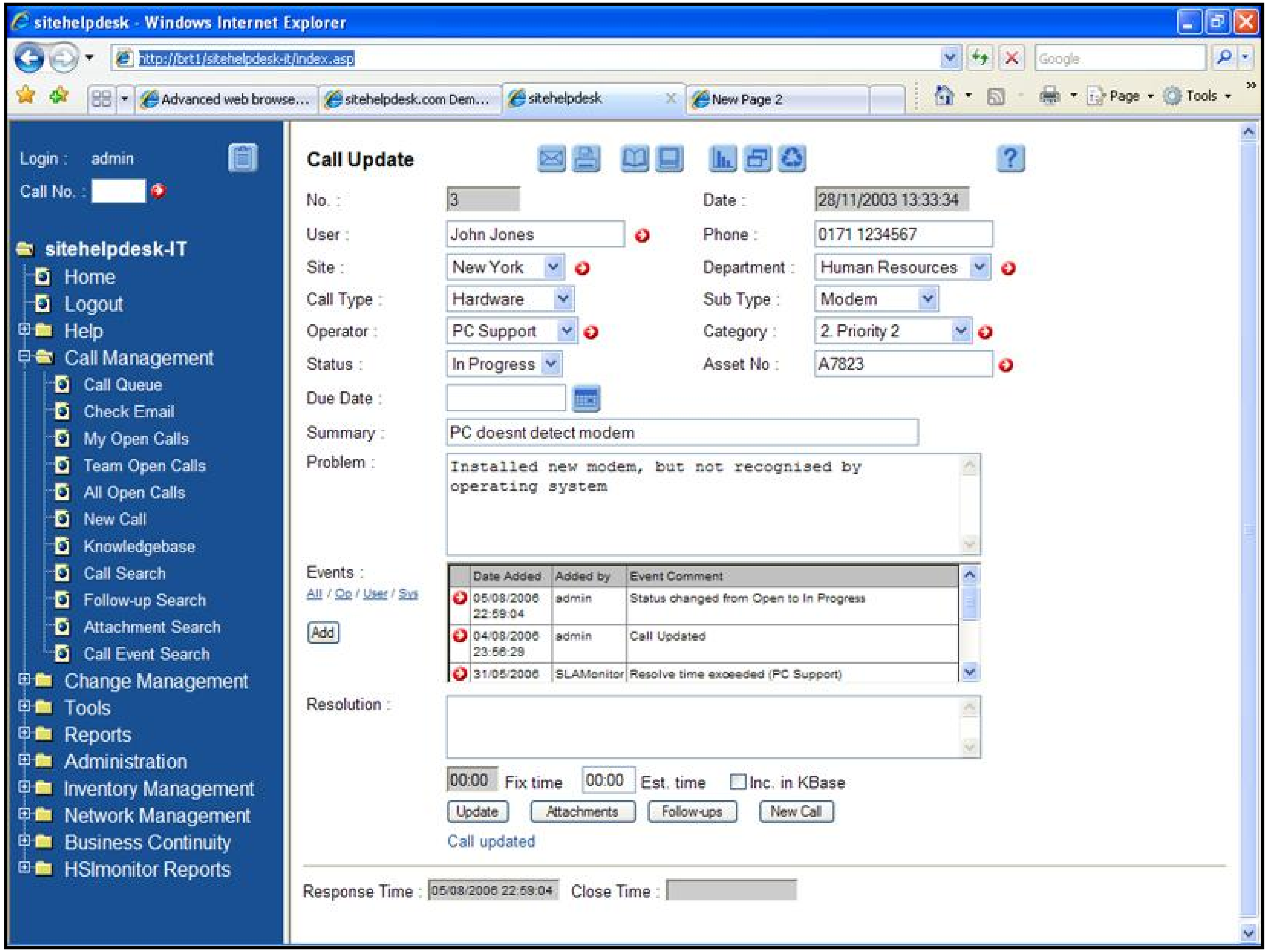


Figure 2.4: Call Management Screen from SiteHelpDesk-IT (SiteHelpDesk.com 2006)

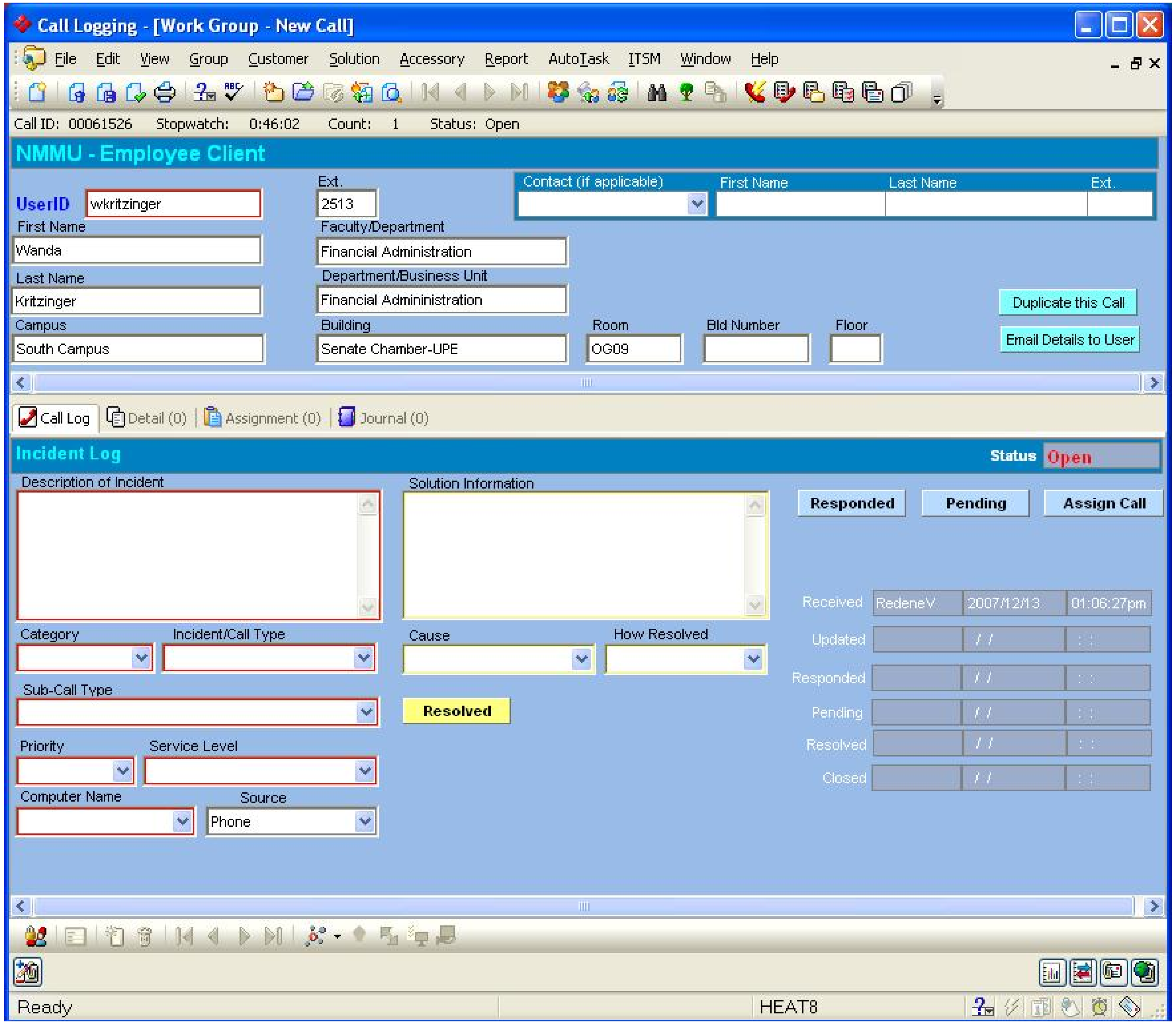


Figure 2.5: HEAT Call Log Screen

2.4.5 Serio Service Desk

Serio Service Desk is a modular service desk application developed by Serio and is based on ITIL's standards and specifications (Serio 2006). It boasts the ability to reduce costs and improve operational reliability. Serio Service Desk presents the following features (Serio 2006):

- Incident, problem and change management;
- Knowledge-base facilities;
- Mobile support;
- Short Messaging Service (SMS) support;
- Integration with Active Directory;
- Embedded workflow information; and
- Service Level Agreements (SLA) and escalation management.

Figure 2.6 shows how all of these features co-operate with each other in order to form a comprehensive and versatile application.

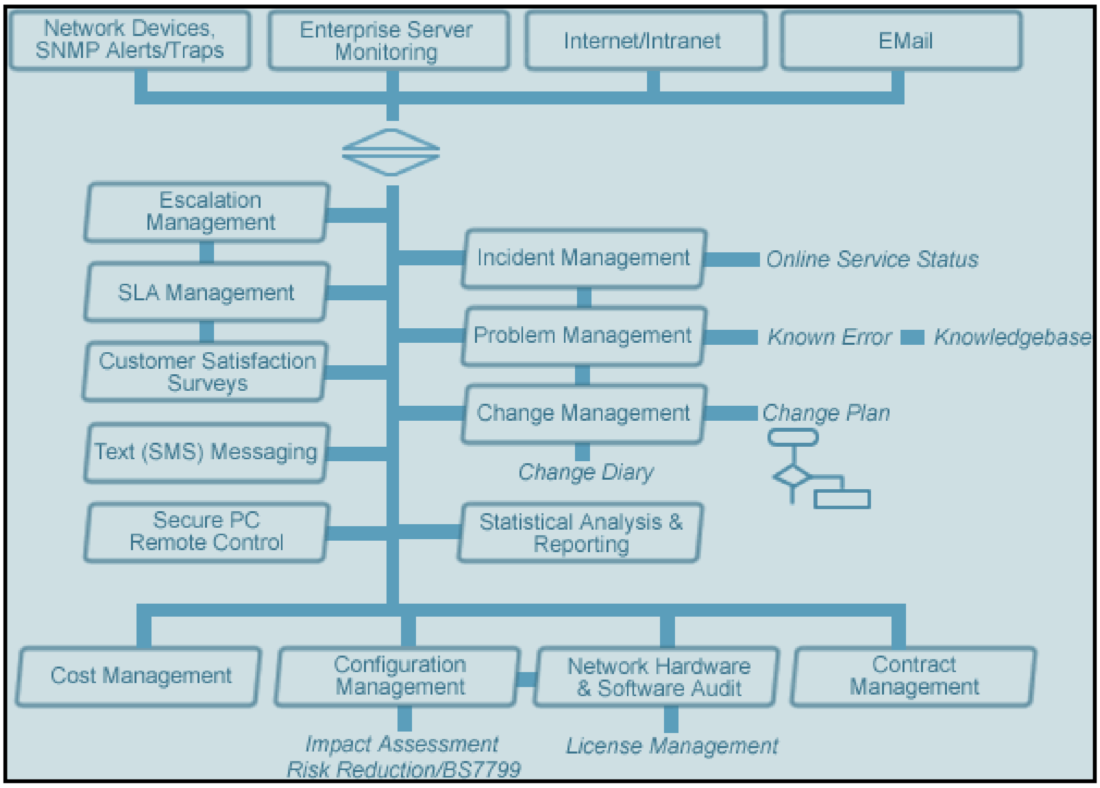


Figure 2.6: Serio Service Desk Feature Overview (Serio 2006)

2.4.6 Polar Help Desk

Polar Help Desk is a help desk or service desk application engineered and sold by Polar Software (Polar Software 2006). Polar Help Desk comes in standard and professional versions. It supports ITIL standards and specifications for a service desk and is built using ASP. The professional version allows for seamless integration with Microsoft Windows Server Active Directory and provides various service level management (SLM) functions (Polar Software 2006). Polar also provides the source code, but at a price of $20 000. Figure 2.7 is an illustration of the web-based interface for Polar's help desk application. The illustration shown is the incident tracking screen, which is used to track and monitor all incidents currently active or resolved (Polar Software 2006). Polar Help Desk does not possess an IUI.

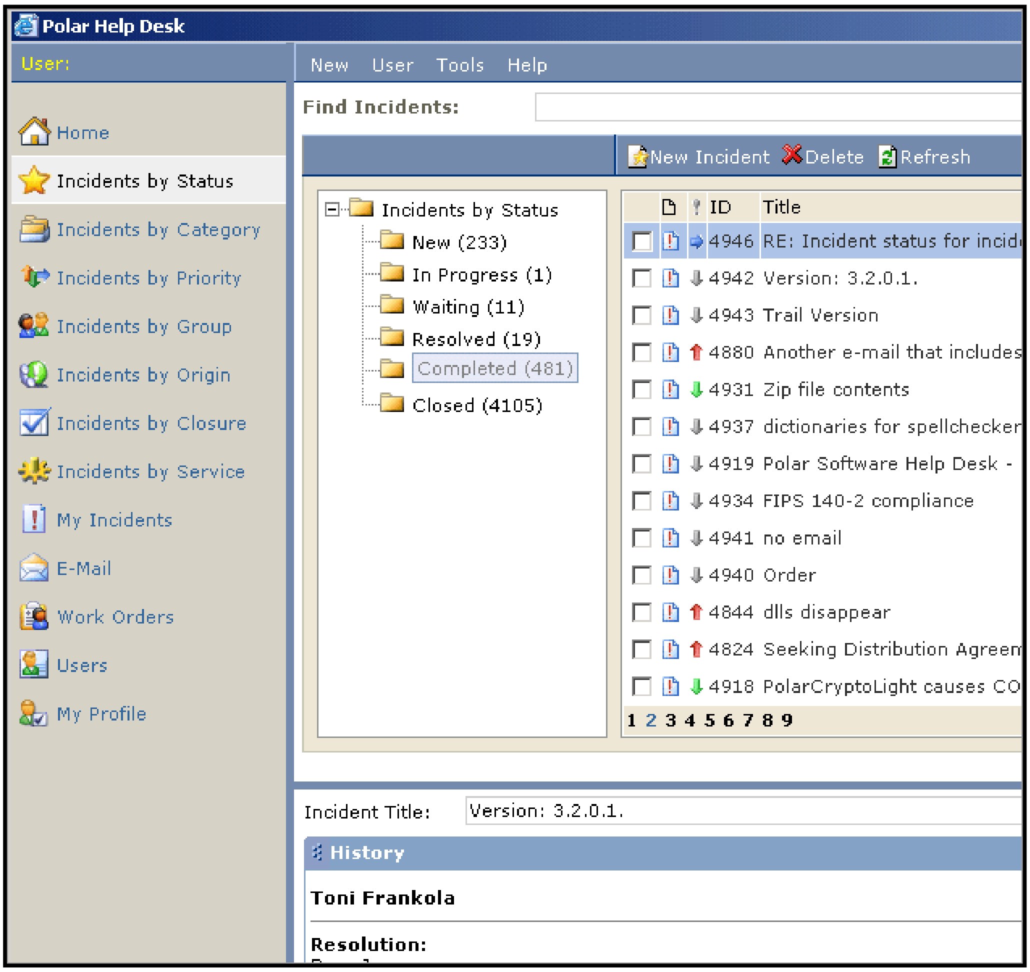


Figure 2.7: Polar Help Desk Track Incidents Screen (Polar Software 2006)

2.4.7 Software Comparison

Sections 2.4.1 to 2.4.6 presented the features and benefits of various help desk and service desk applications. This section presents a tabulated (Table 2.3) comparison of these products in regards to the following criteria:

- Support for ITIL;
- Web-based interface;
- Support for mobile devices;
- Extensibility and customisability;
- Availability of source code; and
- Inclusion of an IUI.

The criteria presented were established by identifying the common features of the various products.

	Product					
	ASSYST	*LiveTime Help Desk*	*SiteHelpDesk–IT*	*Polar Help Desk*	*Serio Service Desk*	*HEAT*
ITIL	✓	✓	✓	✓	✓	✓
Web based	✓	✓	✓	✓	✗	✗
Mobile Support	✓	✗	✓	✗	✓	✗
Extensible and Customisable	✓	✓	✓	✗	✗	✓
Source Code Availability	✗	✗	✓	$20000	✗	✗
Intelligent Interface	✗	✓	✗	✗	✗	✗

Table 2.3: Service Desk Software Comparison

Only LiveTime's solution possesses an IUI. This application is extensible but the source code is not provided; therefore, a need exists to develop an IUI model and prototype for CC operations, to provide intelligence and extensibility.

2.5 Challenges

The need to capture and retain customer loyalty to an organisation's brand requires that CCs be used as a tool to facilitate the building and maintenance of the relationship between an organisation and its customers (Siebel Systems 2005). A successful CC is one that is able to achieve an elevated sense of loyalty and satisfaction from its customers and employees whilst keeping the operational cost optimal. Figure 2.8 illustrates the triangular nature of this situation. This phenomenon is described as the balancing act which needs to be done by CC management (Mehrotra and Fama 2003). The balancing of these three factors is critical because they compete for interest within the CC environment (Mehrotra and Fama 2003).

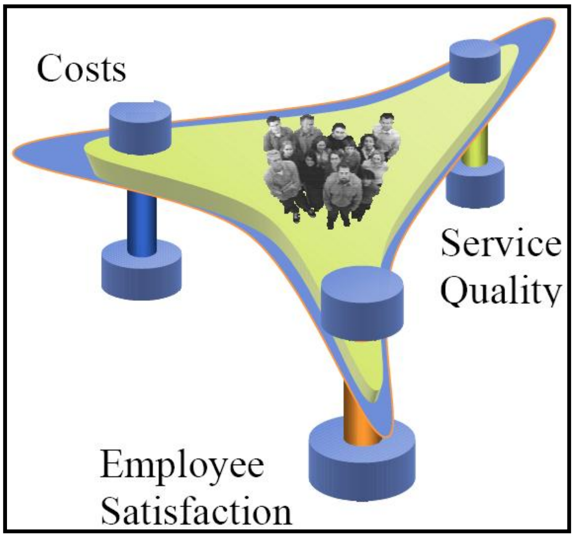


Figure 2.8: CC Management Balancing Act (Mehrota and Fama 2003)

Research shows that there is currently a high turnover rate for CCAs, organisations are struggling to provide customer satisfaction at an optimal-level and are constantly presented with inflated costs for both labour and technology (Fusion 2004).

The next three sections take a look at the factors involved in obtaining the optimal levels in terms of customer satisfaction (Section 2.5.1), operational costs (Section 2.5.2) and employee satisfaction (Section 2.5.3). Identification of these factors could present possible insight into the requirements of maintaining a balance between customer satisfaction, operational costs and employee satisfaction.

2.5.1 Customer Satisfaction

Customer satisfaction awareness has increased in the business world due to the variety of alternatives available, together with low switching costs (Siebel Systems 2005). The need for improved customer satisfaction could be the major differentiator in a customer selecting a particular brand over another (Siebel Systems 2005).

Despite efforts to increase customer satisfaction and produce a dedicated sense of brand loyalty, organisations need to reduce the cost at which services are provided. This is a major issue as organisations struggle to find and implement programmes that deliver an excellent customer experience whilst being cost-effective (Siebel Systems 2005). In order to realise this goal, organisations need to identify the following (Siebel Systems 2005):

- An understanding of customer needs and behaviour;
- Satisfaction drivers for customers;
- The most cost-effective solution; and
- Customers should have access to the most appropriate and trained CCAs to assist them with their calls.

In order to achieve the greatest sense of satisfaction, customers have certain expectations that need to be met by the organisation when they place a call. These expectations include (Siebel Systems 2005):

- Customer contact history;
- Anticipation of customer needs;
- Handling of customer queries effectively and efficiently; and
- The ability to perform all of the above in a single call.

Delivering excellent service and support to customers requires that CCs define what excellent service means from the perspective of the customer (eGain 2005a). Customers who had problems with a product or service and had the problem resolved by the organisation showed greater loyalty to that organisation than customers who did not experience any problems (eGain 2005a).

The Internet has revolutionised the way in which customers buy products and services. Consumers of today are more informed, and this has resulted in customers being more aware of the choices available to them and less tolerant towards the offering of poor services by organisations. Customers require multiple means of (eGain 2005b):

- Accessing and finding information;
- Buying products and services; and
- Getting support for products and services purchased.

Interaction channels made available to customers should be accessible 24 hours a day, 7 days a week and should provide customers with quick, consistent and accurate service. Service desks support this by providing increased access to customers through the use of multiple channels to receive and respond to customer queries (Section 2.3.4).

2.5.2 Cost Reductions

Reducing service costs without compromising the level of service is a concept that is considered by most CCs. There are various ways in which CCs can reduce service costs, such as (Siebel Systems 2005):

- Avoiding frequent dispatch of costly field service staff; and
- Introducing technologies such as web, web chat, e-mail, IVR, ACD and CTI.

Workforce costs are the largest expense in a CC and consist of recruiting, training, compensation and benefits (Siebel Systems 2005). Due to this phenomenon, many organisations have outsourced their CCs or relocated them to an offshore location in order to take advantage of the cheap labour costs (Siebel Systems 2005).

2.5.3 Employee Satisfaction

Employee satisfaction is a crucial factor as the employees of the CC are the ones with direct contact with an organisation's customers. The level of service provided by the employees can determine the retention or loss of customers.

Only employees with the necessary technical and communication skills as well as the ability to cope in stressful situations with agitated customers should be hired (Gliedman 2005). Training employees effectively, in terms of using the company's available technologies, will cause employees to value their jobs more and result in employees delivering a service that is more effective, efficient and informed (Gliedman 2005).

The ability to provide an efficient service is also a direct contributor to employee satisfaction. The greatest source of inefficiency in CCs resides in the acquisition of complete customer information and the actions performed on customer requests (Anton 2004). Field studies and research have revealed that CCAs use up to 13 different applications in order to access this information. Organisations require CC solutions that are integrated in terms of the addition, maintenance and finding of customer information and knowledge in order to resolve customer queries.

2.5.4 Information Challenges

Delivering an excellent customer service at reduced costs is dependent on the productivity of the CCAs and the effective and efficient management of information (InStranet, Inc. 2003).

CC managers face the constant crises of needing to do more with less and in terms of information (InStranet, Inc. 2003). The ability to optimise CC operations through the use of knowledge-bases has emerged as a result of this concern. Currently, CC management is frequently faced with the following questions in terms of information processing (InStranet, Inc. 2003):

- How can training costs be reduced and effectiveness measured?

- How can first call resolutions (FCR) be improved using existing product information?
- How can the speed in which CCAs find information be improved?
- How can contextual information be provided to CCAs during a live call and what is the best form in which to display it?
- How can consistent processes be implemented and followed by all CCAs?

Providing solutions to the above problems will not only increase general customer satisfaction but also reduce the length of calls, increase the rate at which information is retrieved and stored, as well as provide a consistent problem-solving process used by all CCAs in terms of workflow and process flow models/diagrams.

2.6 Call Resolution Process

The process of logging and resolving a customer's query was identified through the field study (Section 1.2.1). The tasks that are typically performed when logging and resolving a customer's query are to:

- Identify the customer;
- Categorise and classify the customer's query; and
- Resolve the query.

The role played by the CCA in Figure 2.9 is that of a call taker. The tasks that are performed by CCAs typically comprise logging a call placed by a customer and processing the details of that call. In the task of processing a call, the CCA has to identify the customer and capture the problem as described by the customer. Currently, the task of resolving a query is handled either by a technician or an outsourcer. This process is time-consuming and costly.

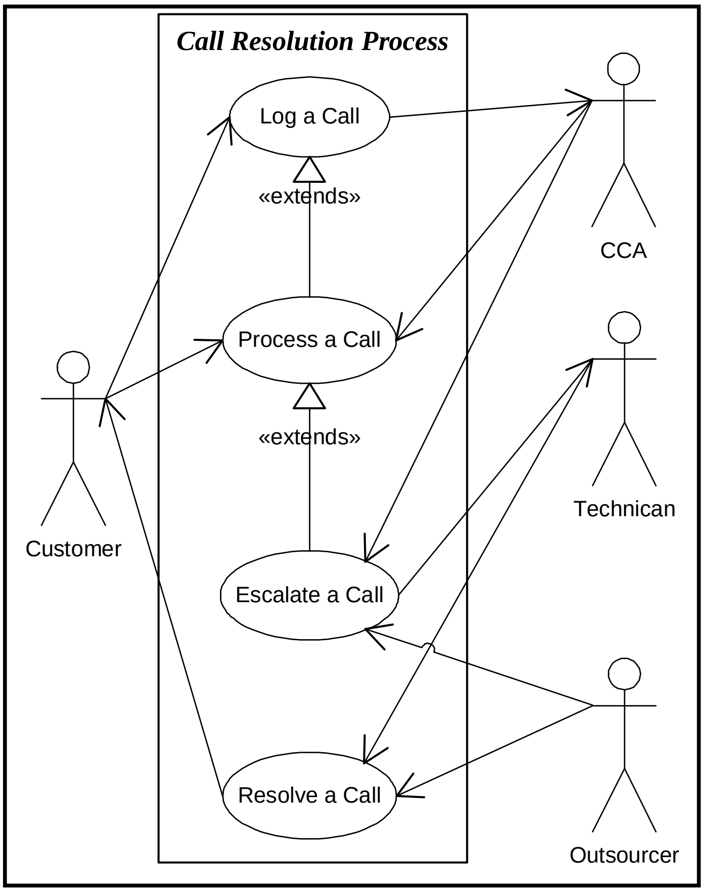


Figure 2.9: Current Call Resolution Process

Reducing the time taken to resolve a customer's query requires that resolution of the query be done at the first-level of contact. This not only offers query resolution capabilities at the first-level but also reduces and, possibly, eliminates the need to escalate the query to a technician or outsourced company. Figure 1.2 showed that an IUI could be used to reduce the number of applications with which a CCA has to interact. Figure 2.10 shows how the inclusion of an IUI could be used to improve the call resolution process within a CC.

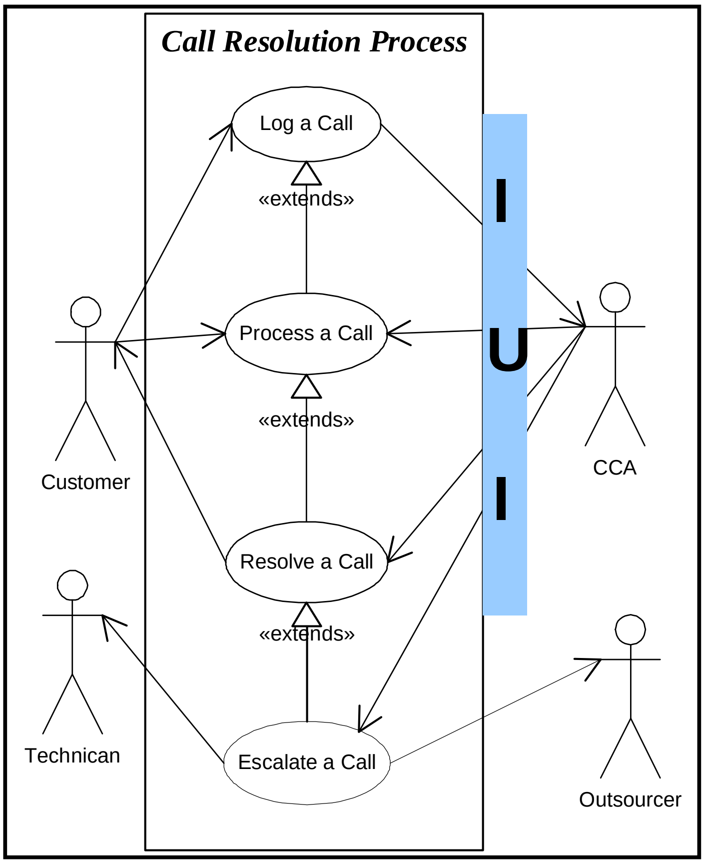


Figure 2.10: Call Resolution Process using an IUI

Use of an IUI would allow a CCA, at the first-level of support, to be both a call taker and a problem solver (Figure 2.10). Escalation of a query would then only be done if the query cannot be resolved at the first-level of support. This revised process reduces the time and costs in which it takes to resolve a query. Time is reduced by potentially having resolved the customer's query on the first call. Costs are reduced by eliminating the need to outsource the problem. Second- and third-level support are then reserved for critical issues.

2.7 Conclusions

CCs are valued-added services, for customers, that are either inbound (Section 2.3.1) or outbound (Section 2.3.2). Based on their area of operations, CCs can be classified as help desks (Section 2.3.3) or service desks (Section 2.3.4).

Several existing CC solutions were evaluated for the presence of an IUI and extensibility. It was established that only one of the software products evaluated possessed an IUI but this possessed a limited degree of extensibility and customisability; therefore, a need was identified to develop CC software that possesses an IUI and is extensible (Section 2.4.7).

Customer satisfaction (Section 2.5.1), operational costs (Section 2.5.2) and employee satisfaction (Section 2.5.3) were identified as the major challenges impacting on CC operations. There is a need to balance these three factors in order to improve the effectiveness and efficiency of a CC and its operations. It was identified that these factors directly influence the rate at which customer queries are resolved, the amount of customer satisfaction received and the operational costs of the CC. An efficient CC application can assist in balancing these three factors by retrieving, storing and presenting information, based on the customer's query, in an intelligent, efficient and effective manner.

The role of the CCA is currently limited to that of a call taker (Section 2.6). Resolution of customer queries is escalated to second- and third-level support. This process is time consuming and costly. It was identified that through the use of an IUI, the role of the CCA, at the first-level of support, can be extended to include problem solving. This eliminates the need to escalate and outsource every query. Escalation and outsourcing are then only necessary for those queries that cannot be resolved at the first-level of support and that are critical to business operations. Thus, the enhancement of the call resolution process could result in time and cost savings for the CC.

Chapter 3 presents a detailed literature study in the field of IUIs. The aim of Chapter 3 is to determine which IUI components would be the most suitable to include in the development of an IUI model for CCs. The inclusion of these components needs to address the challenges experienced by CCs, outlined in Section 2.5.

Chapter 3: Intelligent User Interfaces

3.1 Introduction

The goal of this chapter is to present a discussion on IUIs. This discussion will define IUIs and how they differ from traditional interfaces. The types of IUIs and the components of an IUI will also be discussed. A secondary objective of this chapter is to identify which of these components could assist in resolving the problems currently experienced by CCs, as discussed in Chapter 2.

3.2 Background

Computing systems were, initially, only used by a single user. All of the data residing on that computer related to that single user (Shneiderman and Maes 1997). Data was static and only changed if it was manipulated by a single user. Since there was only one user per computer, the typical computer user was an expert and there was a limited set of instructions that the user could perform (Shneiderman and Maes 1997).

Today, applications are increasing in complexity to compensate for the abundance of information available on local and global networks (Maybury 1998). Applications therefore need to be generalised in order to solve a variety of tasks (Ross 2000). Computer software available today requires interaction in order for a specific task or goal to be accomplished (Preece, Rogers and Sharp 2007). This interaction is achieved by means of an interface, which serves as a communication medium between the user and the computer. The manner in which an interface is designed dictates the way in which data is exchanged between the user and the computer.

Users of applications range from novice to expert and interfaces have evolved from command line to direct manipulation interfaces (Maybury 1998). The problem, however, still remains whereby applications have a limited level of customisability to deal with a dynamic set of users, resulting in interfaces that lack customisability and are too complex to use (Ross 2000). IUIs present a possible solution to this problem.

3.3 What is an IUI?

IUIs are defined as, "*human-machine interfaces which aim to enhance the effectiveness, efficiency and naturalness of interaction by providing multimodal input/output processing and generation, user discourse models and task models*" - Maybury 1999.

IUIs aim to enhance the efficiency, effectiveness and naturalness of interaction by possessing one or more of the following characteristics:

- Detecting and correcting user errors through the use of various models (Koelle 2004);
- Reducing the complexity of applications, through guidance from the application itself (Koelle 2004);
- Providing better multimodal communication of both knowledge and system functionality (Koelle 2004);
- Using multimodal communication through IUIs can increase the accessibility of the computer (Koelle 2004);
- Providing rapid task completion with less effort than with traditional interfaces (Maybury 1999); and
- Supporting spoken, written and gestural interaction (Maybury 1999).

The complexity of a user's task can be reduced by the prevention of errors through the interface (Hefley and Murray 1993). The skills of the user could also be improved by providing him/her with guidance in order to follow the correct mental model when solving an unfamiliar task. Different groups of users interact with computers in different ways. This difference in interaction needs to be catered for by presenting these different groups of users with different interfaces which perform the same tasks (Hefley and Murray 1993). IUIs attempt to reduce this problem by providing a common, customisable interface which provides training, proactive problem solving capabilities and facilitates the appropriate use of system capabilities (Hefley and Murray 1993).

3.4 Intelligent Interfaces vs. Traditional Interfaces

IUIs should utilise technology for the purposes of improving interaction rather than enhancing the complexity of interfaces through their advancement of interfaces (Waern 1997). Several questions need to be asked in order to determine the intelligent nature of an interface (Edmonds 1993):

- What can the interface be intelligent about;
- To what extent is change accommodated; and
- What is the procedure for allowing change?

These questions cannot be answered with a traditional interface as there are elements present in an IUI that are not part of the traditional interface. The dotted lines in Figure 3.1 illustrate that a traditional interface is only capable of distinguishing between the Presentation, Dialog Control and the Application Interface (Maybury 1998).

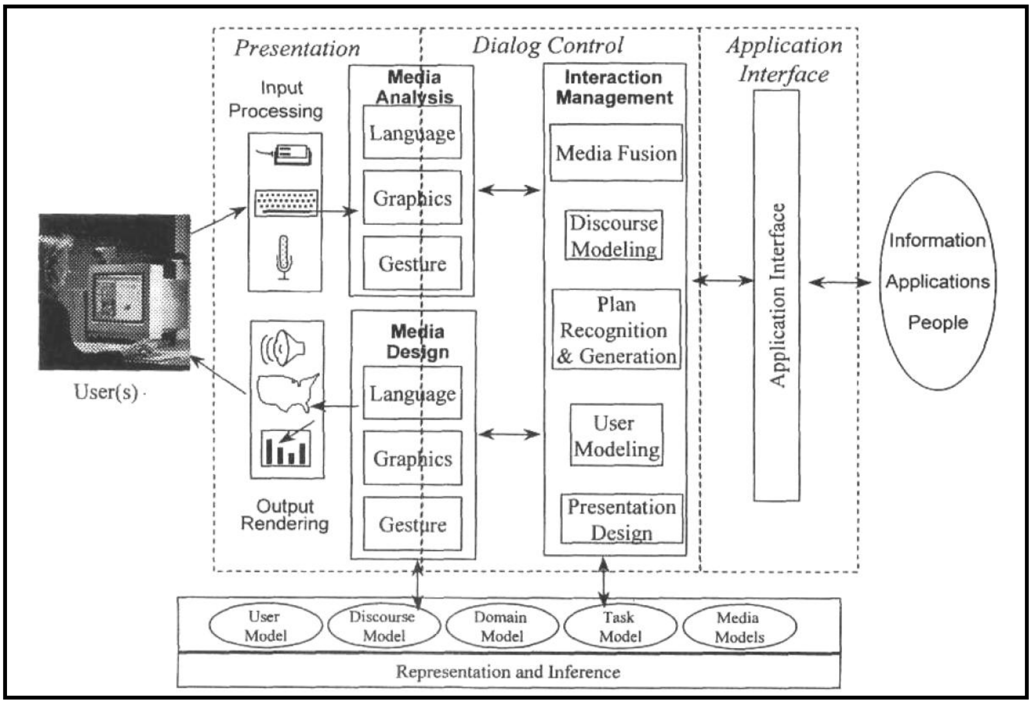


Figure 3.1: Current Interface Practice and its Relation to IUIs (Maybury 1998:4)

The Representation and Inference section in Figure 3.1 depicts the addition of various models to the traditional interface. The inclusion of these models in the design and generation of interfaces is the evolutionary step introduced by IUIs. These refinements to traditional interfaces allow for User and Discourse models, analysis of information retrieval and output generation and presentation (Maybury 1998).

3.4.1 Interface Intelligence

Interfaces can be intelligent about many things, such as (Koelle 2004):

- System functionality;
- Knowledge of how to get around the system or user tasks;
- Interfaces can be intelligent about the user and the
    - User's motor skills (Edmonds 1993);
    - Wants and needs of the user; and
    - User models allowing for individual user interaction with the interface.
- Interfaces can be intelligent about the services available to the user.

In order for an interface to be considered intelligent it needs to answerable to one or more of questions in Table 3.1 (Rich, Sidner and Lesh 2001):

No.	Questions
1.	**Who** should/can/will do ____________?
2.	**What** should I/we do next __________?
3.	**Where** am/was I __________________?
4.	**When** did I/you/we do _____________?
5.	**Why** did you/we (not) do ___________?
6.	**How** do/did I/we/you do ____________?

Table 3.1: Six Questions for an Intelligent Interface (Rich *et al. 2001*)

3.4.2 Accommodating Change

Determining to what extent change is accommodated, needs to be preceded by establishing what the interface is currently being intelligent about. This will enable the evaluation of the interface in terms of the performance it provides (Edmonds 1993).

An example of this would be if the concern was task efficiency, then the intelligence built into the interface would need to assist in reducing task completion times.

3.4.3 Change Protocols

Protocols need to be made available if change has to take place. These protocols need to stipulate who initiates the change, namely the system or the user (Edmonds 1993). Change can be initiated by either the system or the user depending on the level of intelligence built into the system and the nature of the changes that need to be made. Some operations require an effort by both the user and the system in order for the change process to occur.

3.4.4 Inclusion of Artificial Intelligence

The field of IUIs is often characterised by the significant involvement of artificial intelligence (AI). The reason behind including AI into interfaces was to facilitate for better problem solving. Figure 3.2 illustrates the various elements that AI adds to the traditional interface and the benefits achieved (Maybury 1998).

The inclusion of the AI elements in Figure 3.2 is what gives IUIs their ability to effectively and efficiently support natural human-computer interaction (Maybury 1998). These elements also provide a basis for the components that make up an IUI. Figure 3.2 emphasizes the *Representation and Inference* section in Figure 3.1 by providing User Modelling, Discourse Processing and Media Coordination.

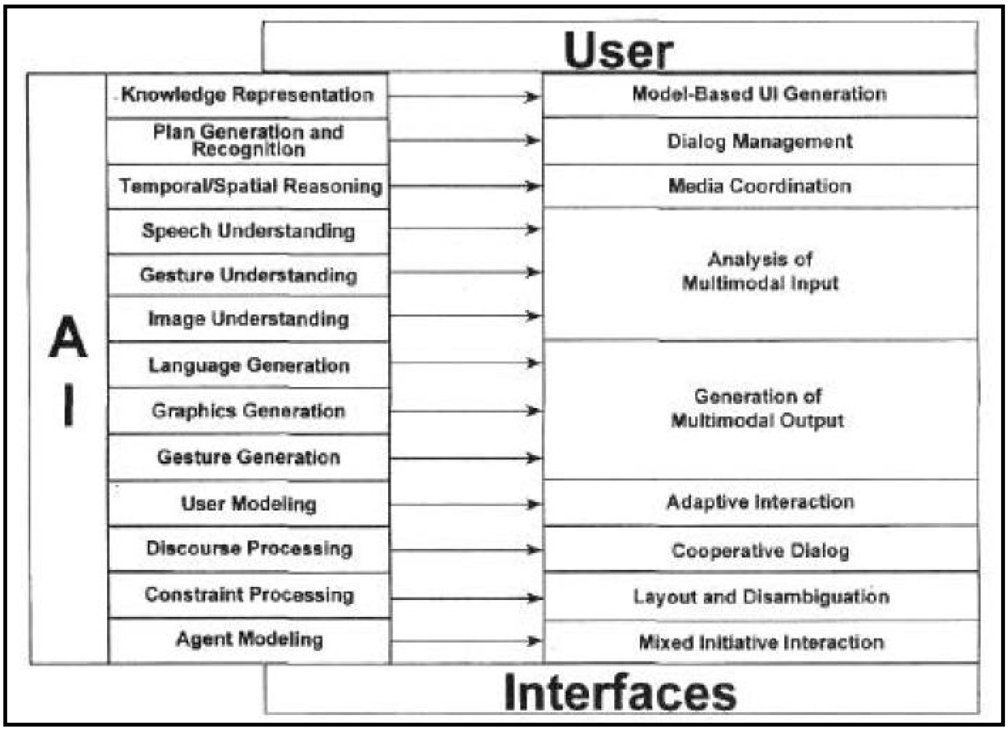


Figure 3.2: Contribution of AI to Interfaces (Maybury 1998)

Communicating naturally with the interface is an ability provided through the inclusion of Gesture Understanding, Speech Understanding, Language Generation and Gesture Generation. Explicit User, Discourse, Task and Context Models assist in supporting interaction between the user and the interface. These models provide access to any content required by the user during a particular session (Maybury 1998). Intelligent agents (IAs) or software agents deliver an autonomous action within a software environment according to its design objectives (Jennings and Wooldridge 1998). The role of a software agent is that of a proactive one; understanding the user's habits, preferences and interests (Shneiderman and Maes 1997).

The difference between a traditional interface and an IUI can be characterised by the inclusion of AI in the interface. IUIs also differ by using technology to ensure users carry out tasks in an appropriate manner whilst at the same time increasing the efficiency in performing those tasks.

3.5 Types of IUIs

IUIs can be divided into three classes (Ross 2000):

- Direct manipulation interfaces supporting adaptation;
- Intermediary interfaces; and
- Agent interfaces.

3.5.1 Direct Manipulation Interfaces

Direct manipulation interfaces supporting adaptation involves the addition of adaptation to direct manipulation interfaces. This is achieved by adding interface objects which contain the predicted future commands. The other alternative is an interface containing multiple commands (unlikely to be used) within single objects (Ross 2000).

3.5.2 Intermediary Interfaces

In order for the interface to process and generate information it must act as an intermediary between the user and the direct manipulation interface (Ross 2000).

IUIs can also be viewed as adaptive interfaces due to their ability to dynamically adapt to the user or the user's task (Ross 2000). The high-level of interaction in modern software will see adaptive interfaces being introduced as the next generation of user interfaces. These new types of interfaces can adapt to the different work ethic, levels of experience and expertise of different users, (Ross 2000). Their ability to adapt to the user and learn involves the use of a machine learning which is achieved through user models. Adaptation in the interface is only useful if it allows the user to complete a task in a shorter amount of time and provides a higher level of satisfaction (Ross 2000).

Hindering the user's productivity by either slowing down the computer or performing any unnecessary operation that will slow down the user's progress towards completing a task will render the adaptation of the interface useless (Ross 2000).

3.5.3 Agent Interfaces

IAs are autonomous and could allow for IUIs to guide users when performing tasks. Users would still have complete control over the interface and the agent would be visible at all times. This form or style of interaction can be referred to as indirect-management (Ross 2000; Shneiderman and Maes 1997). Agent interfaces are typically accompanied by good direct manipulation interfaces (Shneiderman and Maes 1997). IAs interact with the application as a typical user would. The IAs monitor the user's habits and preferences resulting in the automation of the user's most frequently performed tasks (Shneiderman and Maes 1997). IAs do not necessarily rely on AI for their inferencing and knowledge representation; they can also rely on user programming and machine learning (Shneiderman and Maes 1997).

3.6 IUI Components

Interfaces cannot be intelligent by themselves and require additional elements or components that provide them with the necessary intelligence. IUIs need to employ one or more intelligent techniques in order to make them intelligent (Waern 1997).

Techniques that can be employed are (Waern 1997) :

- User Adaptivity;
- User Modelling;
- Natural Language Processing;
- Dialogue Modelling; and
- Explanation Generation.

These techniques can be broken down into various components such as:

- The User Model;
- Multimodal communication;
- Plan recognition;
- Dynamic presentation;
- Natural language;
- Intelligent help; and
- Interface adaptability.

In order for an interface to be regarded as an IUI, it would need to possess one or more of these components (Koelle 2004). These components are discussed in further detail with the intention of providing a greater understanding of the role and purpose of IUIs.

3.6.1 The User Model

User and discourse models are necessary in order for a system to exhibit behavior that is both intelligent and cooperative (Wahlster 1991). The use of user models will have a significant benefit to those systems that (Kass and Finin 1991):

- Seek to adapt to the individual needs of its users;
- Assume responsibility for ensuring successful communication between the user and the system; and
- Systems where there is a diverse set of users.

User modelling is an integral part of IUIs and provides the interface with the necessary information on which it can base its design and individuality (Koelle 2004). These models allow for interfaces to adapt in order to suit their users (Ross 2000). Catering for the individual needs of a user can be done by allow the user to manually modify the system to suit his/her needs. This allows for the creation of an environment within the limitations of the existing system (Rich 1999).

Providing users with this amount of freedom often raises concerns especially if the user is an infrequent novice user. An alternative approach is to provide the system with sufficient information about the user so that it can intelligently construct a personalised environment for the user (Rich 1999).

A user model can be defined as a source of knowledge or an explicit representation of all the aspects of the user (in terms of the user's needs, preferences and behavior) that may be relevant to the behavior of the system (Koelle 2004; Kobsa and Wahlster 1989; Ross 2000; Wahlster 1991). User models are used to get input from the user, provide advice and help, to understand the user's information seeking behavior and to provide output to the user (Kass and Finin 1991). A system containing user models must support representation and maintenance for the contents of existing models, access to the model contents by various parts of the system or interface and it should support acquisition and creation facilities to build user models incrementally (Kass and Finin 1991; Kobsa and Wahlster 1989; Wahlster 1991).

User models can be classified in one of four types (Ross 2000):

- Users can either be modeled as canonical (typical) or individual;
- Modeling information can either be constructed by the user or generated by the system based on the user's behavior over a period of time;
- Short-term (time sensitivity) models contain specified information and long-term models contain more general information; and
- Updating methods can be done for either static or dynamic models.

Static user models contain data for a canonical (typical) user and are embedded within a system. Static models do not need to be stored explicitly as compared to dynamic models. Dynamic models cater for the individual user and require explicit methods to determine how the user model influences the system's performance (Ross 2000).

User models in a system will only be useful if they have one or more of the following characteristics (Kass and Finin 1991):

- The system must adapt its behavior to that of its individual users;
- The system assumes the responsibility for ensuring a successful communication between the user and the system; and
- There is a diverse set of users of the system.

The creation of user models can be done by deriving information of how a user uses the system and by obtaining information from the user's behavior patterns (Rich 1999). An alternative is to explicitly ask the user for information (Ross 2000).

The first approach to creating a user model based on monitoring the user's actions can be implemented by creating a dictionary of the system commands and options and then inferring the skill level of the user by means of the commands entered (Rich 1999).

Building the model by means of patterns involves looking at how users request information and what they do with the results, i.e. do they restructure the query in order to find something more accurate or do they accept the generated results (Rich 1999).

User models can be used to get input from the user in terms of deciding how and what to ask. Combined with any other IUI component, user models provide a powerful and dynamic interface. User models are especially powerful if used to deliver and support multimodal communication (Koelle 2004).

3.6.2 Multimodal Communication

The use of multiple modalities in order to interact with a system (via an interface) is referred to as multimodal communication (Chai, Hong and Zhou 2004; Koelle 2004). Modalities such as text, speech, gesture and gaze can be used and combined to deliver an interface that processes and generates multimodal input and output (Chai *et al.* 2004). As opposed to the traditional keyboard and mouse interface, multimodal interfaces add a new dimension of input through familiar communication modalities (Oviatt, Coulston and Lunsford 2004, Trompf 2001). The ability to facilitate multimodal communication often requires user models in order for the communication to be personalised for particular users (Koelle 2004).

An advantage of multimodal communication is that it allows the user to use the system more intuitively and naturally by using gestures, speaking through a microphone, typing in the language and manner of choice and the ability to receive output in the manner that is preferred by the user (Chai *et al.* 2004; Koelle 2004).

Multimodal interfaces also improve the accessibility of an application to users of different ages, skill levels, sensory impairments, language barriers and temporary illnesses (Oviatt and Cohen 2000).

Building a multimodal system does not imply that users will perform every action and command using the multimodal means available (Oviatt 1999). Users natural communication patterns involve a combination of both multi and unimodal expressions. Multimodal systems should be able to differentiate when users are using multimodal communication and when they are not (Oviatt 1999).

Figure 3.3 is similar to Figure 3.1 as it illustrates the general architecture of a multimodal input/output system which applies intelligent technologies in order to control and interpret the user's actions (Trompf 2001). The blocks on the extreme left and right represent the various modes of input and output either analysed or generated. Controlling and coordinating tasks such as the activating and deactivating of devices, conflict management and temporal reasoning is the responsibility of the Device and Media Management modules on the left and right. The block in the middle represents the libraries which contain the predefined user actions and corresponding system reactions. This module also controls and activates the user interface agent (Trompf 2001).

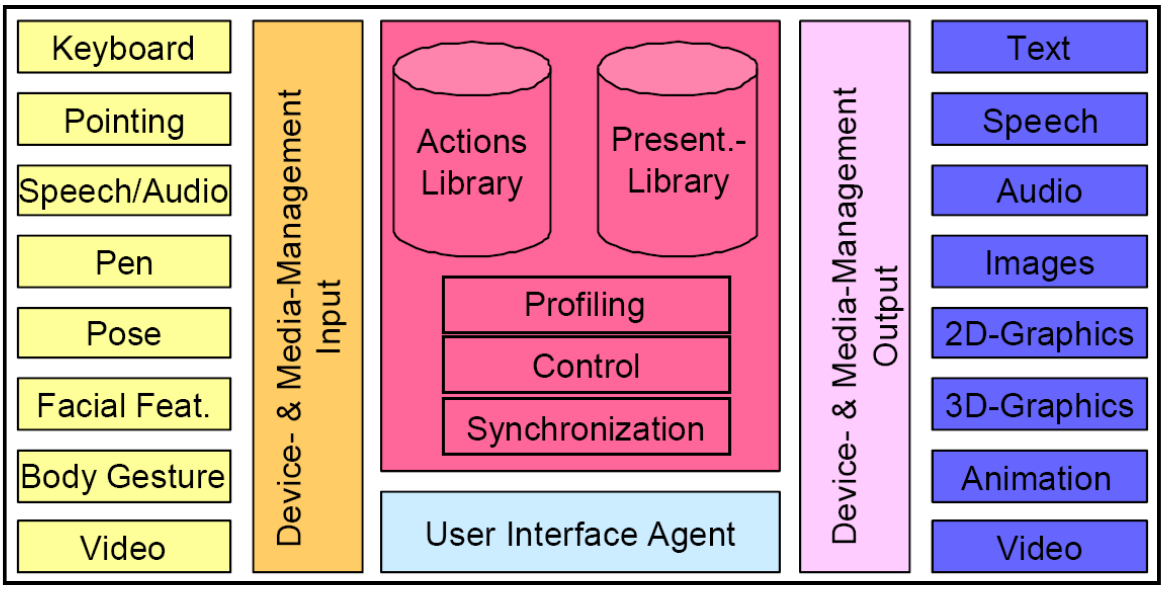


Figure 3.3: Generic Multimodal I\O Architecture (Trompf 2001)

Integrating user models with this architecture resembles the IUI architecture (Figure 3.1) and thus makes the architecture different from a traditional interface consisting of the Presentation, Dialog Control and Application Interface.

3.6.3 Plan Recognition

Plan recognition is used in IUIs in order to anticipate what user's plan. This can be defined as the process of inferring intentions from actions (Koelle 2004; Lesh, Rich and Sider 2001). This concept of recognition is accomplished by combining knowledge of the system, the user model and the user's actions (Koelle 2004). The benefits of plan recognition are evident in its assistance to the user by providing guidance and thus requiring less steps or repetition. Plan recognition is well known within the domain of human collaboration but has proved difficult to implement in human-computer collaboration systems (Lesh *et al.* 2001).

Figure 3.4 illustrates the possibility of two participants communicating whilst observing each other's actions on a shared artifact (interface or task). This illustration relates to the communication between the user and the interface agent (system) (Lesh *et al.* 2001).

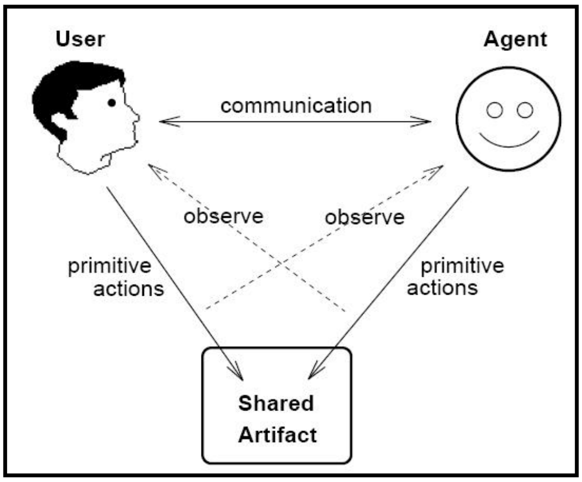


Figure 3.4: Human - Agent Collaboration (Lesh *et al.* 2001)

The collaboration will only be successful if there is a mutual understanding amongst the participants of their shared goals and the actions needed to achieve these goals (Lesh *et al.* 2001).

Figure 3.5 shows the completion of an e-mail task with a system called Collagen. Collagen is an application-independent collaboration manager based on the SharedPlan theory of task-oriented collaborative discourse (Lesh *et al.* 2001). The left column of Figure 3.5 represents how Collagen functions without plan recognition and the right column shows how it functions with plan recognition (Lesh *et al.* 2001).

without plan recognition		*with plan recognition*	
USER:	"Let's work on email."	USER:	"Let's work on email."
AGENT:	"You have 5 unread messages."	AGENT:	"You have 5 unread messages."
USER:	"Let's read a message."		
USER:	<Select message from Chuck>	USER:	<Select message from Chuck>
USER:	"Let's read message in its own window."		
USER:	<Open message>	USER:	<Open message>
USER:	"Let's do something using message."		
USER:	<Visit http://foo.bar.com>	USER:	<Visit http://foo.bar.com>
	...long lunch...		...long lunch...
USER:	"What was I doing?"	USER:	"What was I doing?"
AGENT:	"Working on email.	AGENT:	"Working on email.
	Reading message from Chuck.		Reading message from Chuck.
	Reading message in its own window.		Reading message in its own window.
	Done something using message.		Done something using message.
	<Visit http://foo.bar.com>		<Visit http://foo.bar.com>
	Expecting to react to message (opt)."		Expecting to react to message (opt)."
USER:	"Let's react to message."		
USER:	"Let's send email."		
USER:	<Start new message>	USER:	<Start new message>
USER:	"Let's fill in message."		
USER:	<Address message to Dan>	USER:	<Address message to Dan>
		AGENT:	"Why are you sending email to Dan?
			(a) reacting to message from Chuck
			(b) as a new goal "
		USER:	"I am reacting to message from Chuck."
	...additional editing actions...		...additional editing actions...
USER:	"Done filling in message."	USER:	"Done filling in message."
AGENT:	"Let's send, cancel or save message."	AGENT:	"Let's send, cancel or save message."
USER:	"Please save message."	USER:	"Please save message."
AGENT:	<Save message>	AGENT:	<Save message>

Figure 3.5: Collagen e-mail with and without plan recognition (Lesh *et al.* 2001)

It is evident from the above illustration that when there is no planned recognition the user has to announce each goal before actually performing the action required to achieving that goal. The inclusion of plan recognition, however, does not require the user to announce each goal and sub-goal. Although plan recognition eliminates the need for some communication about the user's intention it does not eliminate the need entirely (Lesh *et al.* 2001). The ability to deliver information in real-time is taken one step further in the next section whereby the format in which the information is delivered is tailored according to the user's preference.

3.6.4 Dynamic Presentation

Dynamic presentation is another component that interacts closely with the user model as it allows for data to be displayed according to the user's specifications (Koelle 2004). The presentation of data should be done in a manner that is effective and understandable and should not under or over compensate by providing too little or too much information (Koelle 2004).

Figure 3.1 illustrates a high level architecture of an IUI and describes the intelligence as being those modules such as Media Analysis, Interaction Management and Media Design. The role of the Media Analysis module is to analyse the input entered via the multimodal interface. This data is then processed by the Interaction Management and the results are sent to the Media Design module which then designs and displays the output according to the requirements of the user as specified in the User Model.

Dynamic representation postpones all presentation decisions until run-time leaving all forms of communication and representation to be generated dynamically (Maybury 1998).

3.6.5 Natural Language Processing

Natural languages are part of the attempt to make interaction between the user and the computer more intuitive, effective and natural (Cohen 1992; Koelle 2004; Lieberman 2001). The ability to provide support for natural languages stems from the possibility of creating a medium of communication between the user and the computer that would be as natural as the communication between two users (Lieberman 2001). This provides the user with the feeling that the system is acting intelligently, thus invoking the feeling of a personalised interaction with the system (Lieberman 2001). Natural languages allow for users to enter commands often replacing the need to use cumbersome menu structures. These commands do not have to be listed in a menu and can be constructed as a combination of other commands (Koelle 2004).

The advantages of natural languages lie in their ability to avoid extensive redescription by making use of pronouns and other anaphoric expressions (Cohen 1992). When integrated into interfaces, natural languages have several disadvantages:

- The user knows that the system cannot interpret everything but he/she does not know to what extent he/she is restricted;
- Systems are error-prone resulting in users having to enter commands several times in order to achieve the desired result; and
- Natural language sentences are ambiguous and phrases could identify more ambiguity.

Natural language can be present in the form of Speech Understanding, Gesture Understanding, Gesture Generation and Language Understanding (Figure 3.2). The intention behind combining these elements is to provide an environment supporting a more effective, efficient and natural-level of human-computer interaction (Maybury 1998). Interaction by means of fluent conversation requires the use of explicit User, Discourse Task and Task models (Maybury 1998).

Natural languages are an important part of IUIs as they deviate from traditional modes of interacting with the system. The ability to communicate naturally with a system falls within the domain of multimodal communication if implemented.

3.6.6 Intelligent Help

The ability of computers to act in the role of an advisor and an advisee is gaining popularity as applications become increasingly complex (Lieberman 2001). Intelligent help presents users with help at a particular or specific moment in time for a specific situation and requires knowledge about the application functionality (Koelle 2004). Plan recognition can also be used with intelligent help to anticipate problems the user might encounter whilst completing a task (Koelle 2004). Utilising a planning based approach to the generation of context sensitive help presents multiple advantages over the techniques used for traditional help generation (Ramachandran and Young 2005).

3.6.7 Interface Adaptability

Users might not be satisfied with the interface with which they are presented for a given system and might prefer a degree of preference in terms of customising the interface. The ability for an interface to react or adapt explicitly to the requirements of the user or implicitly by observing the user's actions and preferences can be defined as an adaptive interface (Koelle 2004). Adaptive interfaces utilise adaptive systems to determine what type of interface to present to the user depending on the user model (Koelle 2004). An adaptive system is one which adapts its behavior to individual users through the processing and application of user models involving some form of learning, inference or decision making (Jameson 2003).

Figure 3.6 illustrates this definition and represents the general schema for processing within a user-adaptive system.

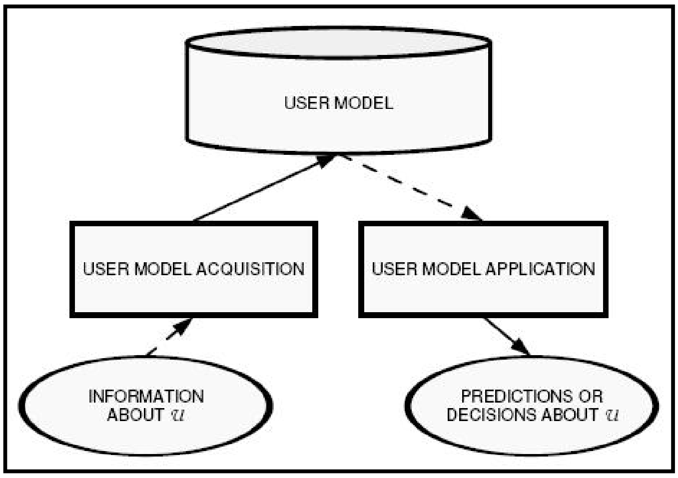


Figure 3.6: General Processing Schema for Adaptive Systems (Jameson 2003)

The adaptive system applies the user model to current situations in order to determine how to adapt to the behaviour of the user (Jameson 2003). Adapting the interface to suit the user is one of the ways in which to increase the effectiveness with which the user interacts with the system. This involves taking into account any perceptual or physical impediments of the user to facilitate minimal error whilst interacting with the interface (Jameson 2003).

This concludes the discussion on the components of IUIs. All of the components mentioned play a significant role in defining the purpose of IUIs, however when deciding which of these components are most suitable to the domain of CCs, one must keep in mind that the purpose of an IUI is to increase the rate at which information is captured and processed.

CC applications process large amounts of data daily and CCAs are escalated from novice to experts within a matter of weeks. This is due to the repetition of their daily tasks. IUIs would therefore need to increase the rate at which CCAs move from being a novice to an expert. The scope of this research (Section 1.3.2) is limited to diagnosing customer queries and gathering information in order to resolve those queries, with respect to a CC and its operations. Based on this definition, it is evident that natural language processing would benefit the domain of CCs as it would allow for the description of customer queries in a naturalistic manner. The other component that will be useful to the domain of CCs is intelligent help and plan recognition. These components would facilitate the finding of information in order to solve customer queries. The inclusion of intelligent help and plan recognition implies the use of user models.

3.6.8 Intelligent Agents

Agents can be defined as autonomous, problem-solving computational entities that are capable of operating effectively in environments that are open and dynamic (Luck, Ashri and D'Inverno 2004). Lieberman and Selker (2003) describe agents as multiple goal-seeking entities which compete and cooperate to produce intelligent behaviour. The term *agent* can be used in two ways - a weak notation and a strong notation (Wooldridge and Jennings 1995). The weak notion of the term agent describes the role of agents as being able to operate without intervention by the user and to have some control over their actions (autonomy) (Wooldridge and Jennings 1995). Agents are also capable of interacting with other agents through a familiar protocol. The strong notion of the term agent describes agents as being computer systems which are either implemented or conceptualised by using the concepts which are applied or associated to users (Wooldridge and Jennings 1995). One of the ways of doing this is to present the agent as a graphical representation (interface agent).

There are various types of agents such as software agents, intelligent agents, mobile agents, interface agents and virtual agents (Luck *et al.* 2004). One of the most familiar agents is Clippy from Microsoft Office which first appeared in Microsoft Office 97. Software agents recognise the user's habits, preferences and interests and are able to act proactively (Shneiderman and Maes 1997). These types of agents can be described as being autonomous and adaptive as they track the user's interests over a period of time (Shneiderman and Maes 1997). This section will focus primarily on the role of an IA. IAs can also be defined as programs that affect the objects in a direct manipulation interface without having any explicit instruction or intervention from the user (Lieberman 1997).

The inputs entered by the user to the interface can be read by the IA, which would then make changes to the objects on the screen that are visible to the user based on those inputs. Over a longer period of time, the IA can observe the actions of the user before deciding to take action (Lieberman 1997).

Due to the complexity of current user interfaces (UIs) there is an increasing interest in the field of IAs (Lieberman 1997). Figure 3.7 illustrates the structure of an application utilising IAs. The user can provide an application with input which is then processed and the respective output is generated. In an IA application this is taken one step further whereby the IA observes the actions of the user and makes the necessary changes to the interface to suit these actions.

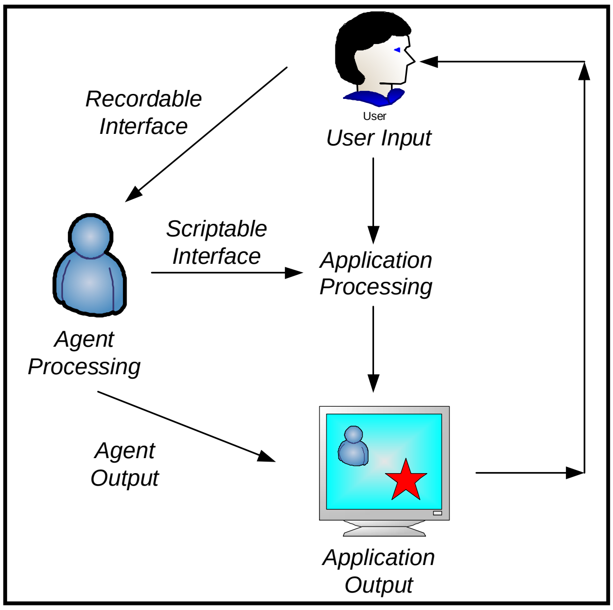


Figure 3.7: Structure of an IA Application (Lieberman 1997)

IAs can make changes to the visual interface without having to run in parallel with the user (Lieberman 1997). User models can be used in conjunction with IAs in order to make the user model more explicit for the purposes of manipulation and reasoning (Section 3.6.1).

If a dynamic user model is employed then the system could identify that the user is using a certain section of the system for the first time and offer assistance (Lieberman and Selker 2003). IAs could aid the construction of adaptive user models which are not pre-programmed and are based on the user's interaction with the system.

Construction of user models via the IA can also be done by explicitly interacting with the user and by implicitly observing the user's behavior and make suggestions to improve the performance of a task (Lieberman and Selker 2003).

Figure 3.8 shows the processes for initialising and updating the user model as well as the system adaptation based on the user model (de Vrieze, van Bommel and van der Weide 2004).

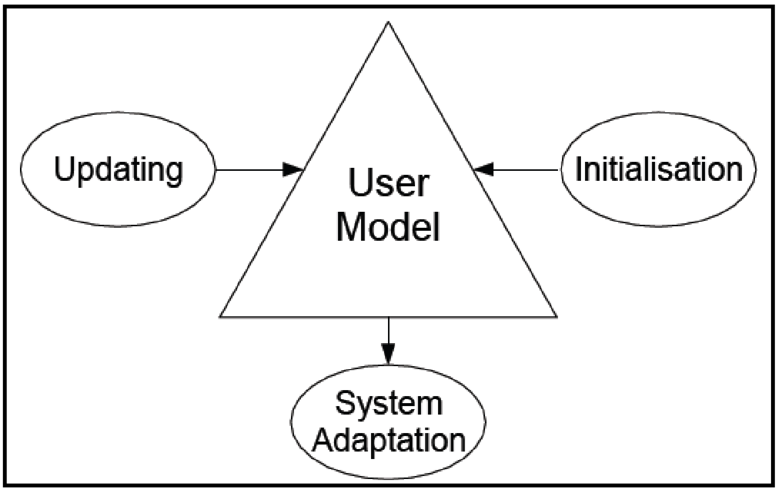


Figure 3.8: User Model Processes (de Vrieze *et al.* 2004)

As discussed above, all of these processes can be handled by the IA. Agent–based interaction is a critical part of an IUI as it facilitates in decreasing task difficulty and presenting users with a more natural environment in which to interact (Maybury 1999).

3.7 Conclusion

The goal of this chapter was to determine how IUIs could address the problems currently experienced by CCs. This objective was met by means of a literature study.

IUIs were defined as human-machine interfaces that aim to amplify the rate of flow of information between the user and the computer. This is accomplished by delivering interaction that is effective, efficient and natural (Section 3.3).

It was identified that the inclusion of AI within the interface is the differentiating factor between an IUI and a traditional UI (Section 3.4). Another differentiating factor is that IUIs utilise technology to ensure and enhance task completion.

This chapter has identified that there are three different types of IUIs. IUIs can either be: direct manipulation interfaces supporting adaptation; intermediary interfaces; or agent interfaces (Section 3.5). These various types of IUIs consist of several components that provide the UI with the intelligence (Section 3.6).

Several components were discussed in order to determine which components would be able to assist in addressing the problems currently experienced by CCAs. The components identified that could potentially benefit the domain of CCs are:

- Natural language processing;
- Intelligent help; and
- Plan recognition.

These components would assist in allowing for queries to be described in a naturalistic manner and assist in the finding and delivery of information. The inclusion of intelligent help and plan recognition would require the use of user models, therefore including user models as a necessary component to improve the operations of a CC.

The next chapter (Chapter 4) will compare several existing IUI models to determine whether any one of these models could be specialised to the domain of CCs.

Chapter 4:

Model-Based Design of IUIs

4.1 Introduction

The main objective of this research is to develop an IUI model for the domain of CCs. An understanding of model theory is required before a model can be designed and proposed. This chapter discusses model-based design theory in order to identify the role played by models in the design process. This is necessary in order to perform an analysis of several existing IUI models. The objective of this analysis is to select the most appropriate IUI model that could be specialised for the domain of CCs.

4.2 What is a Model?

A model can be defined as the representation of a system attempting to capture the essential aspects of a system or process (Olivier 2006; Aagedal and Mohagheghi 2007). The non-essential aspects are ignored (hidden) to give focus to the more important aspects. A model is often proposed before the implementation of a system (Gomaa and Hussein 2007). Proof that the proposed model works is obtained by directing the implementation of a system through the use of the proposed model (Olivier 2006; Gomaa and Hussein 2007).

Models are not merely diagrams representing a single abstract state of a system (Berenbach and Konrad 2007). A model needs to illustrate the different views of a system (Pressman 2005). This includes all levels of abstraction. The different views of a model are often communicated through the use of a modelling language (Gomaa and Hussein 2007). Modelling languages are required as programming languages lack the level of abstraction which encourage discussions about design (Berenbach and Konrad 2007).

Unified Modelling Language (UML) has emerged as the de facto modelling language for modern software systems (Coninx and Van den Bergh 2004).

4.3 Characteristics and Quality of a Model

A model needs to possess the characteristics presented in Table 4.1.

Characteristic	Definition
Simplicity	Allows for the essential aspects of the model to be easily understood.
Comprehensiveness	A model needs to address all of the aspects of a problem.
Generality	The greater the number of variations of a problem a model can address the better.
Exactness	The closer the model fits the problem it is trying to address the better the model and the greater the chances of it being accepted.
Clarity	All components of a model and their interaction should be clearly presented and explained.

Table 4.1: Typical Characteristics of a Model (Olivier 2006)

Whilst possessing these characteristics the quality of a model should not be compromised. The quality of a model is dependent on the factors presented in Table 4.2.

Factor	Definition
Quality of Modelling Language	The modelling language used should be appropriate for the domain and complexity of the problem.
Quality of Modelling Tool	The modelling tool utilised should comply with recognised modelling languages and possess the capability of combining information.
Quality of Modelling Process	The quality of the process used to create/design the model.
Quality Assurance Techniques	The techniques applied to iron out the faults or weaknesses of the model.
Knowledge of Developers	The level of modelling knowledge in terms of languages, processes and tools is important. The more the better.

Table 4.2: Quality Assurance Factors of Model Design (Aagedal and Mohagheghi 2007)

4.4 Types of Models

Models can be broken down into four distinct types. These are: Application Models; Abstract Presentation Models; Concrete Presentation Models; and Task/Dialog Models (Coninx and Van den Bergh 2004).

Application Models can also be referred to as concept or Domain Models and rely on an interface in order for information exchange and manipulation to take place. Data for this type of model is of a specific type and structure. These types of models serve as the layer between the interface and the application program interface (API) (Coninx and Van den Bergh 2004).

The Abstract Presentation Model concerns itself primarily with the user interface. It attempts to describe how the UI is structured and can be presented as a single presentation unit. The Concrete Presentation Model then takes this information and translates it into interface instances for a specific context (Coninx and Van den Bergh 2004).

Task and Dialog Models occur at different levels of abstraction. These models are used during different stages of system development. Task Models are primarily concern with the tasks or activities that have to be performed. Dialog Models differ by dealing with how these activities can be organised within the user interface (Coninx and Van den Bergh 2004).

4.5 Elements of a Model

Designing a model for the purposes of system design and implementation requires that certain elements be present. These elements are: the Data/Class Design element; the Architectural Design element; the Component-Level Design element; and the Interface Design element (Pressman 2005). The Data/Class Design element transforms the analysis-class models into design classes and its associated data structures and algorithms. These classes, structures and algorithms can then be used to implement the data aspect of the system, such as a knowledge base. Architectural design aims to describe the relationship between the major components of the software. The representation of the architecture can be derived from system specifications. The Component-Level Design element is responsible for transforming the components of the architecture into a procedural description. Interface Design describes how the software will communicate and receive information to and from its users. These elements can be designed and represented through the use of UML. Once designed, these elements can then be used to aid the implementation process. Use of a process model could assist in tracking and guiding the development life-cycle of the system (Pressman 2005).

4.6 Stages of a Model

There are three different stages that exist during the design of model. These stages are (Olivier 2006):

- Clarification;
- Differentiation; and
- Generalisation.

The clarification stage occurs when a model is required to address a problem. A thorough understanding of the problem and its domain is needed before a new model can be proposed (Olivier 2006). The second stage is to determine whether the new proposed model addresses the identified problem. This needs to be done in a manner that is different from existing models. An indication as to why the new proposed model is better than existing models is also required. This is necessary in order to establish the contribution made by the new model (Olivier 2006). Generalisation is necessary in order to design a model that is comprehensive. These types of models should cater for addressing all (most) aspects of the impending problem. Designing a generalised model is best done by performing an analysis of existing models. This analysis should include: the components of the model; the contribution of the model in addressing the problem; and the shortcomings of the model (Olivier 2006).

4.7 Existing IUI Models

The purpose of this section is to review existing IUI models to determine which model is the most appropriate for the domain of CCs. The slow rate of progress in the field of IUIs is a direct result of researchers not placing enough emphasis on the study of IUI models (Puerta 1993). The development and description of models is imperative because it establishes the systems requirements and defines the various functionalities that the system can provide (Puerta 1993).

Interfaces have progressed from command line interfaces to mainly direct manipulation interfaces (Maybury 2001). These type of interfaces follow an architecture that is consistent with Figure 4.1.

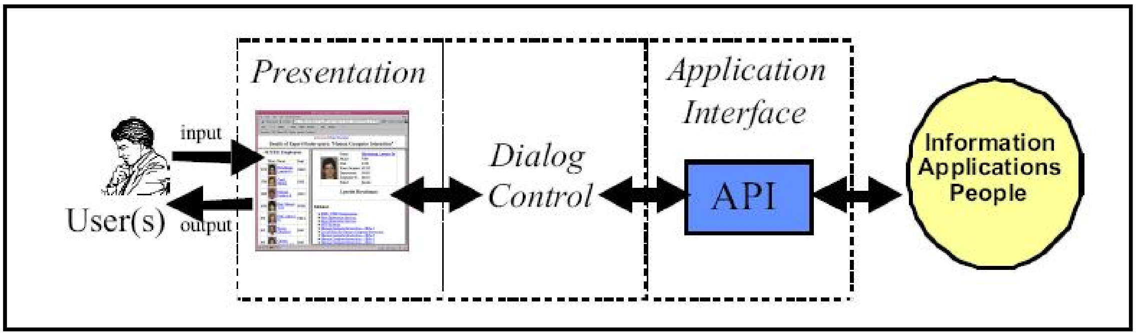


Figure 4.1 : Traditional Interface Architecture (Maybury 2001)

The Presentation section of the architecture refers to the widgets which are the primary means of control between the user and the system. Data and flow control logic that manage the sensing of actions from the user interactions with the Presentation section is done by the Dialogue Control. Lastly, the API contains the methods that are called in order to execute the underlying functions of the system (Maybury 2001).

IUIs differ in that they are both knowledge-based and modular and should be able to make inferences about the user based on the user's behavior and the current task (Tyler, Schlossberg, Gargan, Cook and Sullivan 1991). IUIs must be capable of detecting and correcting user errors and plans in accordance with the application.

IUIs should also be capable of supporting multimodal input allowing users to interact with the system in the mode of their choice (Tyler *et al.* 1991).

In order to determine the effective and extensible nature of an IUI model, a set of evaluation criteria needs to be established. The primary challenge of an interface that is truly universal is to represent, reason and exploit domain, task, user and dialog models (Maybury 2001). The result of this is the ability to process various modes of input, generate output and manage the dialogue and interaction between the user and the system to provide the greatest level of effectiveness, efficiency and naturalness (Maybury 2001). Table 4.3 contains a list of interface components that should be present in any IUI. These components were derived from the components in Section 3.6. The evaluation of various models in this section will be based against the components listed in Table 4.3.

A suitable IUI model should contain an Input Analysis module which presents the user with the opportunity to provide input in a variety of ways. The model should also contain an Output Generation module that delivers the required output to the user in a format that is appropriate for each individual user. Between the Input Analysis and the Output Generation modules there needs to be a Dialog Control module that controls the interaction between the user and the system. It could be used for plan recognition and error detection and correction capabilities. User modelling is a crucial component of IUIs and IAs can be used to create and maintain these user models. A module for Agent / User modelling is therefore required. All of these models and domain information need to be stored within one or more repositories. These repositories take the form of a Knowledge Base. The last module that needs to be present in an IUI is the API which is responsible for executing functions requested by the system.

Interface Components	Purpose
Input Analysis	Facilitates multimodal means of input
Output Generation	Delivers multimodal output tailored to the user's specifications
Dialog Control	Responsible for the tailoring flow and controlling the interactions of the user and facilitates for error detection and correction
Agent / User Modelling	Utilising IAs to build and maintain user models to deliver a personalized and natural means of interaction
Knowledge Base	Repository(ies) for storing the User Mode, Task Model, Domain Model, User Interface Model and Vocabulary
API	Methods called to execute the underlying functions of the system

Table 4.3: IUI Architecture / Model Evaluation Criteria

Figure 4.2 depicts an IUI architecture that considers the interface as a module that is separate from the target application (Tyler *et al.* 1991). It operates as a set of functions which complement several knowledge bases which facilitate user learning and operation of the target system.

Evaluating this architecture against the criteria in Table 4.3 reveals that the Input Analysis and Output Generation components are present in the form of the Input/Output Manager. The Dialog Control component is supported in terms of the Plan Manager and the Presentation Manager. Agent/User Modelling is handled by the Adaptor component of the architecture. The API is represented by the Translator component of the Architecture.

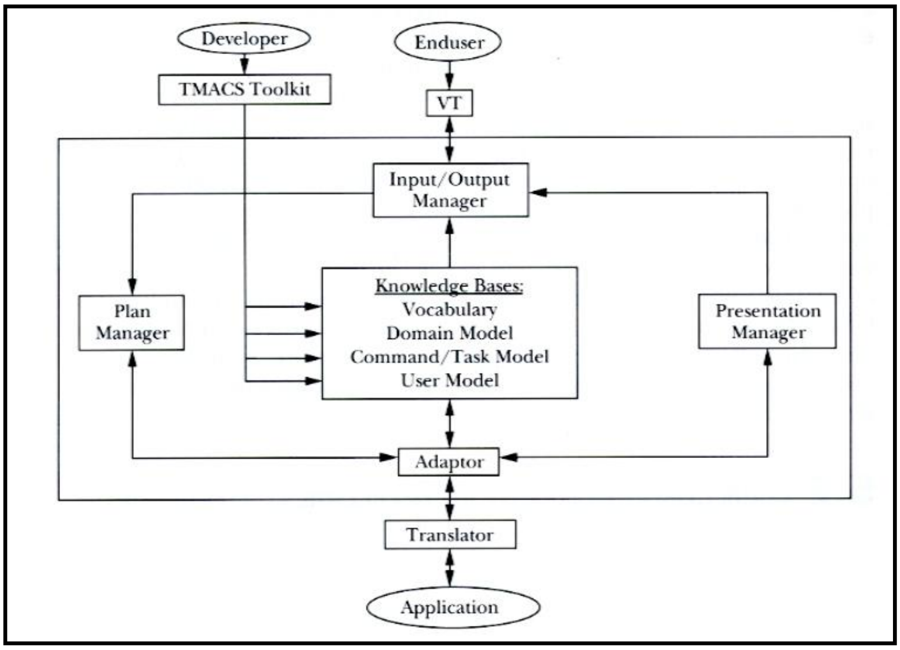


Figure 4.2: Intelligent Interface Architecture (Tyler *et al.* 1991)

The IUI Architecture as proposed by Tyler *et al.* (1991) meets all the requirements of an IUI model and could be used as a basis for this research. The role of the Adaptor is similar to that of an agent according to Agent / User Model description in Table 4.3. This component could be renamed to meet all of the criteria stipulated in Table 4.3. Only the name of the component would change to that of the Agent Manager, the functionality of the component would remain the same.

The next IUI model is the FOCUS architecture (Edmonds 1993). The FOCUS architecture contains Knowledge-Based Support Modules. These modules extend the front-end of the system and are intelligent about the user's tasks and the availability of software to support it. The role of the Back-End Manager is to separate the rest of the interface from the application, and is intelligent about the use of existing services. Between the Knowledge-Based Support Modules and the Back-End Manager lies the Harness which is intelligent about the dialogue management and the presentation (Edmonds 1993).

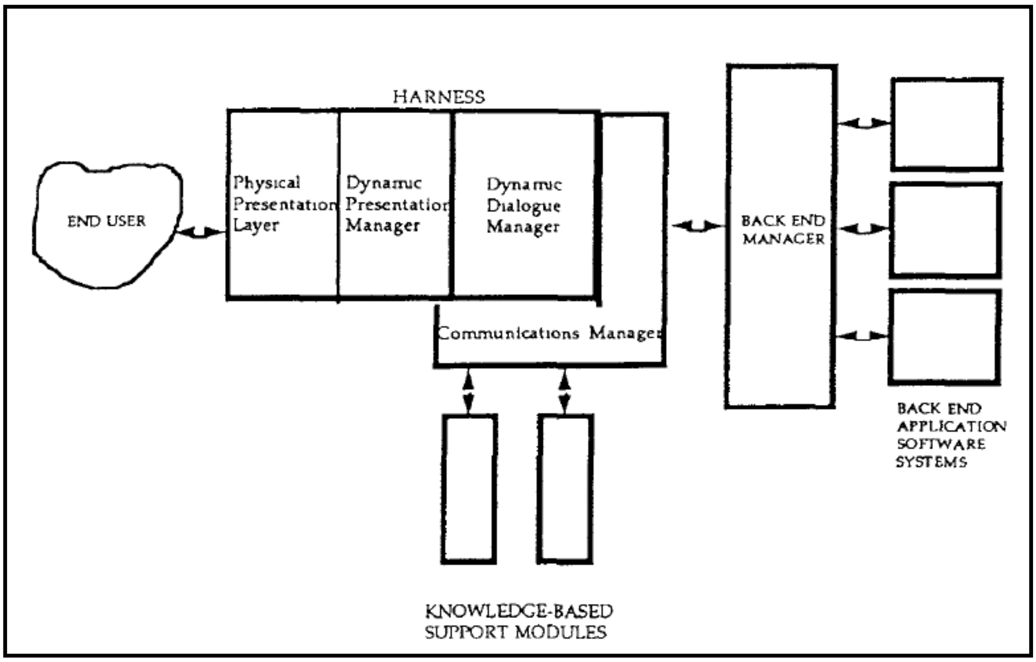


Figure 4.3: Focus Architecture (Edmonds 1993)

Evaluation of the FOCUS Architecture (Edmonds 1993) reveals a model that has limited capabilities to deliver a multimodal system but no explicit reference is made to Input and Output Generators. The role of the Presentation Manager is not described in detail and therefore no further information is provided about the interaction of these managers with user models. User modelling is not evident in Figure 4.3 and thus the availability of agents supporting it is also not present. Knowledge Base support modules are present, however no explicit reference is made to the presence of a Knowledge Base component. The Harness and Back-End Manager do however, perform tasks that are similar to the Dialogue Control and API.

Figure 4.4 shows the architecture of an adaptive intelligent human-machine interface. The domain specific components are responsible for storing user model data as well as domain specific and dynamic knowledge bases. The Domain Adaptation component in Figure 4.4 deals with the overall goal management and high-level user modelling. The Discourse Management layer deals with the management of the interaction and action/presentation-level of the user modelling process (Hefley and Murray 1993).

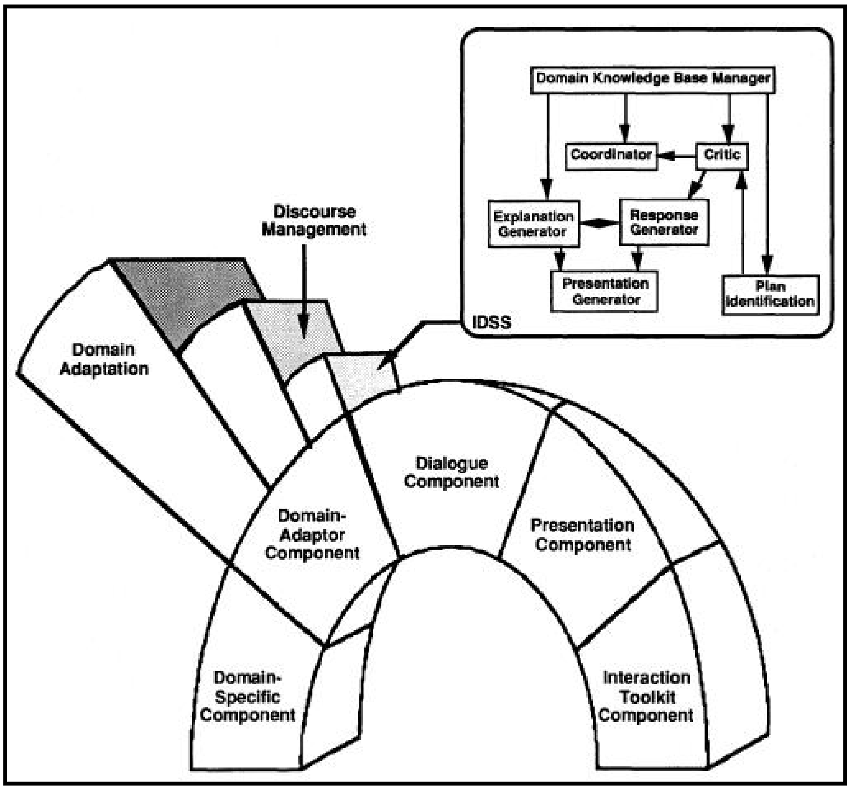


Figure 4.4: Adaptive Intelligent Interface Architecture (Hefley *et al.* 1993)

The architecture proposed by Hefley and Murray (1993) does not meet all of the criteria specified in Table 4.3. It lacks directional representation for illustrating how the various components interact with each other and also the flow of data. The use of IAs is not explicit. A Knowledge Base Manager is mentioned however, no reference is made to a Knowledge Base component. There is also a lack in terms of describing the components of the model which represents the API. Therefore it can be assumed that the model does not have an API module.

Puerta (1993) proposes a model which is called L-CID and is a self-adaptive intelligent interface model. L-CID supports knowledge base interaction, self–adaptation and automatic generation (Puerta 1993). L-CID addresses the knowledge requirements and functionality that must be represented in an intelligent interface model. These requirements are the knowledge of the system, organisational environment and knowledge of how and when to assist the user (Puerta 1993).

L-CID is a multilayer model with the performance element of the learning layer being expanded into the lower interface layer. The L-CID model is represented by Figure 4.5.

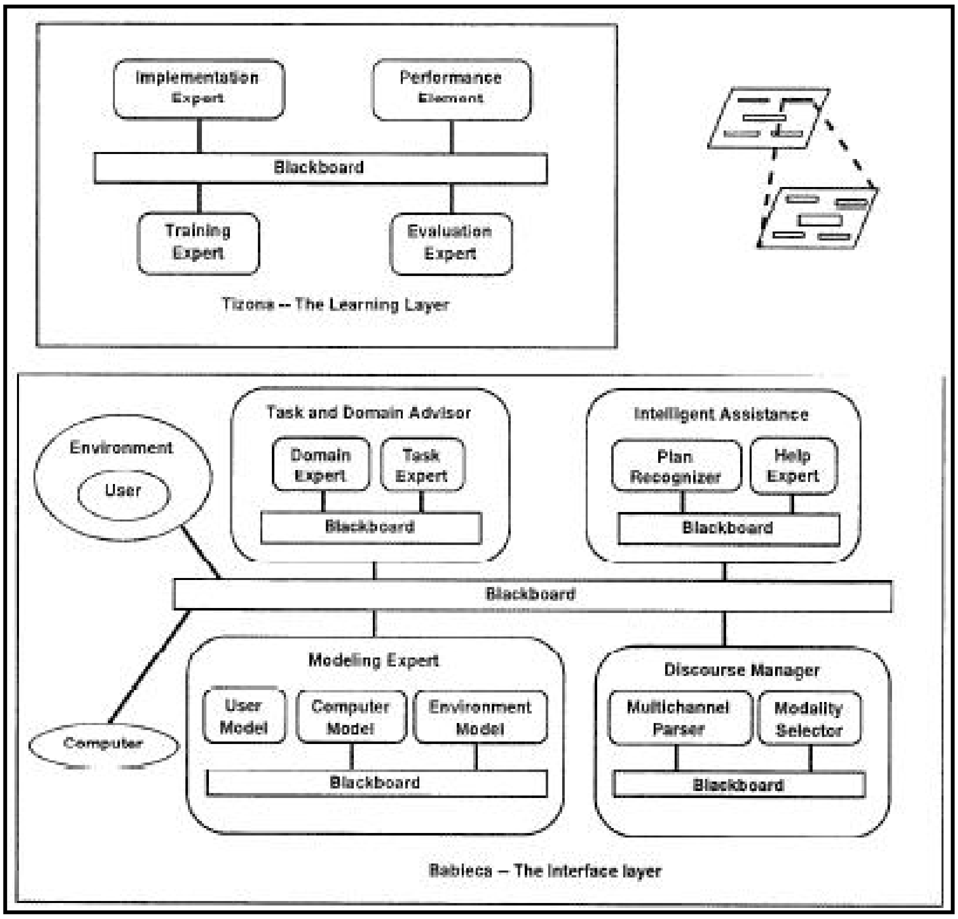


Figure 4.5: L - CID Model of a Self Adaptive Intelligent Interface (Puerta 1993)

The use of the Discourse Manger for Input Analysis and Output Generation is assumed due to it containing Multichannel Parsing and Modality Selector components. The presence of a component which would assume the role of the Dialog Controller is not present in the model. There is also no evidence to support that the model uses IAs in order to interact with the Modeling Expert Component.

The models discussed in this section are not specific to the CC domain. These models do provide the basic architecture for an IUI and a more specialised model could be derived based on these architectures to support the development of an IUI for CCs.

4.8 Conclusion

The aim of this chapter was to determine if any existing IUI models could be adapted to the domain of CCs. This objective was achieved by establishing a set of criteria (Table 4.3) and evaluating the various existing IUI models against these criteria.

Models were defined as a representation of a system or process and are required to illustrate the different views of a system or steps of a process (Section 4.2). The typical characteristics of a model and the factors which could affect the quality of a model were established (Section 4.3). Four types of models were identified, namely Application Models, Abstract Presentation Models, Concrete Presentation Models, and Task an Dialog models (Section 4.4). Models representing a system should consist of certain elements. These elements were identified as the Data/Class Elements, the Architecture Element, the Interface Element and the Component-Level Element (Section 4.5).

The development of a model for a system requires that the model go through various stages. These stages describe the development of the model from the clarification to the generalisation stage (Section 4.6). It was established that the key to developing a new model is to understand the problem and its domain (clarification). The next stage would be to stipulate how the new model would differ (differentiation) from exiting models that attempt to solve a similar problem. The last stage of the model's development involves generalisation. Generalising the model would allow the model to handle all aspects of the problem. Generalisation is best done by evaluating existing models.

An analysis of several existing IUI models (Section 4.7) revealed that none were specific to the domain of CCs. The IUI Architecture (Tyler *et al.* 1991) is the most comprehensive and generalised model as it meets all of the criteria stipulated in Table 4.3. This architecture is therefore selected to be the most suitable architecture to be specialised for the domain of CCs.

The objective of the next chapter (Chapter 5) is to discuss how the IUI Architecture (Tyler *et al.* 1991) can be specialised for the domain of CCs.

Chapter 5: Model Design

5.1 Introduction

The IUI Architecture, proposed by Tyler *et al.* (1991), was selected in the previous chapter (Chapter 4) as the most suitable model to be specialised to the domain of CCs. The aim of this chapter is to discuss how that architecture can be specialised for the domain of CCs. This will be done by discussing the current limitations of the existing architecture and how these limitations were addressed. Following this will be a discussion of the design of the proposed IUI model.

5.2 Current Limitations and Modifications

Specialising the selected architecture to the domain of CCs requires that its limitations be addressed. Modifications, both internal and external, are necessary to qualify the selected architecture for the domain of CCs and for it to be regarded as a complete model.

5.2.1 Limitations

The selected architecture (Figure 4.2) possesses some limitations. One of these limitations is that it cannot infer the user's goals based on the low-level commands entered by the user (Tyler *et al.* 1991). Secondly, the Knowledge Base component of the architecture suggests the use of various models, however the structure and design of these models is not supplied. Although the architecture is for IUIs, a template for prototyping IUIs is not provided.

A model needs to represent all of the views of a system (Section 4.2). The IUI Architecture (Tyler *et al.* 1991) only represents the Architectural Design element and cannot be considered as a complete model.

5.2.2 Modifications

The limitation of being unable to infer the user's goals based on low-level commands provided was addressed. This was done by allowing the Plan Manager direct access to the Knowledge Base (Figure 5.1). The Plan Manager will retrieve the Task Model from the Knowledge Base. Depending on the current step and field with which the user is working, the Plan Manager will have a sense of what is expected from the user and be able compare that to what is provided by user (Section 5.3.1.2). Changes to the selected architecture (Figure 4.2) are shaded green in Figure 5.1.

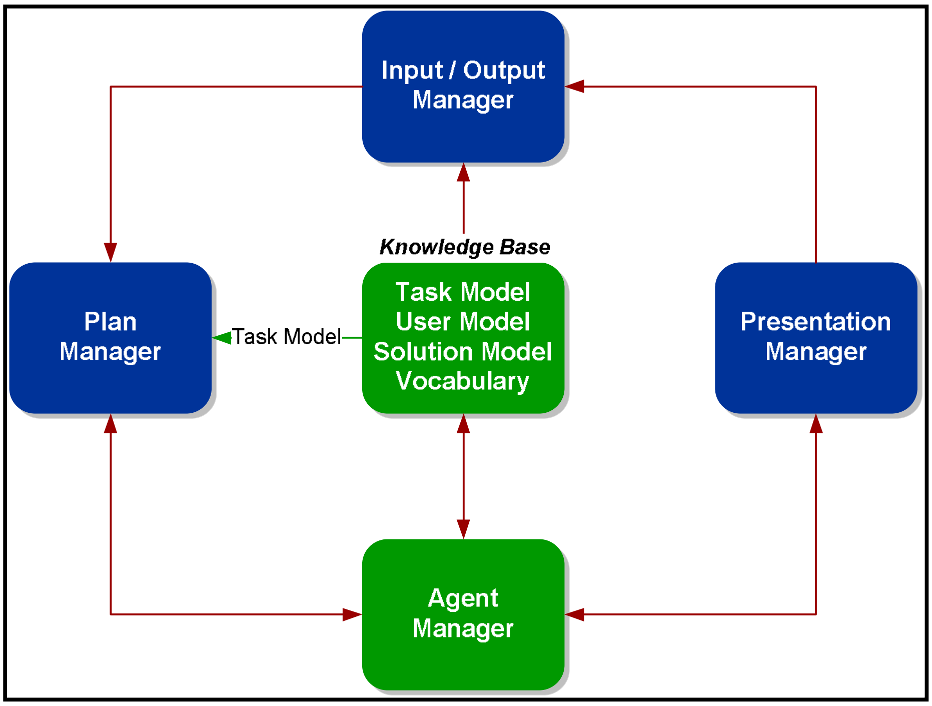


Figure 5.1: Modified IUI Architecture

Changes to the Knowledge Base component were made. A Solution Model was added to facilitate and support query resolution at the first-level of contact within a CC. Solution Models are discussed in Section 5.3.2.3. The last change that was made to the architecture was the renaming of the Adaptor component. The role of Adaptor is to adapt the interface to the needs of individual users' (Tyler *et al.* 1991). This was done by accessing and updating the various models in the Knowledge Base. The Adaptor is also responsible for retrieving information from the Knowledge Base. Due to its autonomous nature in providing an interface that is intelligent and adaptive, the Adaptor was renamed to the Agent Manager. The autonomous nature of this component requires the use of IAs.

Additions to the selected architecture include specifying the Component-Level Design of the models stored within the Knowledge Base. Section 5.3.2 of this chapter discusses Component-Level Design. The objective of this section is to present a discussion on the structure of the various models stored within the Knowledge Base. These structures are currently lacking from the architecture specification. Interface design in the form of an IUI template was also added. The template for constructing IUIs is presented and discussed in Section 5.3.3. Use of this template will enable developers to build IUIs.

5.3 Proposed Model

The objective of this research was to propose an IUI model for CCs that could assist in potentially improving CRR. This section discusses the design of that model. Four types of models were presented in Section 4.4. The proposed model for this research will be a combination of the Application Model, Abstract Presentation Model and Task Model. The proposed IUI model will consist of an Interface Design element that will allow for information exchange to take place (Application Model). A template illustrating how an IUI should be structured is provided (Abstract Presentation Model). Task models are part of the Component-Level Design element of the proposed IUI model and are stored in a Knowledge Base. This Knowledge Base is a component of the Architectural Design element of the proposed IUI model. The proposed IUI model therefore consists of three major elements, namely an Architectural element (Section 5.3.1), Component-Level element (Section 5.3.2) and an Interface element (Section 5.3.3). These elements are illustrated in Figure 5.2.

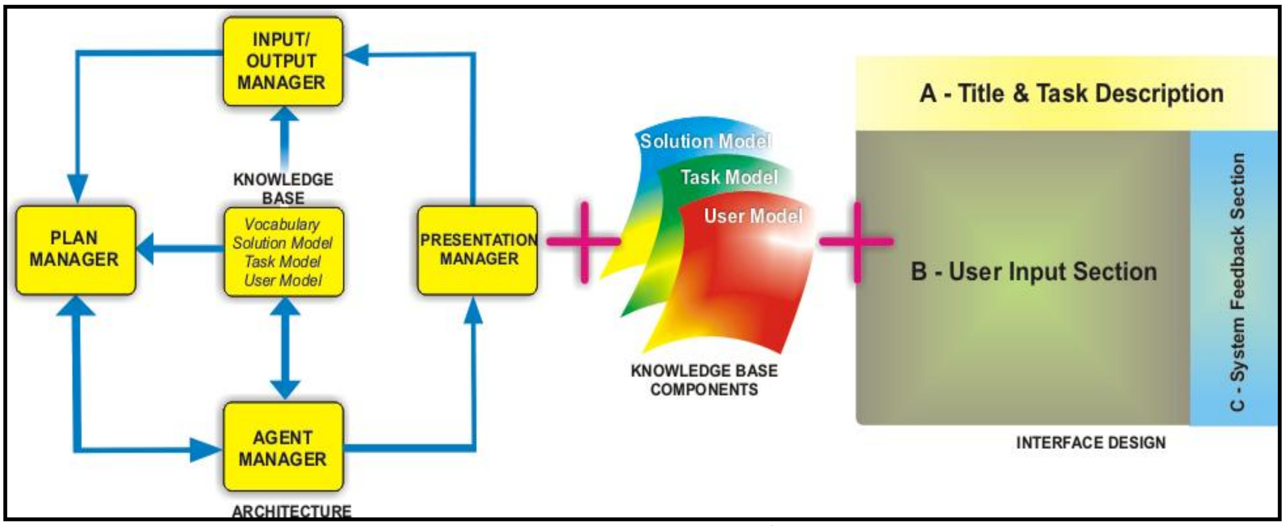


Figure 5.2: Proposed IUI Model for CCs

5.3.1 Architecture

The architectire of the proposed IUI model consists of five components. These components are: the Input/Output Manager (Section 5.3.1.1); the Plan Manager (Section 5.3.1.2); the Agent Manager (Section 5.3.1.3); the Knowledge Base (Section 5.3.1.4); and the Presentation Manager (Section 5.3.1.5). Figure 5.3 illustrates these components.

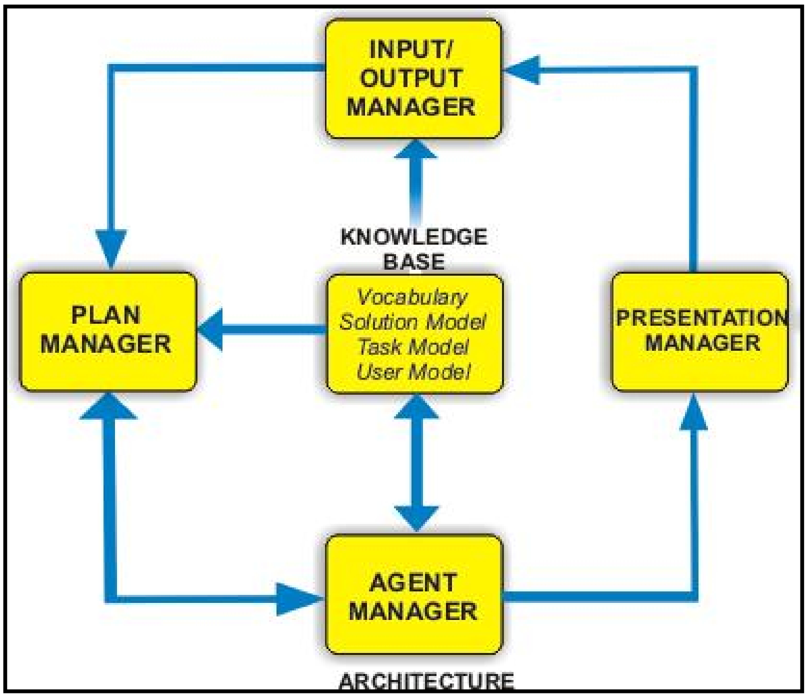


Figure 5.3: Architecture of the Proposed IUI Model

5.3.1.1 Input/Output Manager

The first component of the architecture is the Input/Output Manager. Multimodal user input and output is accepted and delivered through this component. Its purpose is to take multimodal input and translate it into a representation that will allow the interface to make inferences about the user's actions (Tyler *et al.* 1991). Translated input is sent to the Plan Manager. Output from this component is in the form of low-level operations that are translated from high-level commands. These high-level commands are received from the Presentation Manager (Tyler *et al.* 1991).

Providing multimodal capabilities to CCAs will enable them to interact naturally with the interface. Interaction could take the form of natural language processing, gesture recognition and speech recognition. Through the use of this component the CCA's interaction with the interface is customisable and personalised.

5.3.1.2 Plan Manager

The function of the Plan Manager is to assist the user in achieving his/her high-level goals with the greatest amount of satisfaction and efficiency (Tyler *et al.* 1991). It operates based on the current task/step of the user. Through the Plan Manager the IUI will analyse low-level commands received from the Input/Output Manager in order to:

- Automatically fill default parameters and execute system commands;
- Indicate the current state and progress of a step or task; and
- Detect and correct any possible errors in the user's plans.

These processes are achieved by comparing the low-level commands to the task's parameter values. Task parameter values are in the form of high- and low-level goals. These goals are stored within the Task Model (Section 5.3.2.1). Retrieval of the Task Model, from the Knowledge Base, is done at the beginning of a task by the Plan Manager (Section 5.2.2). Low-level commands are sent from the Plan Manager to the Agent Manager in order to satisfy the user's request for data. The Plan Manager performs the functions of the plan recognition component of an IUI (Section 3.3.3).

Including the Plan Manager as part of the Architecture could facilitate in a reduction in the time taken to log and resolve customer queries. It could do this by guiding CCAs to successfully log and resolve customer queries. Use of the Plan Manager could also reduce the number of steps required to log and resolve customer queries, by automating some of these steps. These steps could include the automatic categorisation of customer queries, identification of the customer and providing possible solutions based on the categorisation of the query.

5.3.1.3 Agent Manager

The role of the Agent Manager is to receive interface events (low-level commands entered by the user) and validate them against the various models stored within the Knowledge Base (Tyler *et al.* 1991). The Agent Manager then either updates an existing model or retrieves the required results, from the Knowledge Base, and sends these results to the Presentation Manager. An integral aspect of the Agent Manager is its ability to interact with the User Model, based on the current step of a task (Section 5.3.2.2). This interaction allows the Agent Manager to keep track of various performance data. Performance data in the form of how long it takes the user to complete a task and how many errors have been committed by the user. The Agent Manager is also responsible for performing the error checking required by the Plan Manager and delivering the required information for the automation of steps. Tasks performed by the Agent Manager are done through the use of IAs.

Application of the Agent Manager to the domain of CCs would allow for IAs to retrieve information dynamically, from the Knowledge Base, as the user provides input. This assists the Plan Manager in reducing the time in which it takes to find relevant information for the purposes of processing and resolving a customer's query.

5.3.1.4 Knowledge Base

The Knowledge Base is a key component of an IUI. It serves as the repository for domain and application knowledge, knowledge about the user, instructional and communicational knowledge (Tyler *et al.* 1991). This knowledge is necessary to allow the interface to make decisions using intelligent techniques. These intelligent techniques determine how to adapt the interface and deliver the appropriate information, to meet the current needs of the user.

Specialising the model for CCs requires that the knowledge residing within the Knowledge Base be specific to the domain of CCs. Data residing within the Knowledge Base would need to represent:

- Customer information;
- Query categorisation and classification information;
- Query resolution information;
- Task models relating to the tasks of logging and resolving customer queries;
- Solutions models which attempt to resolve customer queries; and
- User models of the CCAs using the application.

The structure of these various models is discussed in Section 5.3.2.

5.3.1.5 Presentation Manager

The last component of the proposed model's architecture is the Presentation Manager. The role of the Presentation Manager is to retrieve results from the Agent Manager and, based on these results, determine the most appropriate modality in which these results could be presented. The Presentation Manager would play an important role in terms of delivering an adaptive interface to the user. It would not only determine the various ways in which a set of results or an interface could be presented to the user but also allow for users to customise output according to their own specifications. This customisation is then referenced and applied when delivering similar results for the same user.

The benefits that the Presentation Manager could present to the domain of CCs could be in the form of delivering an adaptive interface. An adaptive interface would allow for a more personalised interaction seeing that different users interact with an interface in different ways.

5.3.2 Component-Level Design

The objective of this section is to discuss the structure of the various models stored within the Knowledge Base. These models are the Task Model (Section 5.3.2.1), the User Model (Section 5.3.2.2) and the Solution Model (Section 5.3.2.3). Component-Level design represents the second element of the proposed model (Figure 5.2). The structures of these models are generic, implying that they could be used by existing IUI applications or integrated into the design of new IUI applications.

5.3.2.1 Task Model

Using a Task Model to explicitly store the high- and low-level goals of a task will allow the interface to deliver task-based descriptions of the steps and sub-steps that need to be completed. The Task Model will contain several entries, one per task. These entries will be a representation of the tasks typically performed by a CCA in order to diagnose and resolve customer queries. Figure 5.4 is an extract of the Task Model Schema for the Log Call task.

The Task Model contains the name of the task and a description of the task. This task description represents the high-level goals of the task. Low-level goals are represented by the steps that need to be completed for that task. Each step consists of a description and a set of fields that need to be completed in order for the task to be complete. Each field is accompanied by a name, description, type and the maximum number of input characters for that field. This information can be used by the Agent Manager when determining whether the user has provided the correct information.

```
<Task>
  <Name>Log Call</Name>
  <Description>Log call placed by customer</Description>
- <Steps>
  - <Step>
      <Description>Capture customer's details</Description>
    - <Fields>
      - <Field>
          <Name>UserID</Name>
          <Description>UserID of the customer</Description>
          <Fieldtype>string</Fieldtype>
          <Size>20</Size>
        </Field>
      - <Field>
          <Name>First Name</Name>
          <Description>First name of the customer</Description>
          <Fieldtype>string</Fieldtype>
          <Size>20</Size>
        </Field>
      + <Field>
      </Fields>
    </Step>
  + <Step>
  + <Step>
  </Steps>
</Task>
```

Figure 5.4: Extract Task Model Schema

5.3.2.2 User Model

User models are essential when contributing towards the intelligence of an IUI. They allow for the system to adapt to the needs of the user. A User Model will be stored in the Knowledge Base and an entry will be present for each user. The model will contain a profile of the user's interaction with the interface. This profile attempts to keep track of the user's progress in terms of task completion. Keeping this information will provide an indication of whether using an IUI reduces the time taken to diagnose and resolve customer queries. A list of categories for the problems solved by a particular user will also be present in the model. Having this list will provide an indication of the user's expertise when it comes to solving queries. This could be used when determining the most suitable CCA to resolve a particular query. The details of the user are also stored within the Knowledge Base. Figure 5.5 is an extract of a User Model Schema.

The performance data discussed in Section 5.3.1.3 includes keeping track of:

- The average time in which a user completes a given task;
- The total number of errors the user made when attempting a particular task at a given time;
- The number of keystrokes in which the user completed a task; and
- The last time the user completed a particular type of task.

This information could also be used when performing a usability evaluation in order to determine the usefulness of the application.

```
<User>
+ <Particulars>
+ <Login>
- <Queries>
  - <Category>
      <CatName>Name of Category</CatName>
    - <IncidentType>
        <InName>Incident Type Name</InName>
      - <SubCallType>
          <SubName>Sub Call Type Name</SubName>
          <Dates>log of dates for this type of query</Dates>
        </SubCallType>
      </IncidentType>
    </Category>
  </Queries>
- <Performance>
  - <Tasks>
    - <Task>
        <Name>Name of the task</Name>
        <AvTime>Average time to complete the task</AvTime>
        <KeyStroke>Number of Keystrokes to compelte the task</KeyStroke>
        <LastPerformed>When the task was last performed</LastPerformed>
      - <Errors>
          <Date>Date at which task was performed</Date>
          <NoErrors>Number of errors user committed on the above date</NoErrors>
        </Errors>
        <NoCompleted>Number of times the task was completed</NoCompleted>
      + <steps>
      </Task>
    </Tasks>
  </Performance>
  <DateCreated>Date the user model was created</DateCreated>
  <DateLastModified>Date the user model was last modified</DateLastModified>
  <ApplicationPref>User's Application Preferences goes here</ApplicationPref>
</User>
```

Figure 5.5: Extract of User Model Schema

5.3.2.3 Solution Model

The Solution Model was not part of the original IUI architecture (Figure 4.2) and was added to improve CRR. Figure 5.6 is an extract of the Solution Model Schema. Based on the keywords obtained from the description of the customer's query, the user will be presented with a list of possible solutions. These keywords will be matched against keywords stored in the Knowledge Base. The comparisons of keywords and identification of possible solutions will be performed by the Agent Manager. A Solution Model will need to contain:

- The name of the solution;
- A brief description of the solution;
- The problem category associated with the solution; and
- A list of questions and answer that can be used to address the query.

Providing a Solution Model should facilitate faster problem solving and reduce the time taken to escalate the query or dispatch a technician to solve the customer's query.

```
<Solution>
  <Name>The name of the solution</Name>
  <Description>A short description of the solution</Description>
  <Category>The category under which the solution falls</Category>
  <CallType>The call type for the solution</CallType>
  <SubCallType />
- <Steps>
    <Q>Question to ask the user</Q>
    <A>Answer for the above question</A>
    <Q>Question to ask the user</Q>
    <A>Answer for the above question</A>
    <Q>Question to ask the user</Q>
    <A>Answer for the above question</A>
    <Q>Question to ask the user</Q>
    <A>Answer for the above question</A>
  </Steps>
</Solution>
```

Figure 5.6: Extract of Solution Model Schema

5.3.3 Interface Design

An important challenge faced by IUIs is that they must improve the interaction with the user (Hefley and Murray 1993; Höök 2000). An IUI for CCs would primarily be used for the purposes of information filtering and information presentation in order to improve call resolution. CCAs are currently exposed to the tasks of (Section 2.6):

- Identifying and confirming a customer's particulars;
- Diagnosing the customer's query; and
- Offering a possible solution based on the diagnosis of the query.

A key feature of information filtering would be to assist the CCA to find the most relevant information to complete the tasks in the shortest time possible (Hefley and Murray 1993). The manner in which this information is displayed is very important. This section discusses the last element of the proposed model, namely the Interface Design element.

Reducing CRR requires that the most relevant information be displayed in the shortest amount of time. Successful validation of user input by the Agent Manager will result in finding this information. The section of the interface dedicated to the output generated by the Agent Manager needs to be separate from the direct manipulation section of the interface (Lieberman 1997). The Agent Manager needs to observe the actions performed by the user in the direct manipulation section and present feedback in an unobtrusive way in its own section. An IUI template supporting this is illustrated in Figure 5.7.

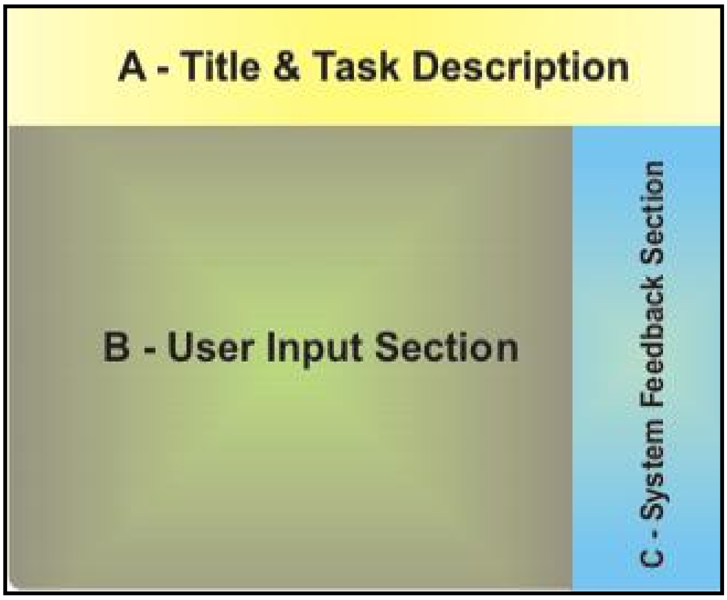


Figure 5.7: Low-Fidelity IUI Design Template

The low-fidelity IUI design template in Figure 5.7 could be used to design IUIs. Section A in Figure 5.7 will display the high-level goals of the current task and the steps (low-level goals) required to be completed by the user. This information is retrieved from the Task Model and provided by the Plan Manager.

Section B represents the direct manipulation section. This is the section in which user input is provided. Input received from this section is translated by the Input/Output manager and then sent to the Plan Manager. The Plan Manager through the use of the Agent Manager analyses the input to check for errors and provides candidate results.

These candidate results will be presented in Section C. Section C is responsible for displaying the feedback generated by the Agent Manager. This feedback will be in the form of information on demand to help reduce the time in which information is found.

5.4 Conclusion

The objective of this chapter was to describe the design of an IUI model for the domain of CCs. This was done by discussing the design of the elements of the proposed IUI model.

In order to propose a model for the domain of CCs an existing architecture was identified in Chapter 4 and specialised. This specialisation was done in terms of addressing the current limitations (Section 5.2.2) of the selected architecture. Modifications to the architecture included allowing the Plan Manager to access the Knowledge Base directly to acquire the Task Model, renaming the Adaptor to the Agent Manager because of its autonomous nature, and the inclusion of a Solution Model to the Knowledge Base to deliver query resolution capabilities at the first-level of support. Specialising the architecture to the domain of CCs required that knowledge stored within the Knowledge Base (Section 5.3.1.4) be representative of the domain of CCs.

In order for the architecture to be considered as a complete model a Component-Level Design (Section 5.3.2) element and an Interface Design (Section 5.3.3) element ware added. The proposed model consists of three elements: an Architectural Design element (Section 5.3.1), a Component-Level Design element (Section 5.3.2), and an Interface Design element (Section 5.3.3).

A need existed to define a formal structure for the various models residing within the Knowledge Base. The Component-Level Design (Section 5.3.2) described the structure of these models. The structures of the Task Model (Section 5.3.2.1) and User Model (Section 5.3.2.2) are general and can be applied to existing IUI applications or could be used to aid the development of new IUI applications. A Solution Model (Section 5.3.2.3) was added in order to support query resolution at the first-level of support. This Model was not present in the original architecture. The Solution Model will contain a description of the possible solution and a list of questions and answers that could be used to address the customer’s query.

The design of a low-fidelity IUI template was proposed (Section 5.3.3). Use of this template would allow developers to construct IUIs. The template included sections for providing task-based information, user input and intelligent feedback in the form of system feedback. Intelligent feedback presented in the System Feedback Section would be a list of results dynamically generated by the Agent Manager based on the user's input.

The next chapter (Chapter 6) discusses how the proposed IUI model can be implemented. This implementation shall serve as a proof of concept to determine whether the proposed IUI model is capable of allowing developers to build IUIs for CC applications.

Chapter 6: Implementation

6.1 Introduction

The objective of this chapter is to discuss and demonstrate the effectiveness of the proposed IUI model (Chapter 5) through the implementation of an IUI prototype as a proof of concept (Olivier 2006). The implementation of the IUI prototype will be discussed in terms of the implementation of the architecture of the proposed IUI model (Section 6.2), the components of the Knowledge Base (Section 6.2.3) and the user interface (Section 6.3). The IUI prototype will provide the capabilities required to log, diagnose and resolve a customer's query, retrieve task-based information from the Task Model and deliver dynamic feedback through the use of an IA. This will support the provision of plan recognition and the delivery of intelligent feedback to potentially improce CRR.

6.2 Architecture

The Plan Manager, Agent Manager and Knowledge Base components directly support the query diagnosis and resolution process of CCAs and were thus implemented. The Input/Output Manager and the Presentation Manager were not implemented as these components are primarily concerned with the processing and generation of multimodal input and output and with providing the interface with adaptive capabilities.

Implementation of the components of architecture of the proposed IUI model was achieved by implementing each component as a separate class. The various classes were implemented in Microsoft C# using Microsoft Visual Studio 2005 as the integrated development environment (IDE). Figure 6.1 contains an activity diagram which illustrates the processes that occur between the User (green), the System (orange), and the related IUI components (blue), namely the Plan Manager and the Agent Manager.

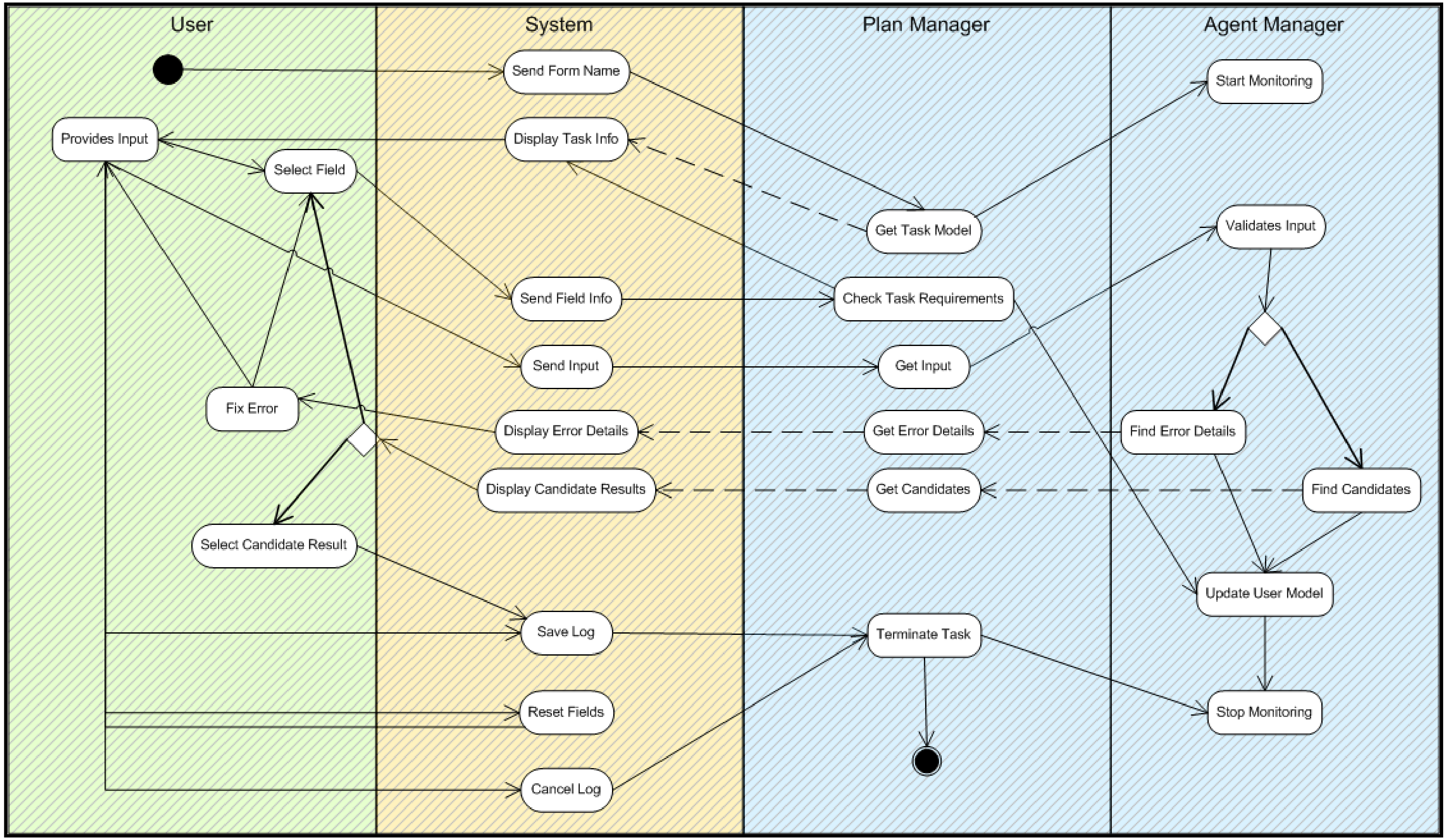


Figure 6.1: Activity Diagram of the IUI Prototype

An understanding of the roles played by the Plan Manager, the Agent Manager and the flow of data between the User and these components is communicated through the various activities illustrated in Figure 6.1.

6.2.1 Plan Manager

The role of the Plan Manager should be an advisory one in order to assist the user to attain the high-level goals of a task (Section 5.3.1.2). This was achieved by allowing the Plan Manager to access the Knowledge Base, retrieve the Task Model and deliver task-based information to the system. The role of the Plan Manager (Figure 6.1) is to retrieve the Task Model and to accept input from the system and deliver it to the Agent Manager. Feedback generated by the Agent Manager is delivered to the system via the Plan Manager. A list of attributes and methods of the Plan Manager Class is illustrated in Figure 6.2.

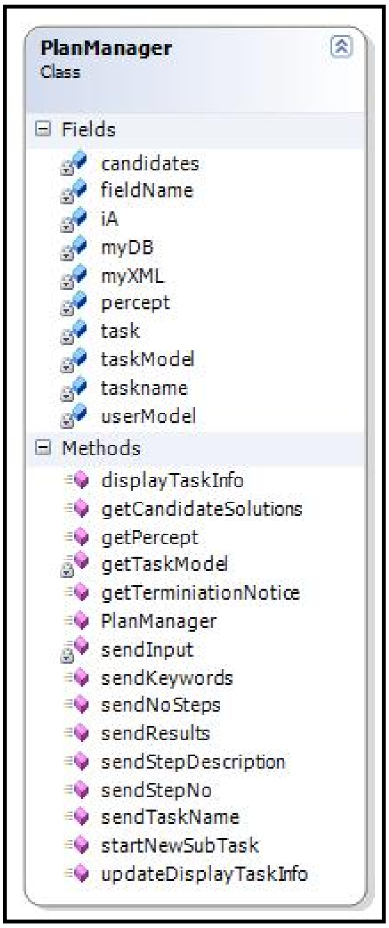


Figure 6.2: Plan Manager Class

The attributes shown in Figure 6.2 are responsible for:

- Storing the Task Model (*task*) and the current input (*percept*) provided by the user; and
- Accessing the Agent Manager (*IA*) and Knowledge Base (*myDB*) components.

Acquisition and storing of the Task Model is made possible through the use of the *getTaskModel()* method. This method performs the first activity for the Plan Manager as illustrated in Figure 6.1. The Task Model is retrieved from the Knowledge Base. Other auxiliary methods, such as *displayTaskInfo(); upaateDisplayTaskInfo(); sendTaskName()*; *sendNoSteps()*; *sendStepDescription()* and *sendStepNo(),* deliver task-based information from the Task Model to the system. Task-based information (in the form of the name of the current task, the description of the current step, the total number of steps required to complete a task and the number of the current step) is provided to the system. This assists in allowing the Plan Manager to fulfil its role as an advisor.

Input received from the system occurs through the use of the *getPercept(percept, fieldname)* method. This method retrieves the input from the system along with the name of the field in which the input was provided. The third activity of the Plan Manager in Figure 6.1 represents this action.

Sending the percept to the Agent Manger occurs when the *sendInput(percept,fieldname)* method is called by the Plan Manager. The *sendResults()* method retrieves feedback generated by the Agent Manager and delivers it to the system. This feedback can be in the form of candidate results *(candidates)* or error details resulting from validation. The fifth activity of the Plan Manager in Figure 6.1 represents this action.

6.2.2 Agent Manager

The Agent Manager accepts interface events from the Plan Manager. Dynamic feedback in the form of providing error reporting or generating candidate results, is delivered to the Plan Manager by the Agent Manager (Section 5.3.1.3).

Figure 6.1 illustrates how the activities of the Agent Manager component contribute towards either delivering candidate results or error feedback. The monitoring activities (Figure 6.1) support the logging of the user's performance data (Section 5.3.1). This can be used to determine whether the use of an IUI is reducing the time taken to diagnose and resolve customer queries. A list of attributes and methods of the Agent Manager Class is illustrated in Figure 6.3.

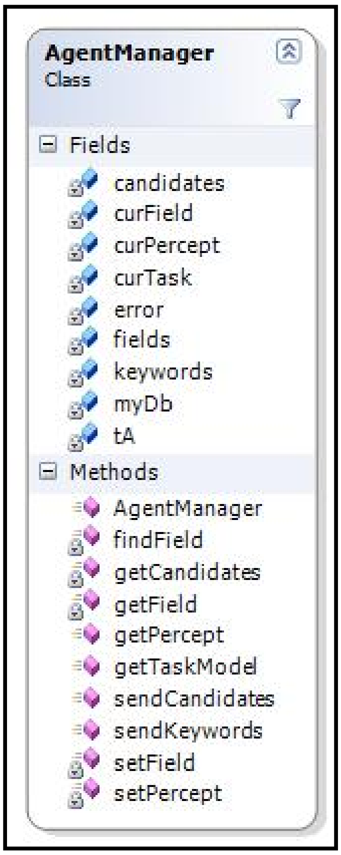


Figure 6.3: Agent Manager Class

The attributes shown in Figure 6.3 are responsible for:

- Storing the current input (*percept*) provided by the Plan Manager, the current field (*curField*) and a list of candidate results (*candidates*); and
- Accessing the Knowledge Base (*myDB*) component.

Interface events (*percepts*) received from the Plan Manager are received in the form of input. The name of the field from which the percept originated is also provided.

The *Get Input* activity (Figure 6.1) illustrates this process. Once the Agent Manager has acquired this data, it retrieves the relevant field information from the Task Model. This is done in order to verify the name of the field and the type of data required by that particular field. The percept is validated in order to determine its validity and to check for errors. Delivering dynamic error-reporting capabilities was implemented by making use of a Validation Class. This class was implemented as a static class so that it need not be instantiated, and its public methods are accessible from any point within the prototype. The Validation Class is illustrated in Figure 6.4.

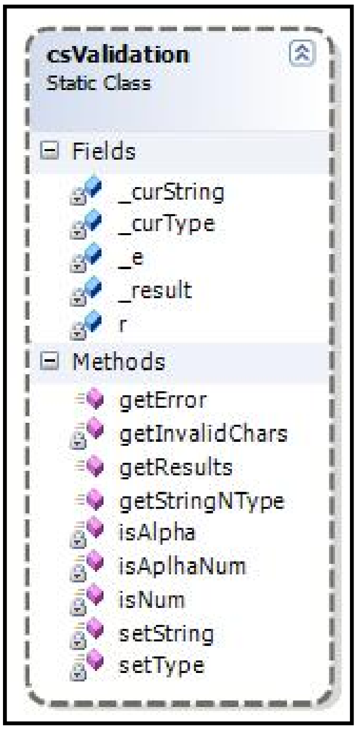


Figure 6.4: Validation Class

The percept together with its specified type is sent from the Agent Manager to the Validation Class. Validation occurs by matching the percept according to its specified type. This is done through the use of Regular Expressions. The methods, *isAplha(), isNum()* and *isAlphaNum()*, make use of a pattern that determines whether the input contains only alphabetic, numeric, or alpha-numeric characters. Figure 6.5 is an extract of the Validation Class and shows how input is matched to a Regular Expression pattern.

```
private static void isAlpha()
{
 //Regular expression pattern to match alphabetic characters.
 r = new Regex(@"[^a-z\s+]", RegexOptions.IgnoreCase);
 result = r.IsMatch(_curString);

 if (_result)
  {
    _e.Name = "Invalid Character Entry!";
    _e.Cause= "You have entered a value(s) that is not an alphabet!";
    _e.Resolution = "Please change the value(s) listed below in the
respective position(s) to only those that are from a-z or A-Z";
     getInvalidChars(r);
  }
}
```

Figure 6.5: Extract of Validation Class (isAlpha() Method)

The results of the pattern-marching are then sent to the Agent Manager. Only if validation is successful will candidate results be found. If the validation is unsuccessful, an error object will be sent to the Plan Manager. The error object contains the name of the error, the cause of the error, a description of how to fix it and a list containing the positions and error values in the given input. Validating input is represented by the second and third activity in Figure 6.1 in the Agent Manager section.

Provided that the validation is successful, generating a list of candidate results occurs when the Agent Manager calls the *getCandidates()* method. These results are then sent back to the Plan Manager when the *sendCandidates()* method is called. The *Find Candidates* activity is responsible for finding the candidates and sending them to the Plan Manager (Figure 6.1). Retrieving candidate results from the Knowledge Base relies on the input as well as the name of the field in which the input was provided. The Knowledge Base is queried using standard Structured Query Language (SQL) queries. Once obtained, the results are stored in a list and sent back to the Plan Manager.

The first activity of the Agent Manager (Figure 6.1) is to start monitoring the user and this is complemented by the last activity which stops monitoring the user. Monitoring the user's actions is implemented in order to establish the type of queries that the user has previously solved. This is necessary in order to determine the most qualified available

CCA that can handle the customer's query. Monitoring the user's actions could further assist in determining whether the use of an IUI actually does reduce the time taken to diagnose and resolve a customer's query. Indicators, such as the number of keystrokes, number of errors, and the total time in which it takes to complete a step and a task, are captured and stored in the User Model. This performance data could be used for usability purposes to determine how long a user took to complete a specific step for a particular task. The query categorisation details and the performance indicators are stored within the User Model which resides in the Knowledge Base.

6.2.3 Knowledge Base

Implementation of the Knowledge Base was done using a combination of Microsoft SQL Server 2005 with the various models (Task, User and Solution) being implemented in Extensible Markup Language (XML). Data used to create the Knowledge Base was extracted from the Heat Database used by the NMMU ICT Service Desk. A snapshot of the Heat Database was obtained.

The Task, User and Solution models are stored within the Knowledge Base. In order for the Knowledge Base to validate these models upon insertion or modification, the XML Schemas for these models are also stored within the Knowledge Base. Other domain-related data and vocabulary, which are used for call query diagnosis and classification, are also stored within the Knowledge Base.

Access to the Knowledge Base and the data residing within it is achieved by the use of a class. This Knowledge Base Class is accessed by the Plan Manager and the Agent Manager. The Knowledge Base Class is illustrated in Figure 6.6.

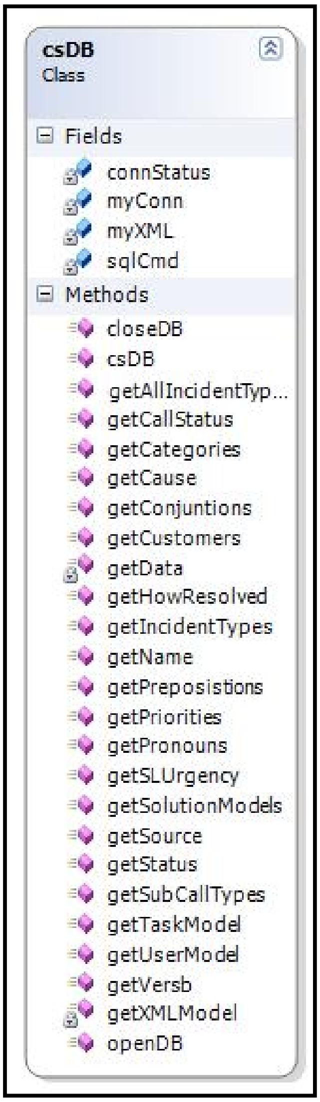


Figure 6.6: Knowledge Base Class

Figure 6.6 lists various methods that are used for retrieving domain information and the Task, User and Solution models. A list of candidates is derived from these methods by the Agent Manager. The Task, User and Solution models are retrieved by calling the *getTaskModel()*, *getUserModel()* and *getSolutionModels()* methods.

The *getXMLModel()* method is responsible for calling the appropriate stored procedure and returning the required model in XML format. These models, in XML format, are then parsed and instantiated into Task, User or Solution objects. The reason for this is due to the structured nature of a XML document. Instantiating the XML document into an object eliminates having to parse the XML document from the root node until a desired node is found each time a Task, User or Solution Model is queried. The object version of the XML document reduces access time and provides random access to data. Updating the model is done in a similar fashion.

6.2.3.1 The Task Model

The Task Model stored within the Knowledge Base is acquired by the Plan Manager (Figure 6.1). The Plan Manager uses the Task Model to deliver high- and low-level goals of the task to the user via the interface. These high- and low-level goals are represented by the description of the task (high-level) and the descriptions of the various steps (low-level) that need to be completed. The methods used by the Plan Manager to acquire these goals are illustrated in Figure 6.7.

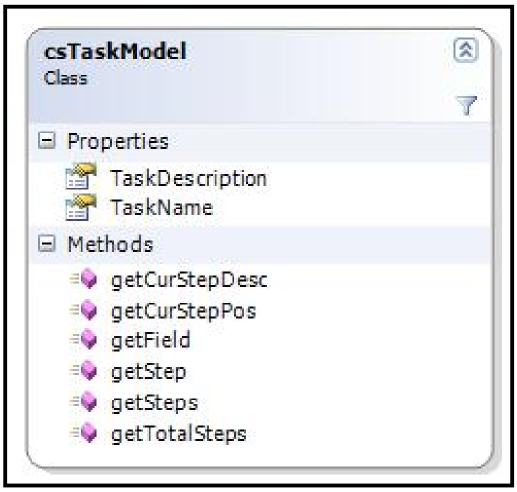


Figure 6.7: Task Model Methods used by the Plan Manager

The Task Model is used by the Agent Manager to assist with input validation. The field type, along with the current input, is sent to the Validation Class in order to determine the relevance of the input provided by the user. The *getField()* method (Figure 6.7) provides the Agent Manager with the required information.

6.2.3.2 The User Model

The User Model stored within the Knowledge Base contains representations of the user's ability to diagnose and resolve a customer's query (Section 5.3.2.2). The User Model was divided into two sections:

- The user's details; and
- The user's profile.

The details of the user are recorded as attributes in a table residing in the Knowledge Base. Each user also has an attribute that stores the profile of that user. This profile keeps tracks of the user's progress in terms of completing the required task. The Agent Manager is responsible for acquiring and updating the User Model (Figure 6.1).

The recording of the performance data serves to identify which user is most qualified to resolve a particular query. It also provides an indication in terms of determining whether the IUI is reducing the time taken by the user to diagnose and resolve customer queries.

6.2.3.3 The Solution Model

Implementation of the Solution Model was done in XML, similar to the User and Task models. The intention behind having a Solution Model was to provide possible call resolution at the first level of support.

6.3 The User Interface

A UI was implemented in order to visually meet the requirements of the proposed IUI model (Chapter 5). The various components of the architecture are encapsulated in the form of an IUI (Section 5.3.3). Two windows were implemented, based on the low-fidelity IUI design template proposed in Chapter 5. The first window supports the logging and diagnosing of customer queries and the second window supports the resolution of those queries.

6.3.1 The Query Diagnosis Window

Figure 6.8 contains a prototype of an IUI for the process of logging and diagnosing a query at a CC. The layout of this window was designed according to the low-fidelity IUI design template (Figure 5.7). It is divided into three sections, namely A, B and C. Section A contains Task Status Information (Section 6.3.1.1). User input is provided in Section B (Section 6.3.1.2). This allows for a customer to be identified and for his/her query to be logged. Section C contains the System Feedback Section which is responsible for delivering dynamic intelligent feedback (Section 6.3.1.3).

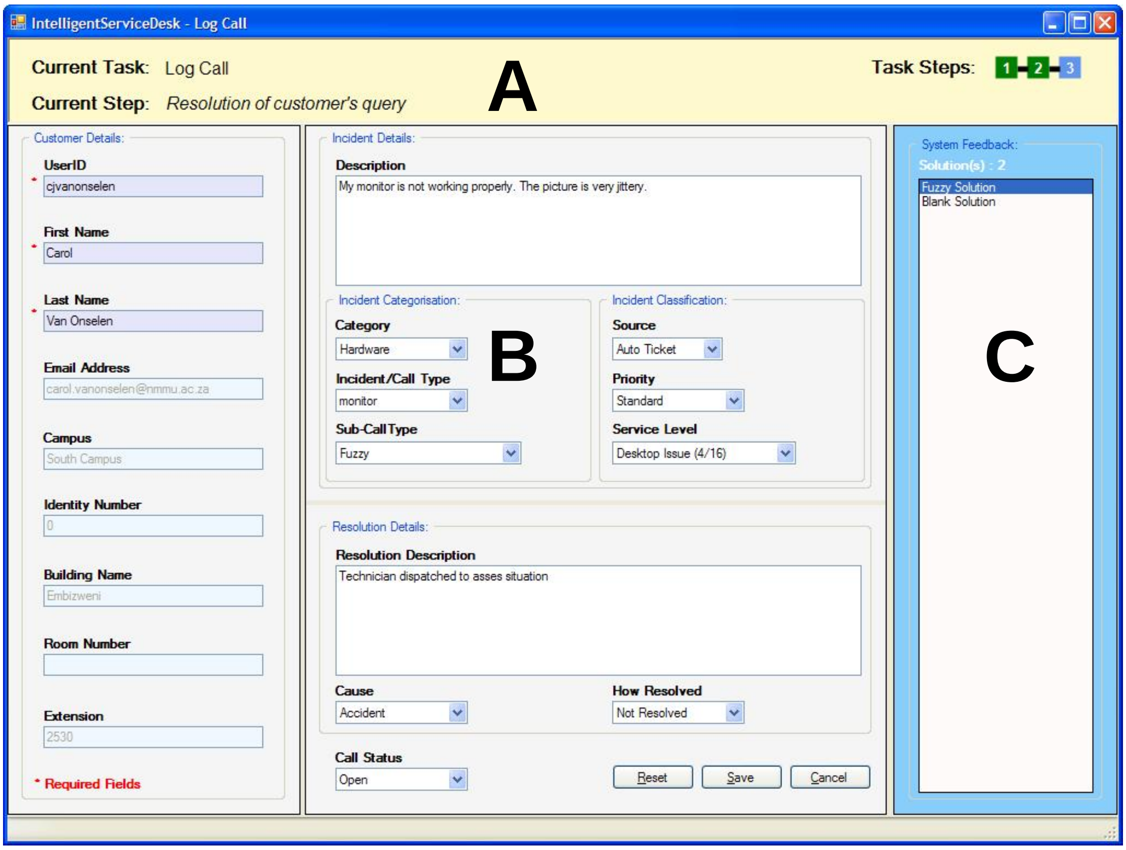


Figure 6.8: Query Diagnosis Window

6.3.1.1 Task Status Section

Section A (Figure 6.8) contains the Task Status Section. It contains information relating to the current task. This section can be mapped onto Section A in the low-fidelity IUI design template (Figure 5.7).

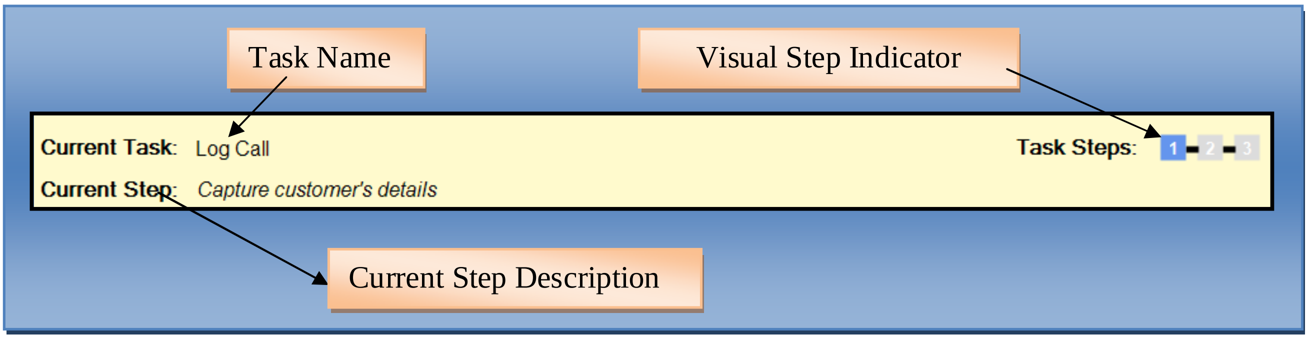


Figure 6.9: Task Status Information Section

Information residing within the Task Model (Section 6.2.3.1) is displayed in this section of the interface. The name of the current task, a description of the current step and the total number of steps that need to be completed is extracted from the Task Model and displayed (Figure 6.9). The Current Step description changes as the user focus on a different section of the interface. The total number of steps is calculated by counting the number of steps, for a particular task, within the Task Model.

The Visual Step Indicator represents these steps and their completion status through the use of colour. This allows the user to see which step is currently being performed as well as the status of the other steps. Figure 6.10 indicates the range of colours and their meanings that could be displayed by the Visual Step Indicator.

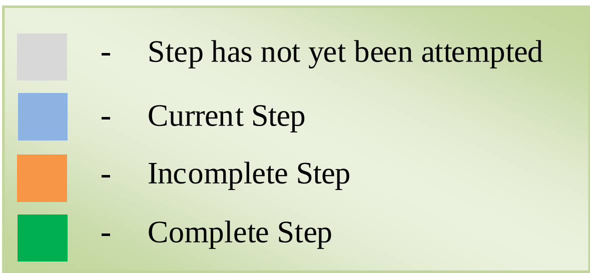


Figure 6.10: Visual Step Indicator Colour Chart

6.3.1.2 User Input Section

Section B (Figure 6.8) is the User Input Section. This section can be mapped onto the direct manipulation section in the low-fidelity IUI design template(Figure 5.7). Input is provided in this section is in the form of customer details, incident details and resolution details (Figure 6.11).

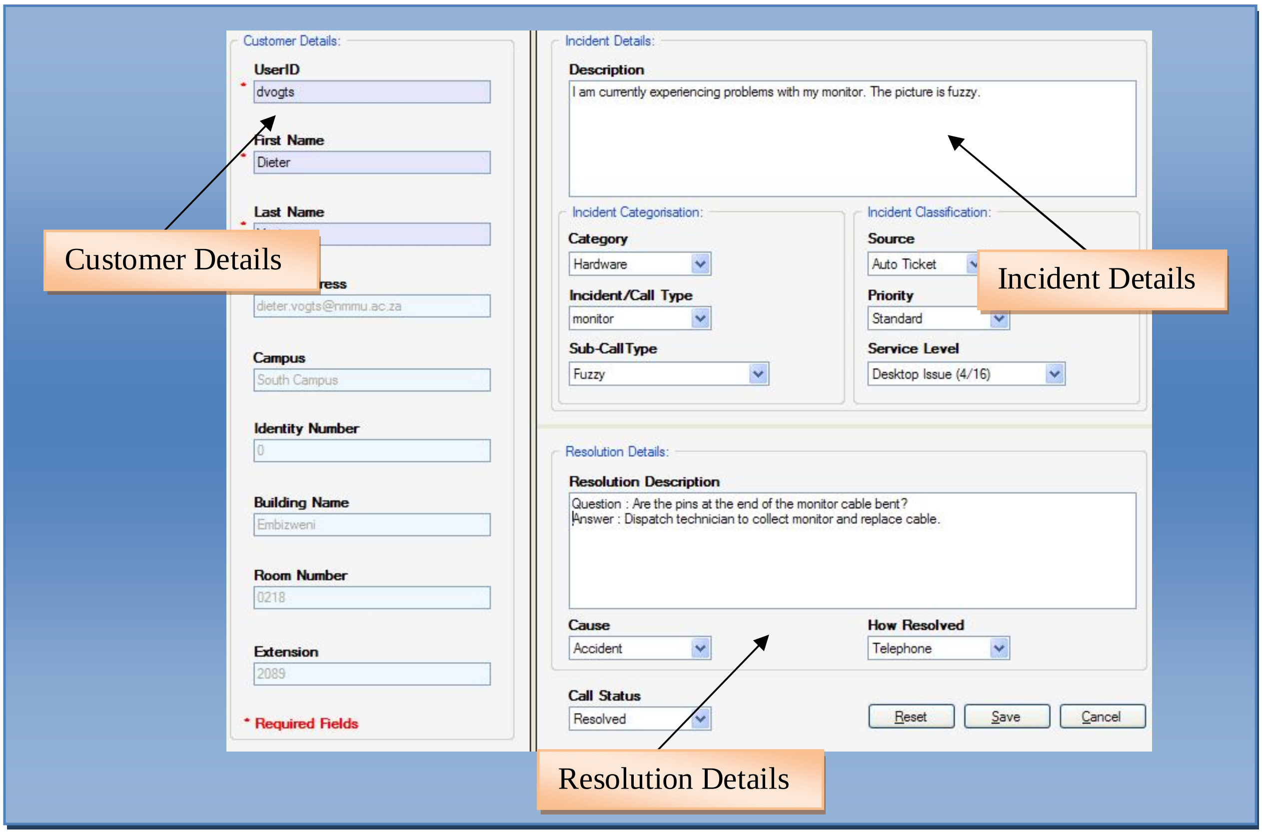


Figure 6.11: User Input Section

Customer Details allow for the identification of the customer who is currently experiencing a problem. The customer's query is described, categorised and classified in the Incident Details' Section. If a solution has been provided to the customer, this solution is described in the Resolution Details' section.

Input from the User Input Section is sent to the Agent Manager via the Plan Manager. Validation is performed by the Agent Manager against the input provided. The validation is based on the type of data required by the particular field. This information resides within the Task Model (Section 6.2.3.1).

6.3.1.3 System Feedback Section

The System Feedback Section (Section C Figure 6.8) is responsible for presenting candidate results found by the Agent Manager. Candidate results delivered by the Agent Manager are acquired from the Knowledge Base. These results are dependent on the current step and can either take the form of customer details, incident categorisation keywords or possible solutions (Figure 6.12).

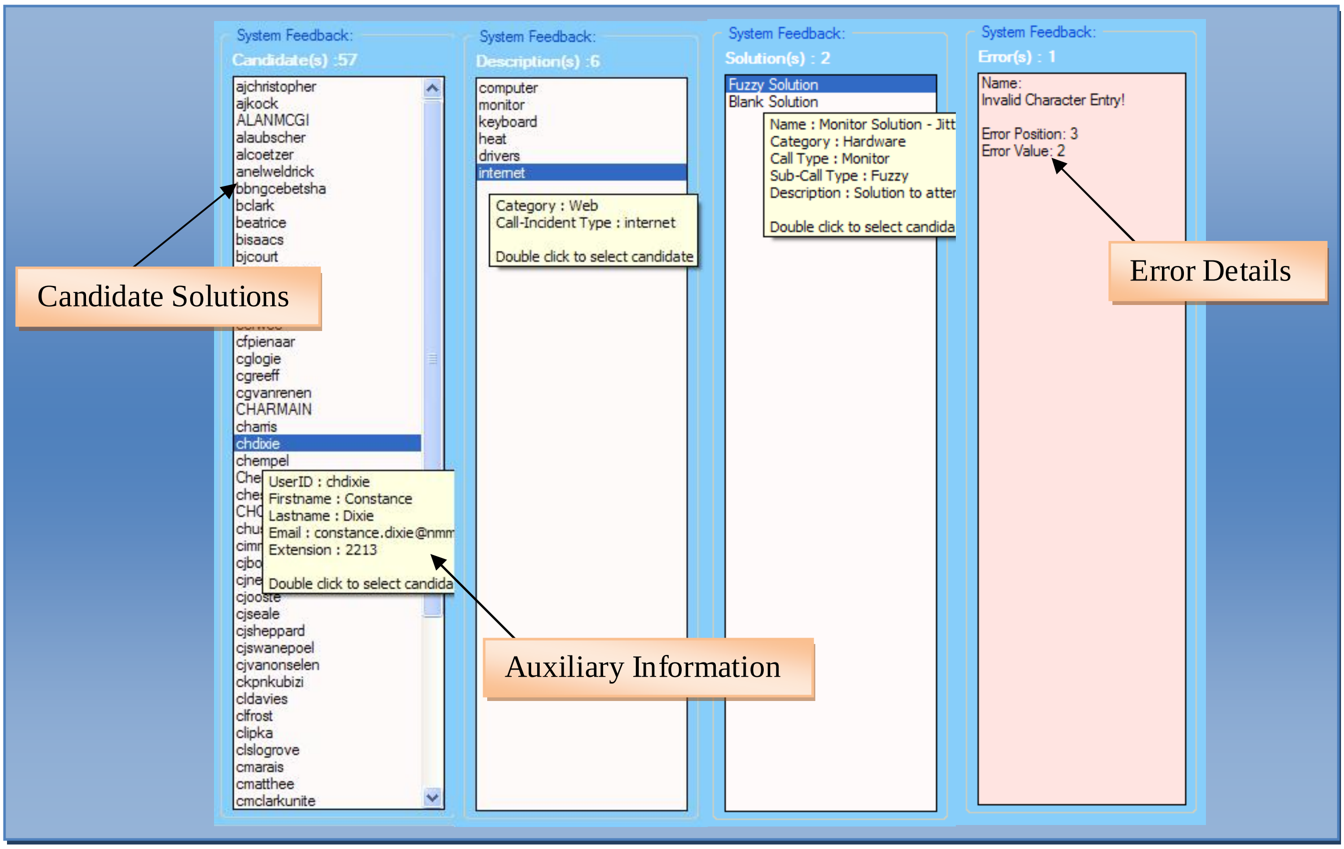


Figure 6.12: System Feedback Section

If the user has entered an incorrect data type in a particular field, an error will be displayed in the System Feedback Section (Figure 6.12). These errors are identified as a result of the validation process performed by the Agent Manager (Section 6.2.2). Auxiliary information is displayed in the form of a tooltip. This is viewable upon clicking on a candidate result. The purpose of the auxiliary information is to provide the user with

details on demand, in relation to a particular step, without the need to switch between windows in order to search for and verify customer, incident and resolution details.
The list of candidates or error details displayed within the System Feedback Section are sent from the Agent Manger to the Plan Manager and then to the UI (Figure 6.1).

Searching the Knowledge Base for possible candidates was achieved by using a Trie. A Trie can be described as a tree-based data structure that is used to support the pattern matching of strings (Goodrich and Tamassia 2002). A Trie T represents a set of strings S which extends from the root to the external nodes of T. It operates by being given a string X and traversing the tree in order to look for all the possible strings in S that contain X. Figure 6.13 illustrates the structure of a standard Trie for the strings: bear; bell; bid; bull; bye; sell; stock; and stop.

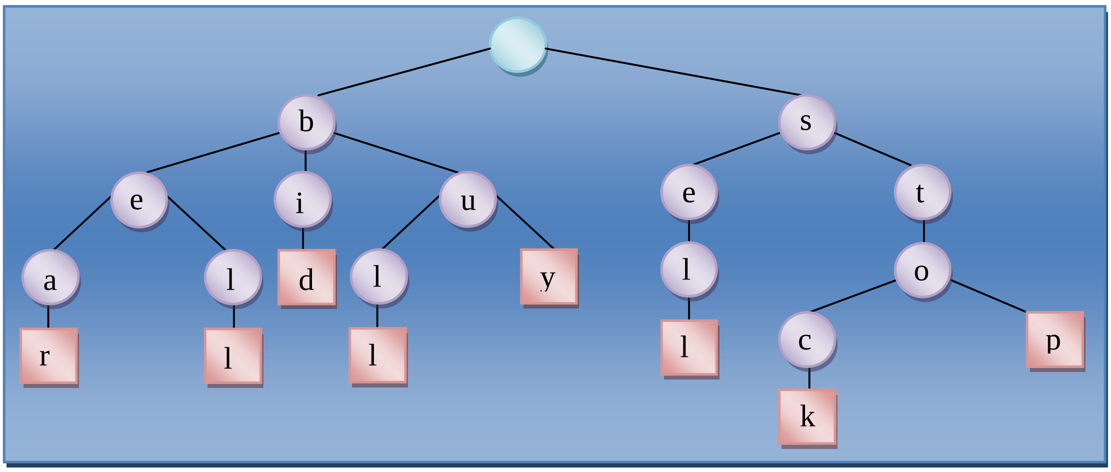


Figure 6.13: Standard Trie (Goodrich and Tamassia 2002)

A Trie was chosen to perform the pattern matching as its searching time for a particular string is $O(dm)$ where d is the size of the alphabet and m is the size of the string. The maximum amount of time spent on a single node is $O(d)$ (Goodrich and Tamassia 2002). This facilitates efficient searching of the Knowledge Base and could contribute to reducing CRR by improving the time in which information is retrieved and delivered to the user.

The Incident Details section (Figure 6.11) allows for a description of the customer's query to be entered. The Plan Manager sends this information to the Agent Manager which then calls a textual analyser. The role of the textual analyser is to remove all punctuations, conjunctions, prepositions, pronouns and verbs. These common words are stored within the Knowledge Base. A list of all possible incident/call types are loaded into a Trie from the Knowledge Base. The words of the new refined description are matched to those keywords within the Trie. All matches are indicated in the System Feedback Section (Figure 6.12).

The goal of the System Feedback Section (Figure 6.12) is to assist in providing plan recognition capabilities. The ability to dynamically deliver candidate results supports this. The candidate results can be utilised to reduce the time in which it takes to identify a customer, categories and classify his/her query, as well as to provide a possible resolution to the query based on its diagnosis.

A list of possible resolutions is identified (based on the categorisation of the customer's query) and is displayed in the System Feedback Section. The user can use the Auxiliary Information provided in the form of a tooltip in order to determine the most appropriate option (Figure 6.12).

6.3.2 Query Resolution Window

A second window (Figure 6.20) was implemented to support the resolution of customer queries. This window is a modal window and when active does not permit any interaction with the Query Diagnosis window (Figure 6.8) (Galitz 2002). A modal window was used to ensure that the user focuses, at that point, on resolving the customer's query.

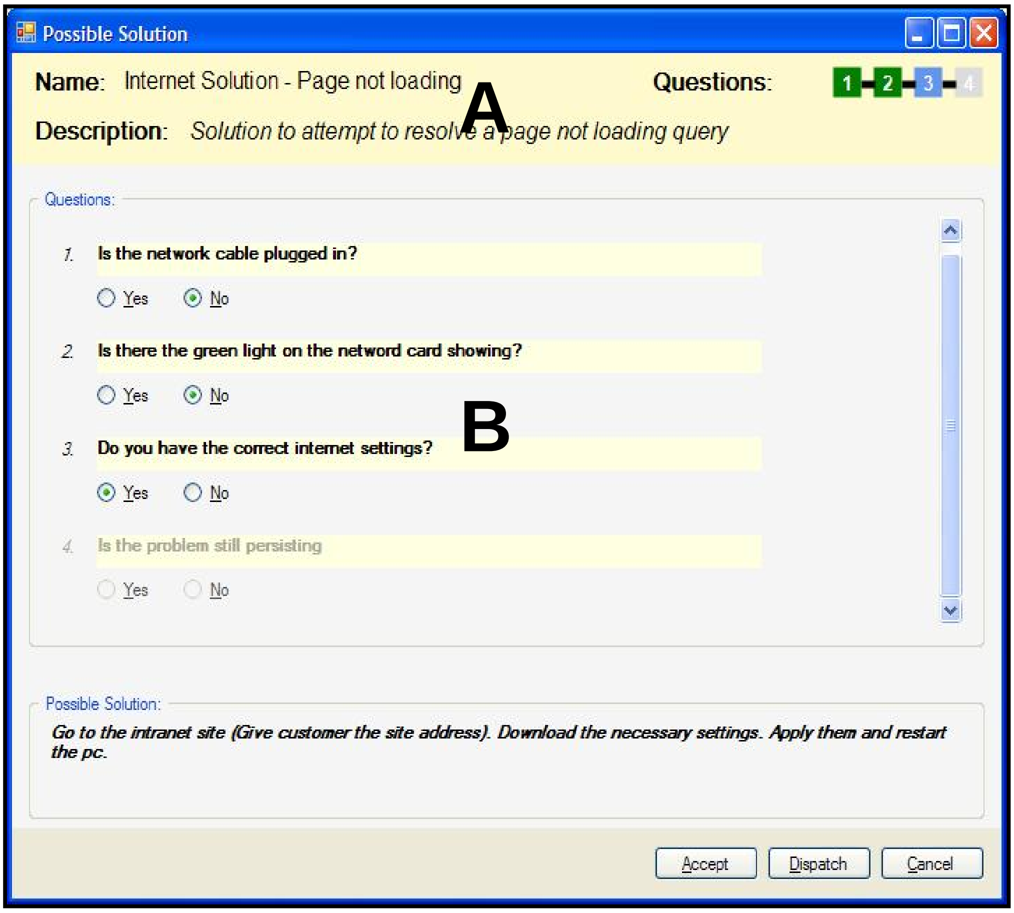


Figure 6.14: Query Resolution Window

The layout of this window was also designed according to the low-fidelity IUI design template (Figure 5.7). It consists of a Task Status Information Section (Section A) and a User Input Section (Section B).

Section A provides the name of the current solution selected by the user, its description and a Visual Question Indicator. The Visual Question Indicator displays the total number of questions and the completion status of each of those questions. The User Input Section (Section B) allows the user to ask the customer questions in order to determine the cause of the problem. Depending on the response of the customer, either the next question is enabled or a possible solution is displayed. This possible solution could be given to the customer in an attempt to resolve his/her query. The questions and answers are retrieved from the Solution Model (Section 6.2.3.3) which is stored in the Knowledge Base and are dynamically displayed depending on the possible solution selected from the System Feedback Section (Figure 6.13).

6.3.3 Customer Query Scenario

Section 6.3.1 described the implementation of a UI for the purposes of logging and diagnosing customer queries. The ability to provide query resolution at the first-level of support was achieved through the implementation of a UI that delivered this capability (Section 6.3.2). The purpose of this section is to describe a scenario of an actual customer query and to illustrate how the two windows function in order to log, diagnose and resolve a customer's query.

Scenario:

> Dr. Dieter Vogts is an employee at the Department of Computer Science and Information Systems. His email address is dieter.vogts@nmmu.ac.za and his office extension is 2089. Currently he is experiencing some the problems with the Internet. The pages continuously load as *Page Not Found*. He calls into the NMMU ICT Service Desk in order to receive assistance that could possibly solve the problem he is currently experiencing.

In order to successfully log, diagnose and resolve the problem experienced by Dr. Vogts there are three tasks that need to be completed by the CCA (Section 2.6):

a) Identify customer details;
b) Enter incident details; and
c) Enter resolution details and save the query.

6.3.3.1 Customer Identification

The first step that needs to be completed is customer identification. This is necessary in order to identify the customer who is reporting the query. Identification of the customer is done in the *Customer Details* section (Figure 6.15). Note that the requirements for the current step (which is to capture the customer's details), is displayed in the Task Status Section. The Visual Step Indicator highlights that the user is currently attempting the first step and the remaining two steps have not been attempted.

The UserID, first name, or last name of the user can be provided. As the user enters the customer's details, various candidate options are displayed in the System Feedback Section (Figure 6.15). A single click on any one of these candidates will reveal more information about the customer in the form of a tooltip. Double clicking on any one of the candidate options will automatically fill in the particulars of the customer in the Customer Details Section (Figure 6.16). This completes the customer identification process. The next step is to diagnose the customer's query.

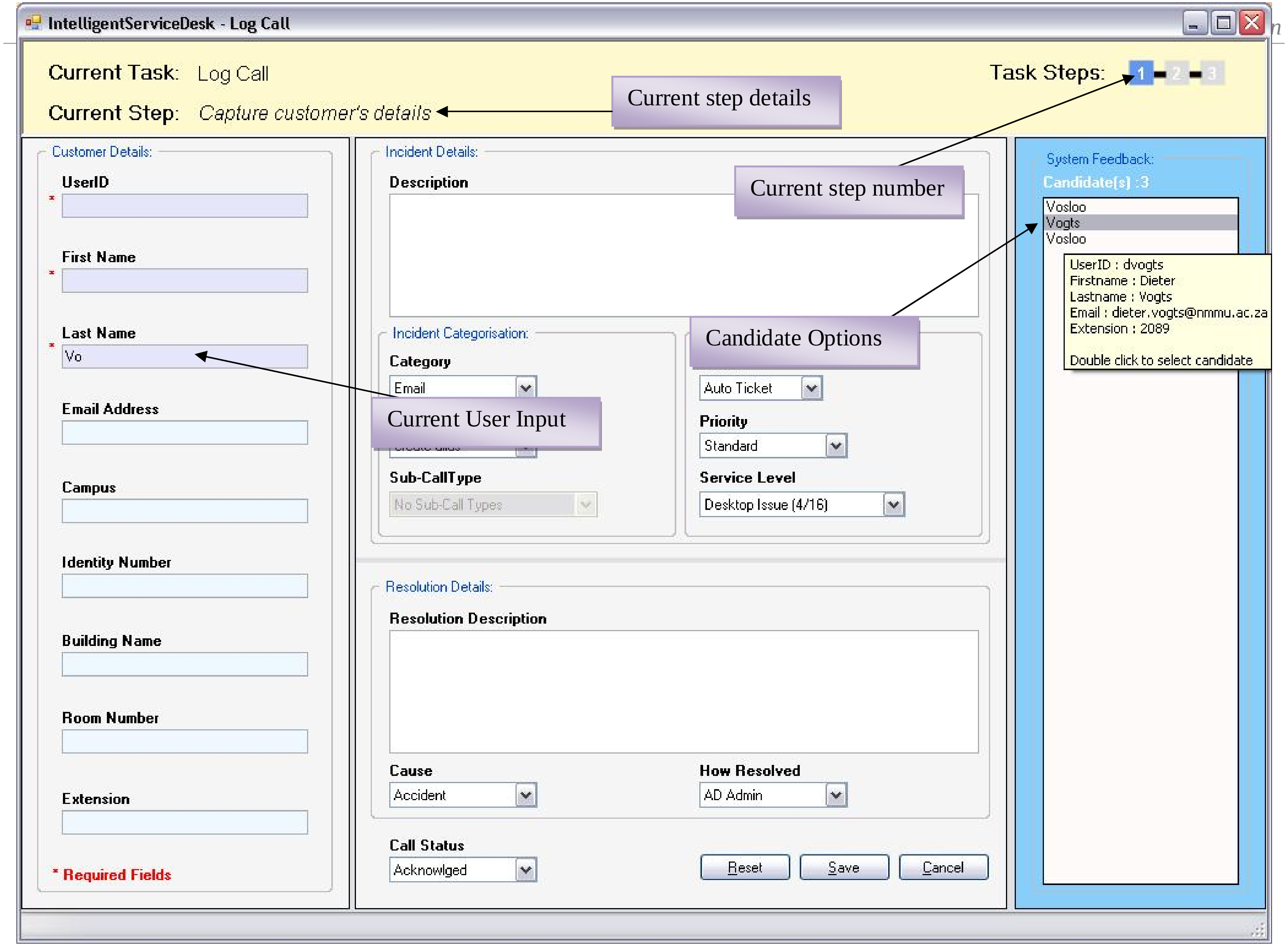


Figure 6.15: Identification of Customer Details (Step 1)

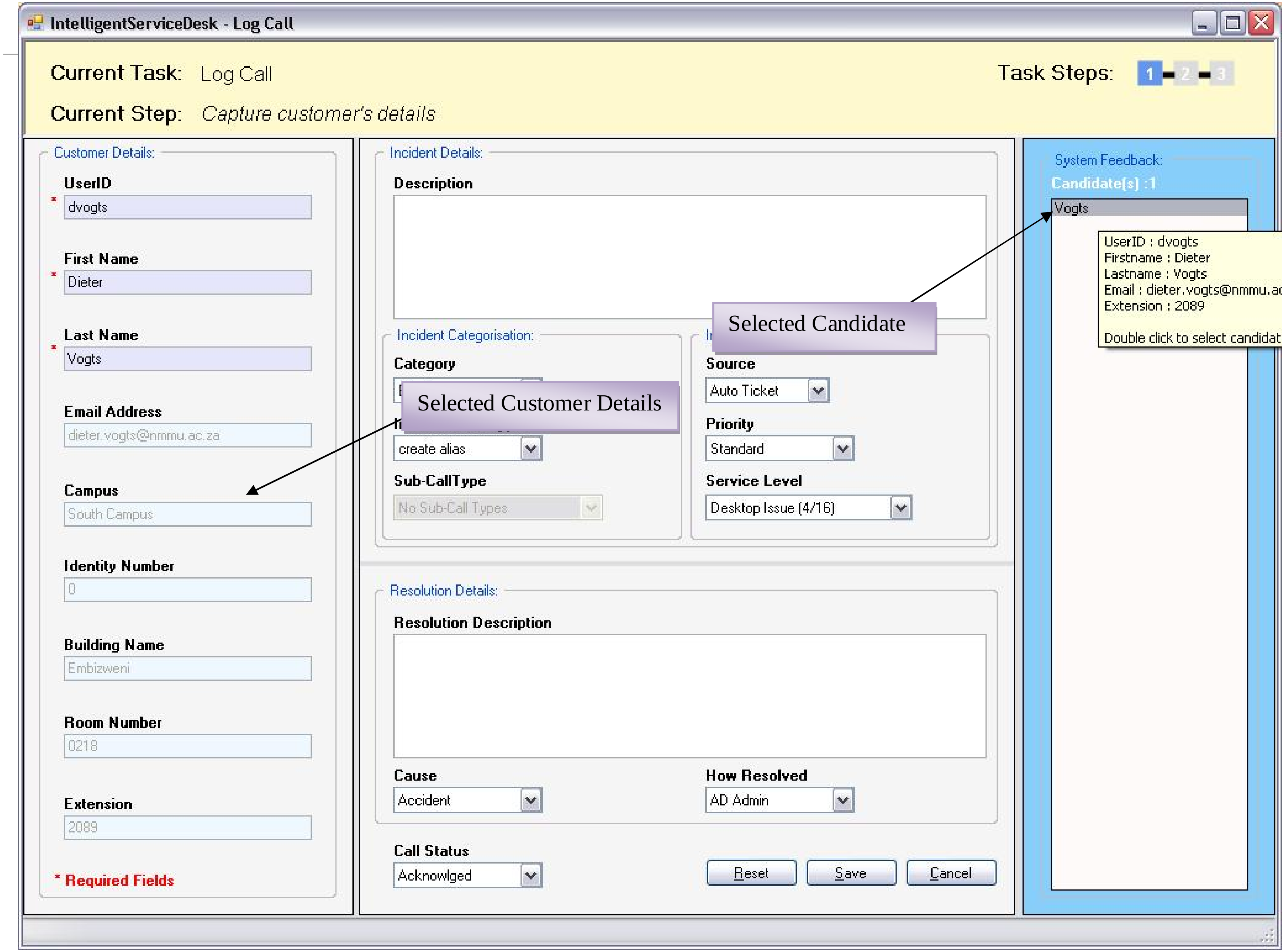


Figure 6.16: Selection of Customer Details (Step 1)

6.3.3.2 Query Diagnosis

Diagnosing the customer's query involves completing the Incident Details Section (Figure 6.17). The Incident Details Section consists of a description field for the customer's query, an Incident Categorisation Section which can be used to categorise the type of problem the customer is experiencing and an Incident Classification Section in which the source and severity of a customer's query can be classified. Figure 6.17 reflects the changes in the window when progressing to the query diagnosis step. The current step information is changed to indicate that the user must capture and diagnose the customer's query. Indication that the first step has been completed is illustrated by the Visual Step Indicator. The user can determine that he/she has completed the first step and is currently busy with the second step through the use of colour.

The description of the customer's query can be entered in the same manner in which it has been dictated by the customer (Figure 6.17). The pattern-matching algorithm will automatically parse the text and attempt to dynamically categorise the customer's query. This dynamic categorisation is displayed in the System Feedback Section (Figure 6.17). Double clicking (selecting) a candidate option will automatically categorise the query and this categorisation will be reflected in the Incident Categorisation Section (Figure 6.18). Capturing the Sub-Call Type and the Incident Classification fields need to be done manually.

The last step is to provide a possible resolution to the customer's query based on its categorisation. This feature was added in an attempt to improve CRR by delivering query resolution capabilities at the first level of support.

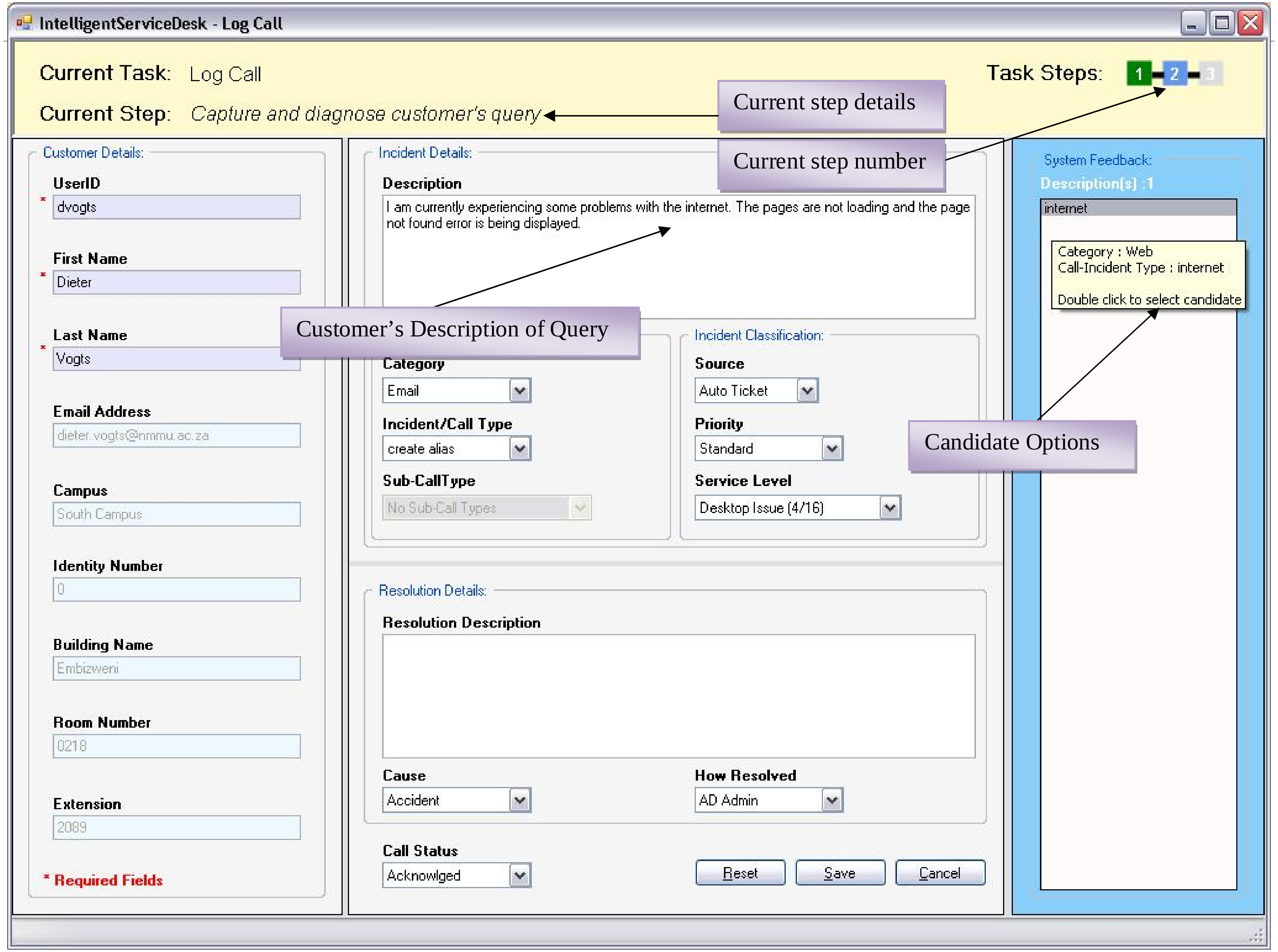


Figure 6.17: Diagnosis of Customer Query (Step 2)

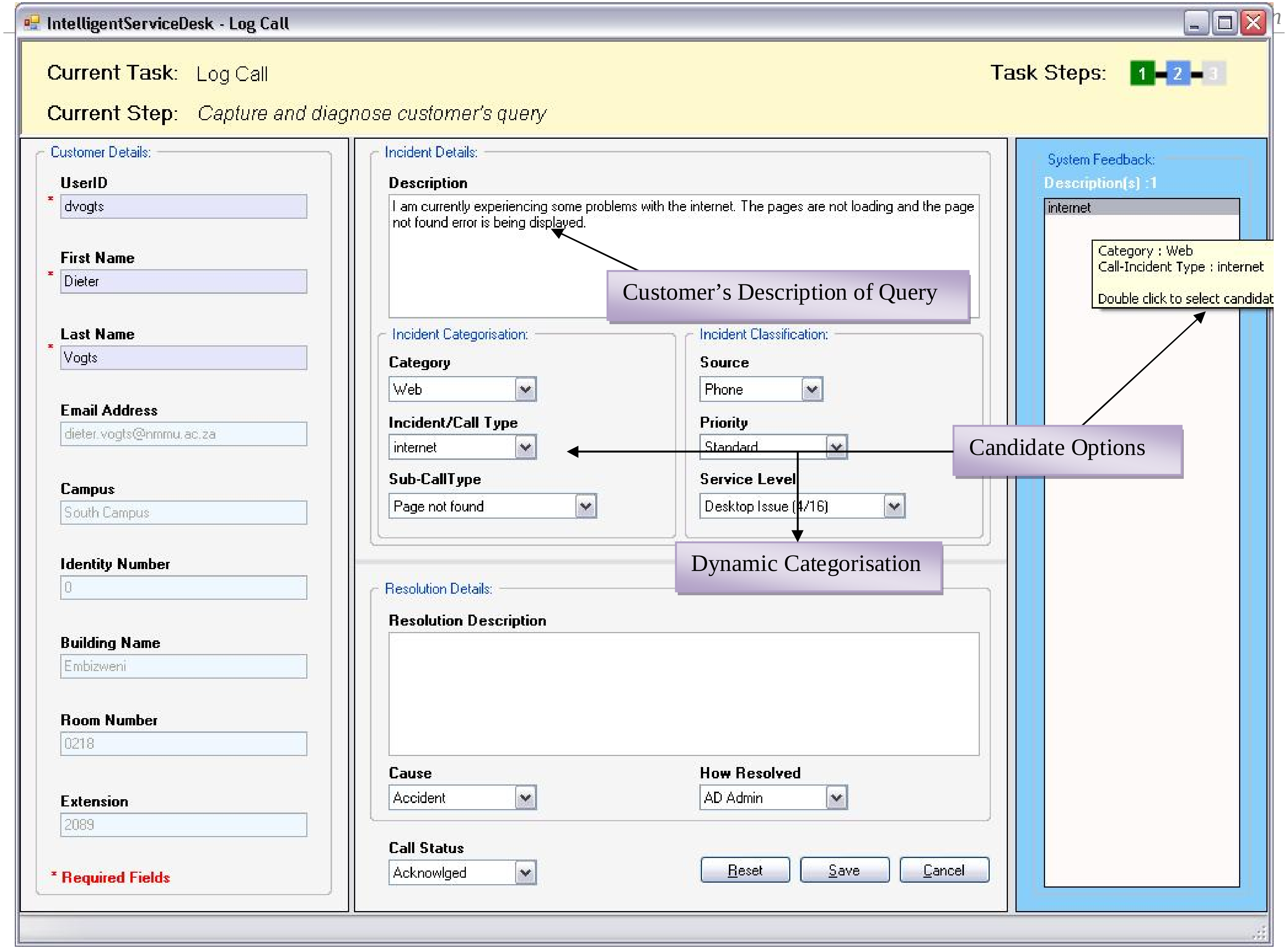


Figure 6.18: Dynamic Query Categorisation (Step 2)

6.3.3.3 Query Resolution

The previous steps have described how to identify a customer and diagnose the problem that he/she is currently experiencing. Clicking on the resolution description box in the Resolution Details Section will change the description of the current step, update the Visual Step Indicator and offer candidate solutions in the System Feedback Section (Figure 6.19).

Double clicking on a candidate solution in the System Feedback Section will display the Query Resolution window (Figure 6.20). A description of the current solution is provided along with a list of possible questions the user can ask the customer. Based on the customer's response to the questions, a possible resolution could be displayed or the next question enabled. Progress through the questions is reflected in the Visual Question Indicator.

If the appropriate solution is found, the user can click the Accept button and return to the Query Diagnosis window. The description of the possible resolution from the Query Resolution window is displayed in the Resolution Description box in the Resolution Details Section (Figure 6.21). The Cause, How Resolved and Call Status fields need to be completed manually. This completes the process of diagnosing and resolving a customer's query.

This section has highlighted how through the use of the Query Diagnosis and Query Resolution windows, a customer's query can be logged and resolved. Logging the customer's query was discussed in terms of identifying the customer, diagnosing the query as described by the customer and offering a possible resolution to the customer based on the categorisation of the query.

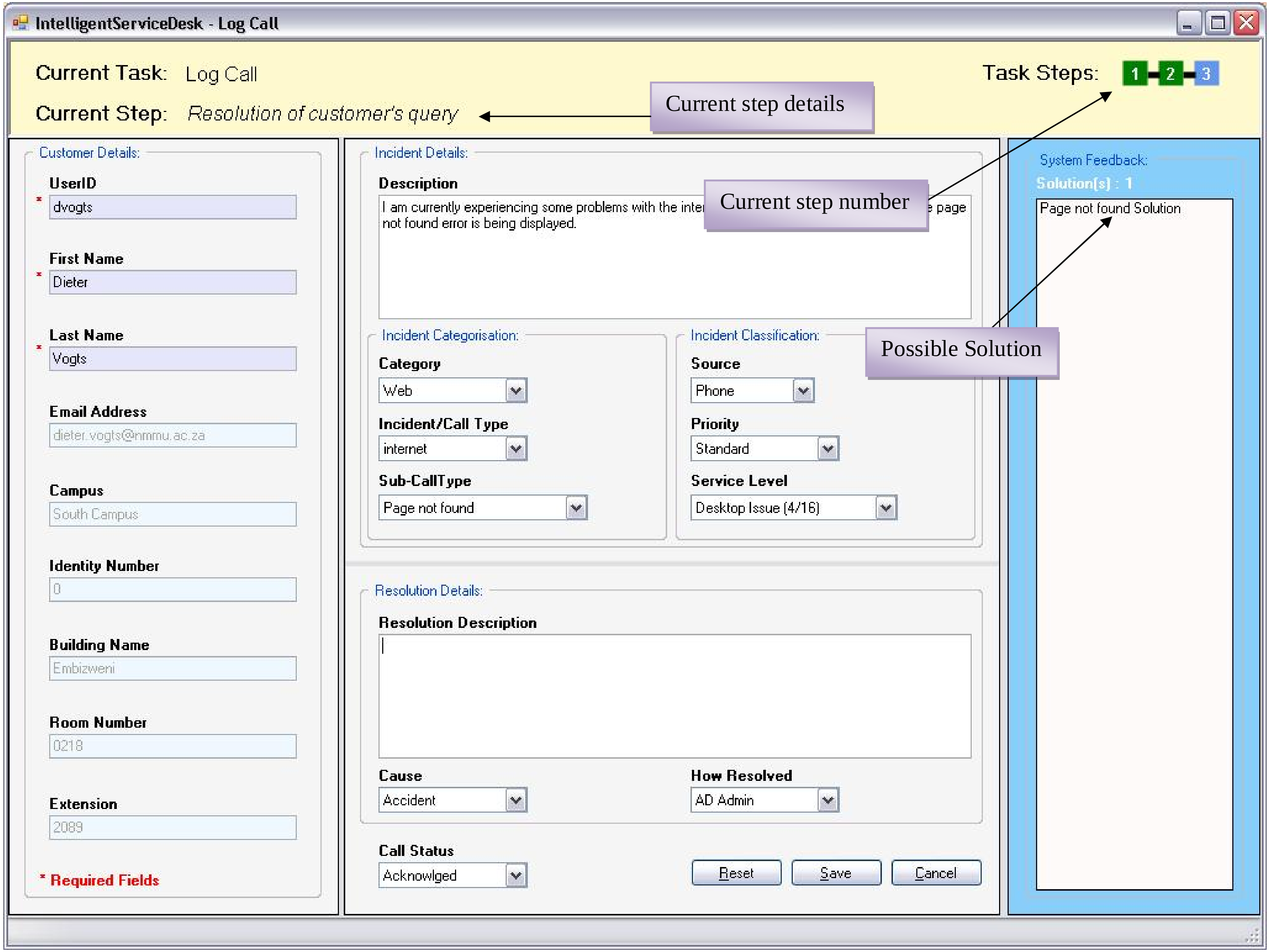


Figure 6.19: Resolution of Customer's Query (Step 3)

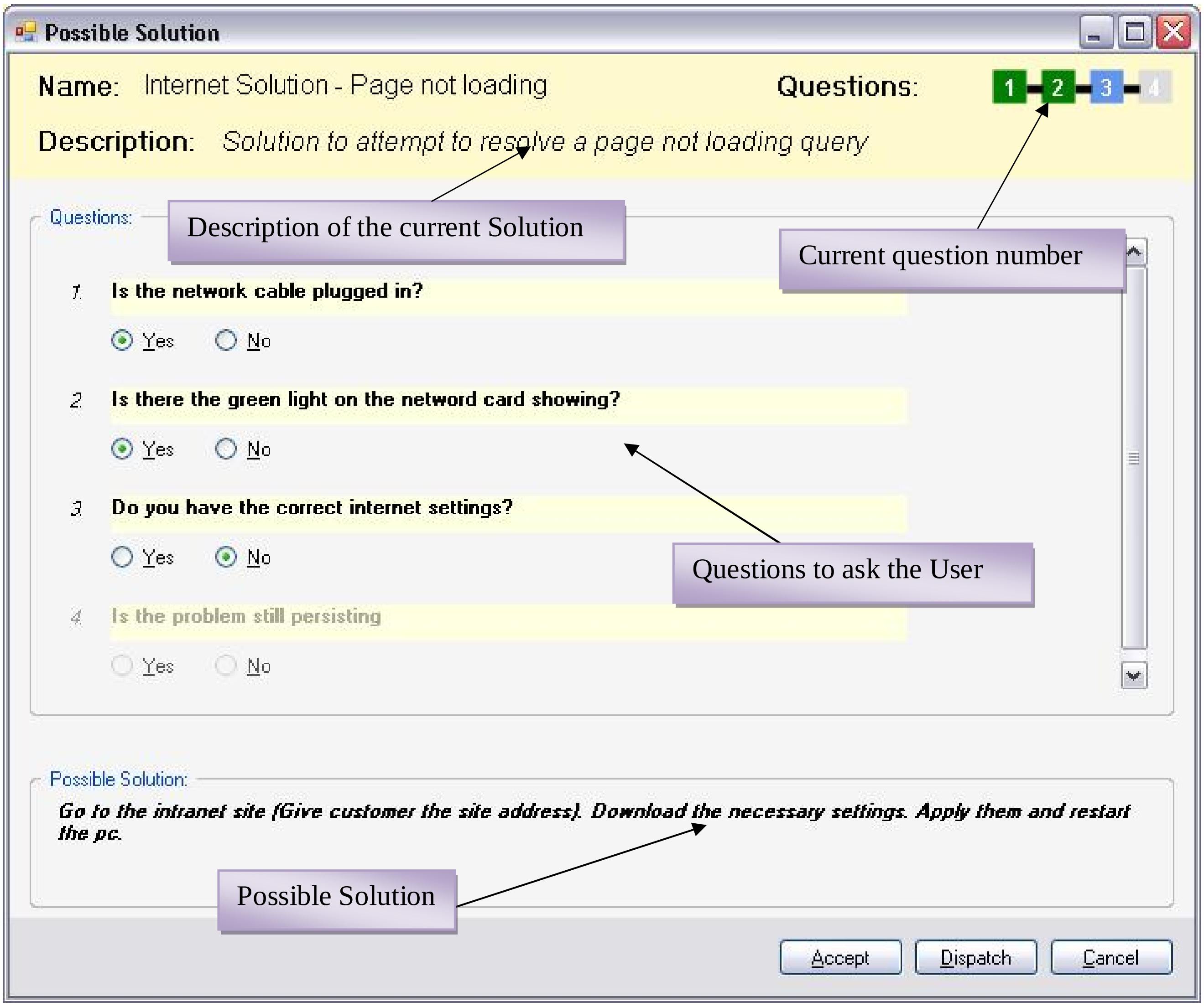


Figure 6.20: Query Resolution Window (Step 3)

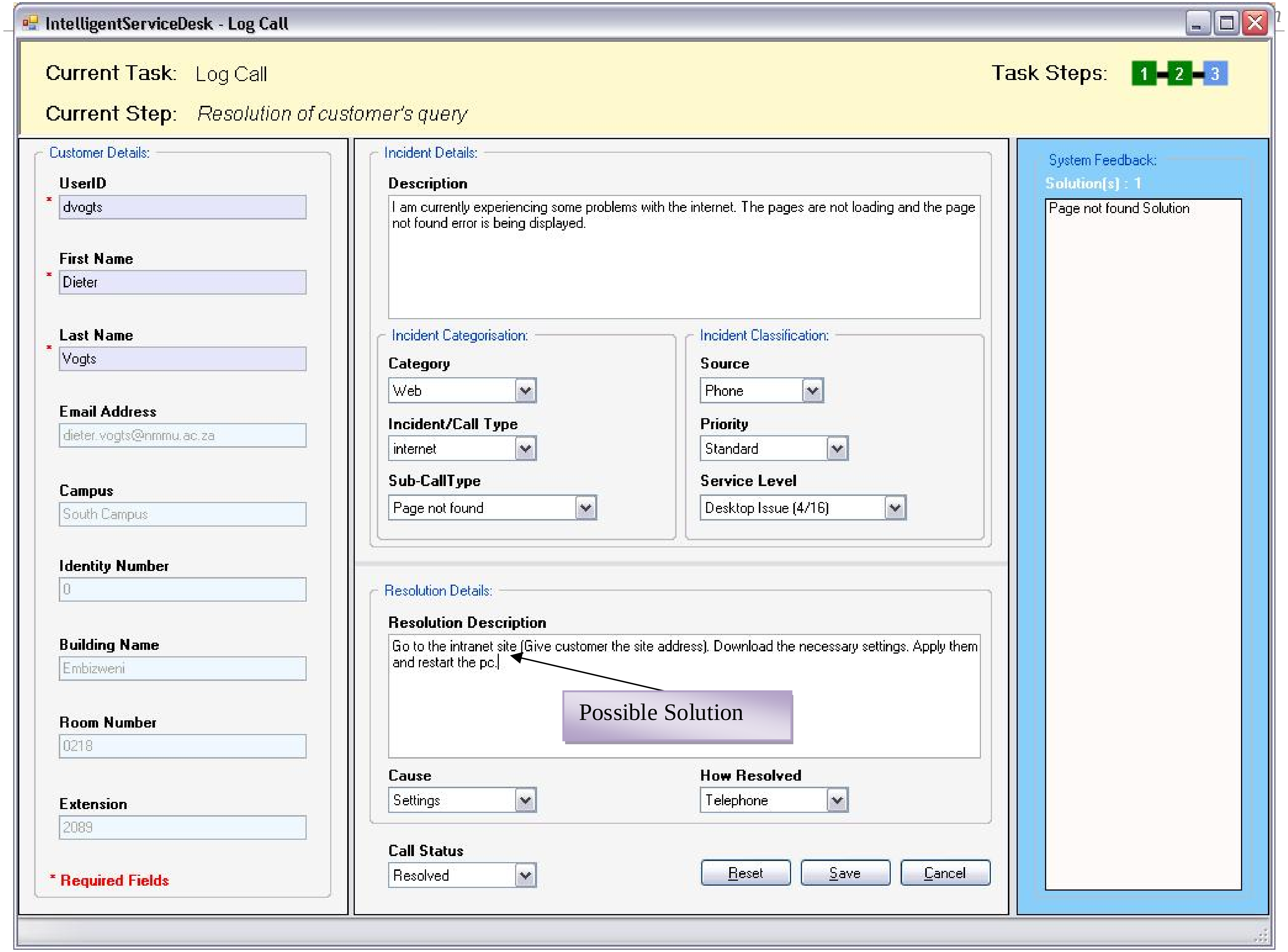


Figure 6.21: Resolution of Customer's Query (Step 3)

6.4 Conclusion

This chapter showed that the IUI model defined in Chapter 5 can be successfully implemented to produce an IUI prototype for CCs. Through the implementation, it was identified that the IUI prototype was capable of plan recognition capabilities as well as the ability to provide intelligent feedback in order to potentially reduce CRR.

Implementation of the IUI architecture (Section 6.2) resulted in the creation of the Plan Manager (Section 6.2.1), the Agent Manager (Section 6.2.2) and the Knowledge Base (Section 6.2.3) components. It was identified that the remaining components of the proposed IUI model need not be implemented as they do not directly support the call resolution process.

The Plan Manager was implemented to provide plan recognition capabilities. This was achieved by allowing the Plan Manager to access the Knowledge Base, retrieve the Task Model and deliver the high- and low-level goals of the current task to the user (Section 6.2.1).

Processing of user input and the ability to provide intelligent feedback was achieved through the implementation of the Agent Manager (Section 6.2.2). The implementation of this component incorporated the use of regular expressions to support and facilitate the validation of user input. A Trie data structure was used to support pattern-matching user input against data stored in the Knowledge Base for the purposes of customer identification, categorisation and classification of customer queries.

All of the Knowledge Base components discussed in Chapter 4 were implemented. The implementation of these components was done in XML as it provided a high-level of customisability, flexibility and adaptability.

Two windows were implemented to support the call resolution process. A Query Diagnosis window (Figure 6.8) was implemented to provide the capabilities of identifying a customer, dynamic categorisation and classification of customer queries and the ability to capture query resolution details. The implementation of the second window, the Query Resolution window (Figure 6.14), supported the need to provide query resolution capabilities at the first-level of support.

The next chapter (Chapter 7) will focus on the evaluation of proposed IUI model and the IUI prototype. The objective of Chapter 7 is to determine to what the proposed IUI model and the IUI prototype meet their objectives.

Chapter 7: Evaluation

7.1 Introduction

In Chapter 5, an IUI model was proposed for the purpose of potentially improving CRR within CCs. Chapter 6 presented a discussion that demonstrated that the proposed IUI model could be implemented. The implementation of the proposed IUI model, as a proof of concept, only demonstrates its effectiveness and does not determine the extent to which the IUI prototype meets the criteria of an IUI. Further evaluation needs to be done, in the form of user testing, to determine the usability and usefulness of the IUI prototype.

The aim of this chapter is to evaluate to what extent the proposed IUI model and IUI prototype meet their objectives. This evaluation involves determining to what extent the proposed IUI model aids the development of IUIs for CCs and the extent to which the IUI prototype supports plan recognition, task support and query resolution.

7.2 Existing Methods and Instruments

An investigation into existing methods and instruments for evaluating IUIs was performed. This investigation revealed that many researchers focused on the results of the evaluation and presented minimal information on the methods and instruments used for the evaluation. This resulted in no formal methods and instruments being identified.

Due to the lack of methods and instruments, in evaluating IUIs, the sections that follow attempt to evaluate the IUI prototype by means of conducting a model evaluation (Section 7.3) and user testing (Section 7.4). Eye-tracking was used to aid the user testing. Instruments used during the evaluation process were a test plan and a post-test questionnaire.

7.3 Model Evaluation

If an interface is considered intelligent, it would require that the interface is able to present answers to the questions in Table 3.1 as well as exhibit the characteristics of a typical IUI (Section 3.3).

The purpose of this model evaluation is to determine whether the implemented IUI prototype, called Intelligent Service Desk (ISD) (Figure 7.1), meets the criteria of an IUI. In order to do this, the first step is to see how many of the six questions in Table 3.1 the interface is capable of answering. These questions will be answered with respect to the primary task of logging a customer's query.

1. Who should/can/will do?

The tasks that need to be performed by the user are clearly indicated in the Task Status Section (Section 6.3.1.1). Information is provided as to which steps need to be completed, the order in which they could be completed, and a brief description of the purpose of each step. A clear distinction is made by means of colour in terms of which areas of the screen are for providing user input (Section 6.3.1.2), and which areas deliver system feedback (Section 6.3.1.3).

2. What should I/we do next?

Determining what should be done next is handled by the Visual Step Indicator (Section 6.3.1.1). Based on its numbering, it provides an indication of the sequence of steps.

3. Where am/was I?

Identification of the current step is done in the Task Status Section (Section 6.3.1.1). A description of the current step as well as the colour of the Visual Step Indicator provides the user with this information.

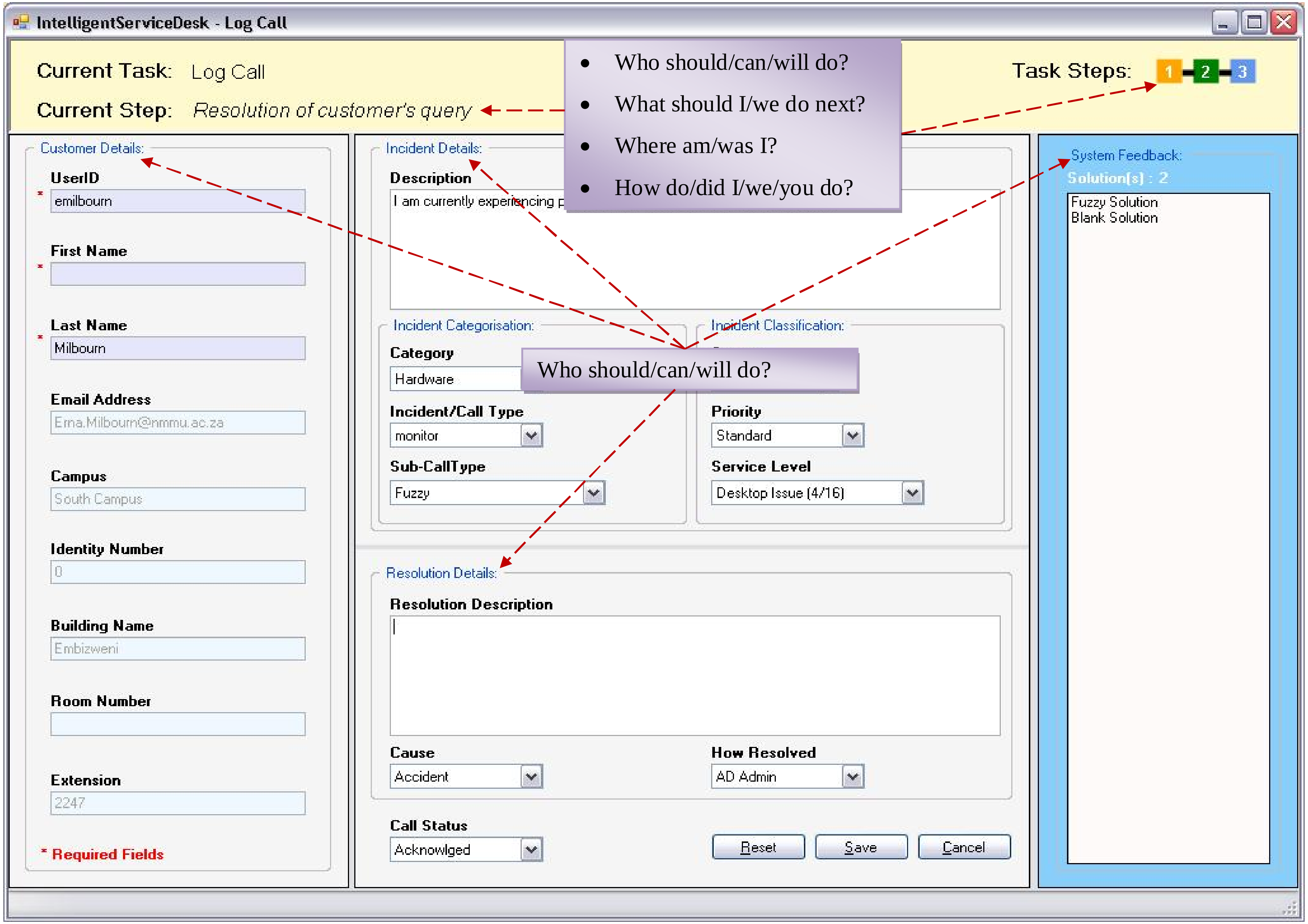


Figure 7.1: ISD – Query Diagnosis Window

4. When did I/you/we do?

No feeedback is provided in terms of when a step is attempted and completed other than the status of the step. This status is reflected in the Visual Step Indicator.

5. Why did you/we (not) do?

The main task is to log a customer's query and to provide a possible resolution. Any subsequent steps assist in achieving this task. Information provided within the Task Status Section assists the user in completing the required task.

6. How do/did I/we/you do?

The goal of the task is to log and attempt to resolve the customer's query. This is done by means of completing various steps. Indication of the total number of steps and the completion status of each of these steps are represented by the Visual Step Indicator. As the user attempts a step, a description of the current step, and what is required, is displayed in the Task Status Section. During this process of completing the task the Visual Step Indicator is updated accordingly.

Five out of the six questions (Who should/can will do, What should I/we do next, Where am/was I, Why did you/we (not) do and How do/did/I/we/you do) could be answered by the interface. This shows that the interface does contain intelligence. A confirmation of this would be to determine whether the IUI prototype reflects the characteristics of an IUI (Section 3.3).

Although the proposed model does support multimodal interaction, through the Input/Output Manager, this component was not implemented as it did not directly support the logging and resolving of customer queries (Section 6.2). Therefore, the characteristics of providing multimodal, natural and gestural interaction will not be evaluated against.

1. Detect and Correct User Errors

The ability to detect and correct user enforced errors is provided (Figure 7.2). Notification in the System Feedback Section alerts the user that an error has occurred. This notification occurs by means of:

- A change in background colour of the System Feedback Section;
- An indication of the field in which the error has occurred by means of a visual aid;
- A description of the error; and
- The position and values of the incorrect characters.

Therefore, the user is not only provided with information regarding what caused the error, but also information on how it could be corrected.

2. Provide Guidance through the Application

Guidance is provided through the application by means of the Task Status Section and the System Feedback Section. The Task Status Section (Figure 7.3) provides guidance by means of:

- Providing an indication of the current task and step;
- A description of the current step that needs to be completed; and
- The total number of steps that need to be completed.

The System Feedback Section (Figure 7.3) also guides the user by providing a list of candidate results. These results vary depending on the current step. Candidate results could be in the form of:

- Possible customers;
- Possible categories; and
- Possible solutions.

These candidate results guide the user by means of identifying the most appropriate option required to complete the step. The example in Figure 7.3 shows a list of possible customers with the UserID containing the character ‘c’. This feature further assists with rapid task completion.

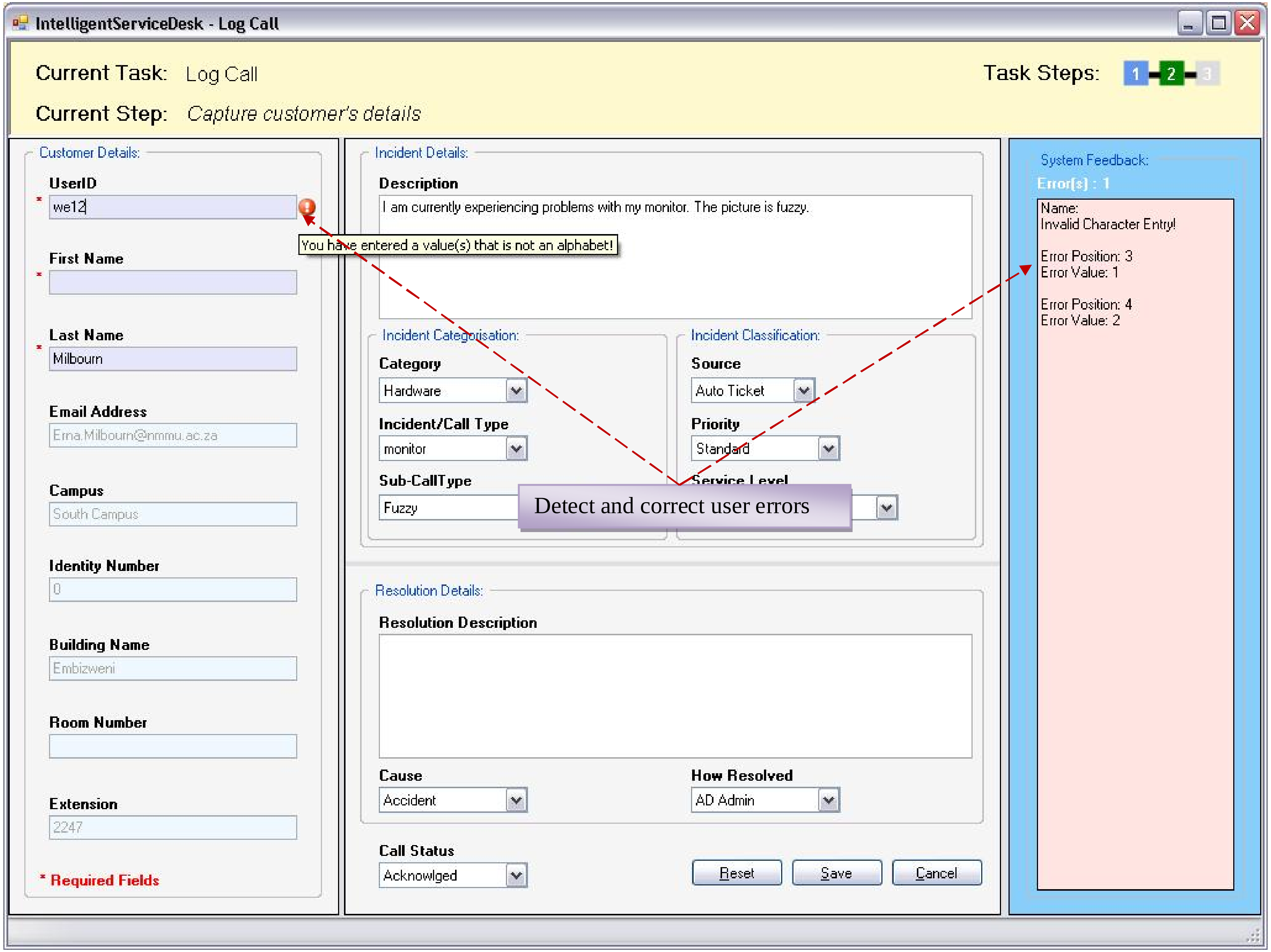


Figure 7.2: ISD - Error Detection

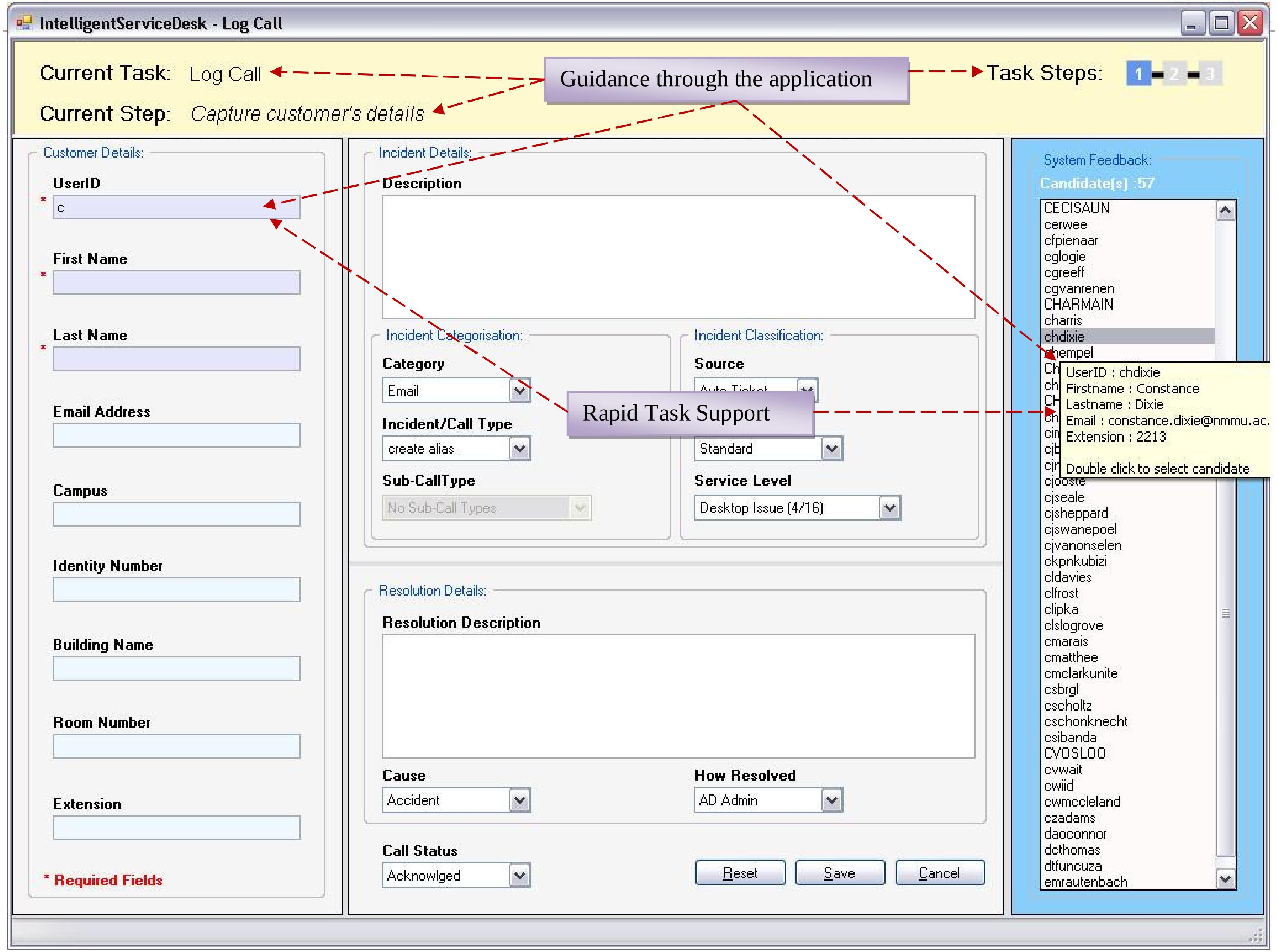


Figure 7.3: ISD - Providing Guidance

4. Support Rapid Task Completion

The ability to provide candidate data as the user provides input supports rapid task completion. This reduces the need to enter tedious amount of data in order to complete a task. Figure 7.3 shows how a list of candidate customers' is generated based on the input of a single character. As more characters are entered, the greater the accuracy of identifying the possible customer. Clicking on a candidate result will reveal more information about that particular candidate. Selecting a candidate result from the list will automatically fill in the required fields (Figure 7.4). Note that in Figure 7.4, all of the customer details have been filled in, as opposed to the single character entered in Figure 7.3. A key feature of the IUI prototype is its ability to offer a diagnosis of the customer's query, based on the description provided. Figure 7.4 illustrates this capability.

The user need not manually categorise the query. As the user enters a description of the query (as dictated by the user) candidate categories and call types are provided in the System Feedback Section. Clicking on a candidate result will reveal the category and call-types. Selecting the candidate will automatically set the Category and Call-Type fields. This reduces the need to manually scroll through lists in order to categorise the customer's query.

The above discussion shows that the interface of the prototype exhibits all of the required characteristics in order for it to be classified an IUI.

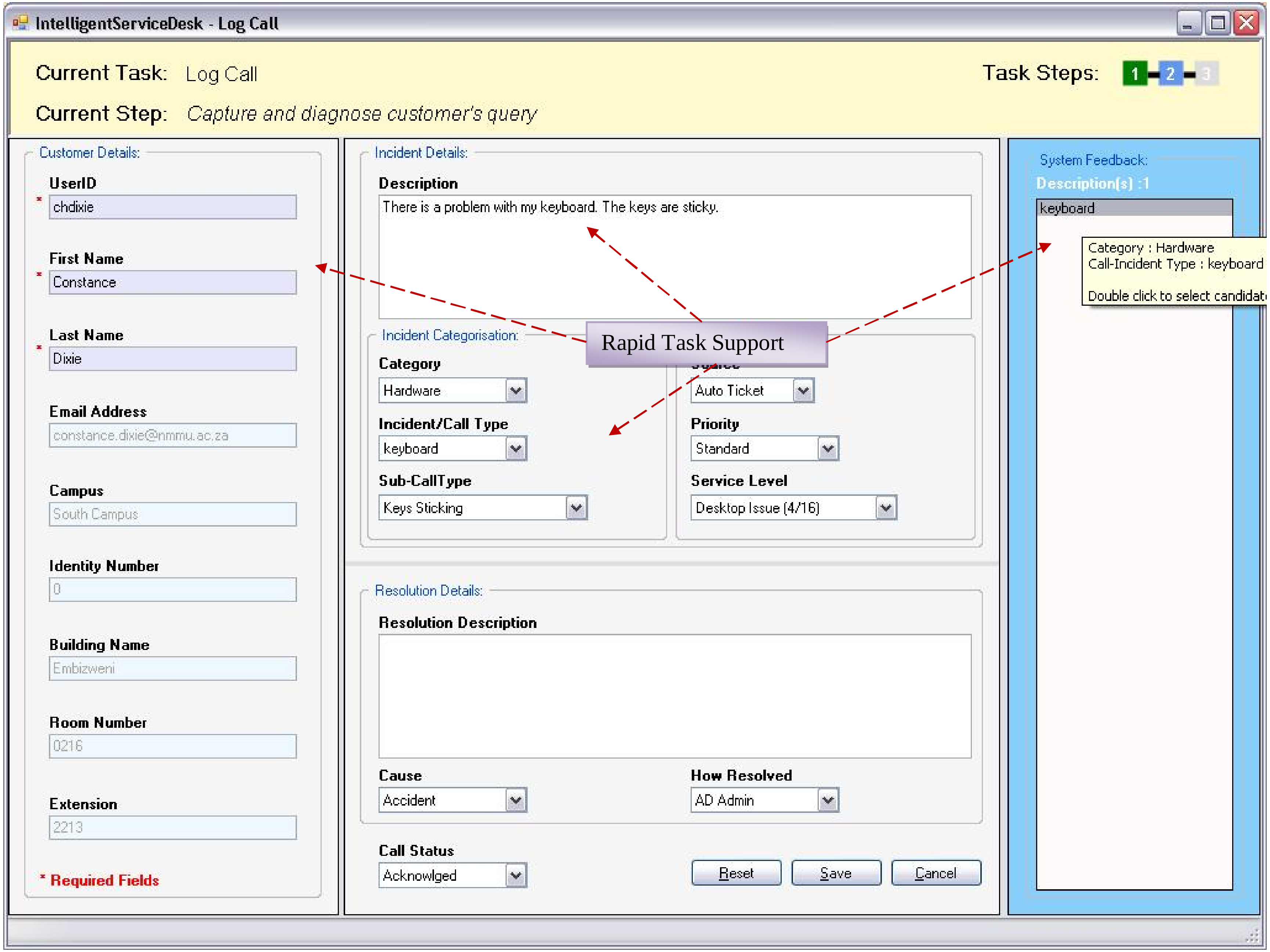


Figure 7.4: ISD - Rapid Task Completion

The aim of this section was to determine to what extent the IUI prototype possesses intelligence. Intelligence was evaluated by means of being able to answer the questions specified in Table 3.1 and to determine whether the UI of the prototype displays the characteristics of a typical IUI (Section 3.3). The IUI prototype was able to answer five out of the six questions, and it displayed the necessary characteristics of an IUI. Based on these results, it can be concluded that the UI of the prototype is intelligent and can be regarded as an IUI.

The next section aims to address the second question of the evaluation – to what extent does the IUI prototype support the users' tasks? This is discussed in terms of user testing.

7.4 User Testing

User testing was conducted to determine the usability of the IUI prototype. Two aspects of usability were measured, namely satisfaction and effectiveness. These aspects were measured in order to determine the extent to which the prototype supports plan recognition, task support and query resolution. The results of this part of the evaluation yielded both quantitative and quantitative data (Section 7.4.3).

Quantitative data can be either objective or subjective (Barnum 2002). Objective measures are a result of observations of the user performing various tasks. Subjective data refers to the quantified opinions of users, expressed as values, emanating from the use of a user satisfaction questionnaire. Eye-tracking was used to measure the effectiveness of the prototype. The data resulting from the eye-tracking was objective and contributed in determining whether the prototype was able to meet the users' goals of successfully logging and resolving customer queries. A post-test questionnaire (user satisfaction questionnaire) was also used as part of the evaluation to obtain subjective data. This subjective data attempted to measure the users' satisfaction with the prototype.

Qualitative data refers to descriptions expressed in the form of users' opinions and suggestions (Preece *et al.* 2007; Barnum 2002). These opinions and suggestions can be ascertained during the testing processes as well as from those opinions expressed in the user satisfaction questionnaire. Qualitative data was analysed by means of a thematic analysis of user opinions and suggestions emanating from the user satisfaction questionnaire.

7.4.1 Evaluation Methodology

The methodology used to conduct the user testing involved a combination of both usability evaluation and eye-tracking techniques. The data gathered from the eye-tracking techniques was used to supplement the data gathered from the usability evaluation.

Table 7.1 tabulates the steps involved in planning an effective usability evaluation, incorporating eye-tracking (Pretorius, Calitz and van Greunen 2005). Conducting the usability evaluation is discussed in Section 7.4.1.4.

Steps	**Description**
1.	Establish the Team
2.	Defining Product Issues and Testing Audience
3.	Setting Test Goals and Usability Measurements
4.	Defining Eye Tracking Metrics
5.	Determining the User Profile
6.	Selecting which Tasks to Test
7.	Determining How to Categorise and Analyse Results
8.	Writing the Test Plan
9.	Prepare Additional Test Materials (Post – Test Questionnaire)
10.	Recruit Test Participants
11.	Conduct Walkthrough and Pilot Test
12.	Conduct Usability Test
13.	Gather and Sort Data
14.	Tabulate and Analyse Data

15.	Recommend Changes to Prototype
16.	Report Results

Table 7.1: Proposed Methodology (Pretorius *et al.* 2005)

Eye-tracking was used as one of the methods in the evaluation. The intention behind using eye-tracking as a method was to determine whether the test participants were looking at certain areas of the prototype. Three areas of interest (AOIs) were identified, namely the Task Status Section, the System Feedback Section and the User Input Section (Figure 7.5). Observational data recorded by the eye-tracker revealed whether the test participant looked at a particular AOI.

7.4.1.1 Goals

The goal of the user testing was to determine whether the IUI prototype meets the users' goals and enables them to successfully log and resolve customer queries. This was necessary in order to assist in realising the goal of the evaluation - determining the extent to which the IUI prototype supports plan recognition, task support and query resolution.

7.4.1.2 Measurements and Metrics

Usability is defined as "*the extent to which a product used by specific users is able to achieve specific goals with effectiveness, efficiency and satisfaction within a specified context of use*" -ISO 9421-11 1998.

User satisfaction (Section 7.4.3.1) and effectiveness (Section 7.4.3.2) are discussed in this evaluation. Efficiency is not discussed because it requires a longitudinal study reflecting the extended use of an IUI within a CC environment.

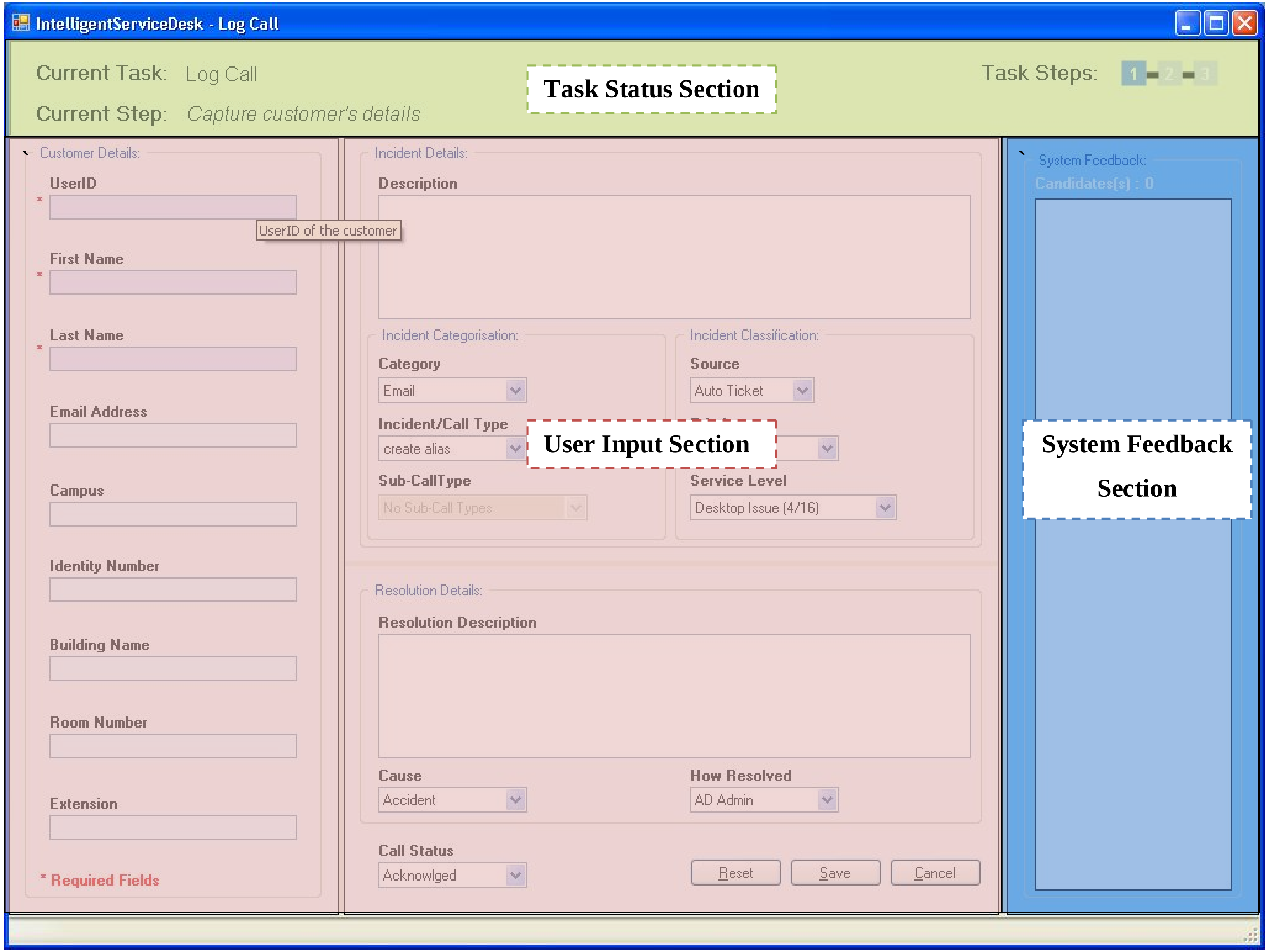


Figure 7.5: Eye-Tracking AOIs

Eye-tracking metrics assist in determining where users are focusing their attention on the screen (Pretorius *et al.* 2005). These metrics indicate how users perceive and use the information presented on the screen, and to evaluate the usability of screen layouts. Data resulting from eye-tracking compliments a usability study, as it is able to detect and provide additional knowledge that would be undetectable through the use of standard usability techniques. It is for this reason that eye-tracking was included as part of the user testing. Eye-tracking metrics provide:

- The number of fixations – time spent looking at an AOI;
- The duration of each fixation;
- The number of gazes at each AOI;
- The scanpath – movement of the eye from one area to another;
- The time to the first fixation on an AOI;
- The observation count of an AOI; and
- The observation duration of an AOI.

Choosing the most appropriate metrics requires that they support the goal of the evaluation. In order to determine whether the IUI prototype supports plan recognition, task support and query resolution, the chosen metrics would need to assist in determining whether the participant looked at the Task Status, System Feedback and User Input sections; and for how long the participant looked at these sections.

Therefore, the metrics that could be of use to the evaluation would be the duration of each fixation and the observation count.

7.4.1.2.1 Usability Measurements

User satisfaction is a subjective measure and was obtained through the use of a post-test questionnaire (NIST 1999). The post-test questionnaire measured user satisfaction by recording the users' reactions using a five-point Likert scale (Barnum 2002). Users were presented with a statement about the IUI prototype (such as, "The system feedback assisted in categorising a problem") and asked the extent to which they agree or disagree with it. This was done by selecting a value from one to five on the Likert scale. A typical Likert scale has the following values (Barnum 2002):

1 = Strongly Disagree
2 = Partially Disagree
3 = Neutral
4 = Partially Agree
5 = Strongly Agree

The post-test questionnaire (Appendix C) used for the evaluation is a modification of the Questionnaire for User Interaction Satisfaction (QUIS) (Chin, Diehl and Norman 1988).

Effectiveness was measured through the use of eye-tracking. The objective eye-tracking data measured the effectiveness of the IUI prototype in terms of whether the participant was able to successfully log and resolve customer queries.

7.4.1.2.2 Eye-Tracking Metrics

The duration of a fixation on an AOI was used to determine how long a participant looked at a particular AOI. This could show the degree of difficulty which the participant had when trying to interpret the information presented in that AOI. Fixation duration data is usually represented by a heatmap.

The observation count metric was used to determine how many times the participant looked at a particular AOI. Use of this metric identified the usefulness of the task support and system feedback provided.

7.4.1.3 Instruments

A test plan was drawn-up for the evaluation (Appendix B) detailing the goals and objectives of the evaluation (Barnum 2002). The test plan contained an overview of the evaluation, the goal of the evaluation, an introduction to the IUI prototype, a tutorial section consisting of three tasks and an evaluation scenario.

The tutorial section of the test plan was scenario-based and guided the participant through each of the required tasks. In order to determine whether the participant could independently solve a customer's query, an evaluation scenario was provided. This scenario did not provide any guidance. The participant was required to complete the three tasks by extracting information from the given scenario. On completion of the test, the participant was required to complete a post-test questionnaire (Appendix C) which required both qualitative and quantitative responses from the participant. Qualitative responses were in the form of general comments. A five-point Likert scale was used to gather quantitative responses.

The post-test questionnaire measured:

- Overall user reactions to the IUI prototype and its use;
- The design of the IUI in terms of its layout, ability to present information and the manner in which the user could switch between various windows;
- The consistency, relevance and obviousness of terminology used and feedback provided;
- The extent to which system feedback was reliable, the amount of information provided, the ability to view more details on demand, and the extent to which system feedback assisted in identifying customers, categorising customer queries and finding a possible solution; and
- The extent to which task support assisted in logging and resolving customer queries.

The structure of the post-test questionnaire differed from the original structure of the QUIS. It was adapted to measure user satisfaction with the use of an IUI. The Overall User Reaction Section (Section B) was used without any modification. Section C of the post-test questionnaire (Interface Design) was modified in order to determine whether the participant was comfortable with the use of an IUI. The System Feedback Section (Section E) was added to measure the participant's reaction and the usefulness of the plan recognition capabilities of the prototype. Section F (Task Support) was added and aimed at determining the usefulness of task-based information during task completion.

Use of the test plan and post-test questionnaire was necessary in order to meet the objectives of the user testing.

7.4.1.4 Procedure

User testing was conducted in the usability laboratory at the Computer Science and Information Systems (CS & IS) Department at the NMMU. The usability laboratory is a controlled environment and consists of an observation and an evaluation room. Users were asked to work through a test plan (Appendix B) and to complete a post-test questionnaire (Appendix C) at the end of the test. This was done in the evaluation room.

The users' actions during the testing process were monitored and captured. Various video-capturing hardware and software in the observation room allowed for these actions to be captured. The evaluation room contained an eye-tracker that was used to capture the users' eye movement during the testing process.

Conducting the usability evaluation involved the execution of the following events:

- The participant, on arrival, was greeted by the test administrator (the author);
- The test administrator seated the participant in front of the computer;
- The participant was then briefed about:
    - The environment in which he/she was seated;
    - The specialised eye-tracking equipment that he/she was about to use;
    - The purpose of the evaluation; and
    - The evaluation process.
- The participant was informed that a test plan and a post-test questionnaire were present and were to be used;
- The purpose of these documents was explained;
- The eye-tracker was calibrated for the participant;
- The participant then commenced with the evaluation. All observations were recorded by the eye-tracker and the test participant was monitored by the test administrator at all times;
- The test participant sat in the evaluation room and the test administrator in the observation room;
- Once the test participant had completed the test plan, he/she was required to fill in the post-test questionnaire;
- A short discussion followed in which the test administrator debriefed the participant by answering any questions;
- The test participant was then thanked for participating; and
- The data captured by the eye-tracker and the data from the post-test questionnaire were gathered and analysed.

7.4.2 Participant Selection

Participants used in a usability evaluation need to match the target user profile (Barnum 2002). The chosen participants were the staff of the NMMU ICT Service Desk and the technical staff at the NMMU Department of CS & IS. Qualifying criteria for participants were that they needed to have had previous experience in solving IT-related queries. The staff from the NMMU ICT Service Desk consisted of first-, second- and third-level support staff. First-level support staff were the call takers, second-level staff the problem solvers and third-level staff the IT technicians. Staff members from the CS & IS Department were IT technicians.

Appendix D represents the demographics of the test participants. Ten participants were selected to take part in the usability evaluation (Nielsen and Landauer 1993; NIST 1999). Eight of the ten participants were from the NMMU ICT Service Desk. The remaining two participants were from the CS & IS Department.

Forty percent (n=4) of the participants were female and represented the population of the ICT Service Desk staff. The remaining 60 percent (n =6) of the sample were males and represented the IT Support Staff. Figures 7.6 and 7.7 represent these statistics.

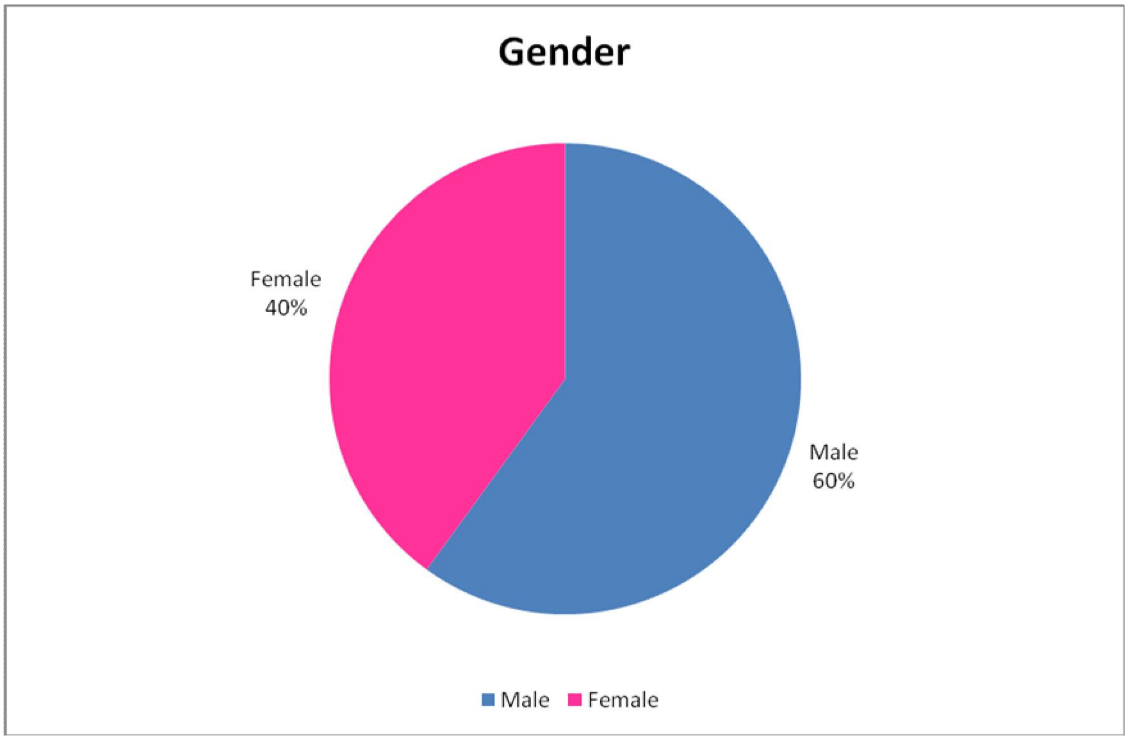


Figure 7.6: Gender Profile of Test Participants (n=10)

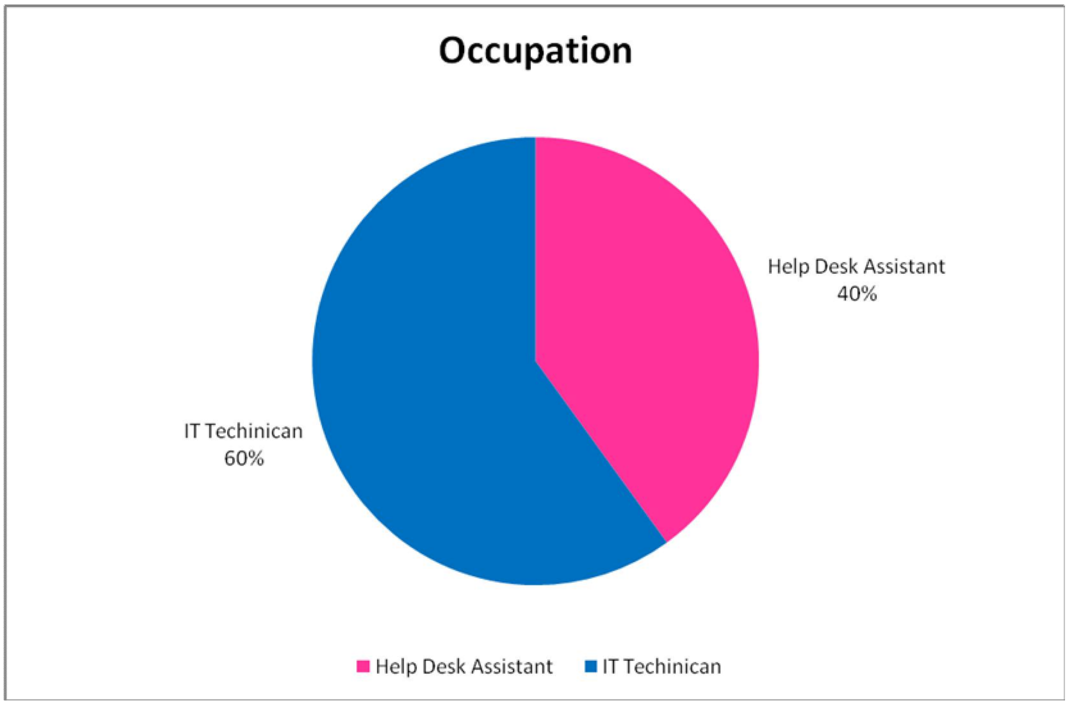


Figure 7.7: Occupational Profile of Test Participants (n=10)

The majority (80%) of the participants were between 24 – 30 years old. The remaining 20% (n=2) were between 35 – 50 years old. None of the participants was older than 50. Figure 7.8 illustrates these statistics.

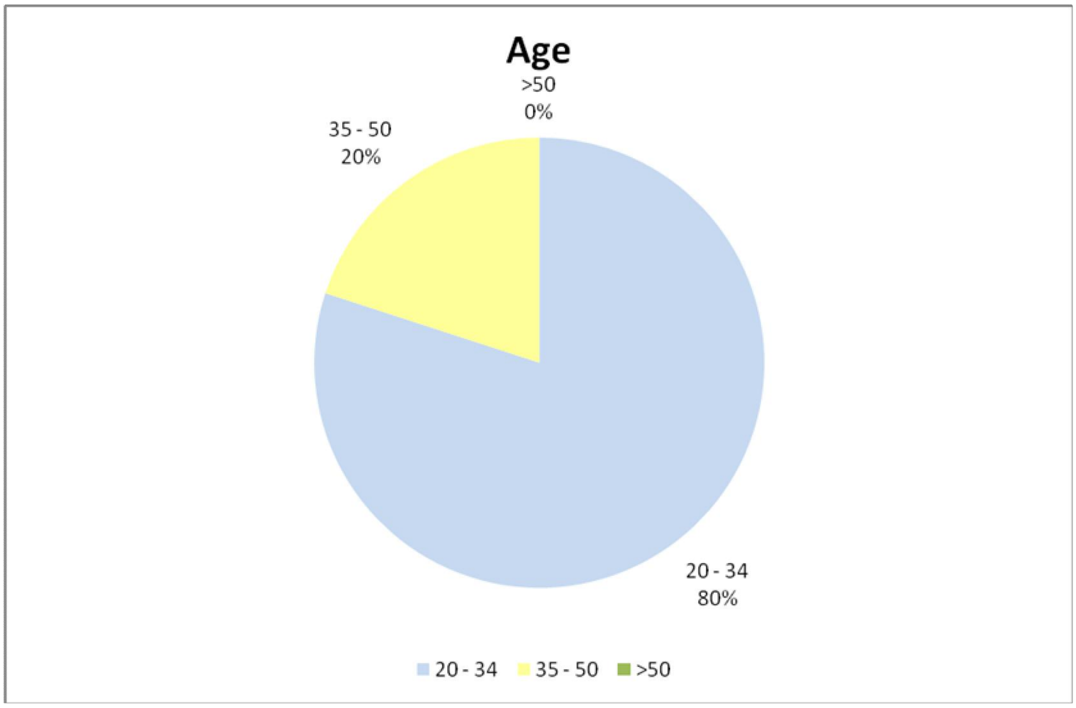


Figure 7.8: Age Profile of Test Participants (n=10)

Sixty percent (n=6) of the test participants had more than four years of computing experience. This 60% can be broken down into 30% (n=3) having five to nine years of experience and the remaining 30% (n=3) having ten or more years of computing experience. The remaining 40% (n=4) of the sample had from one to four years of computing experience. Figure 7.9 illustrates these statistics.

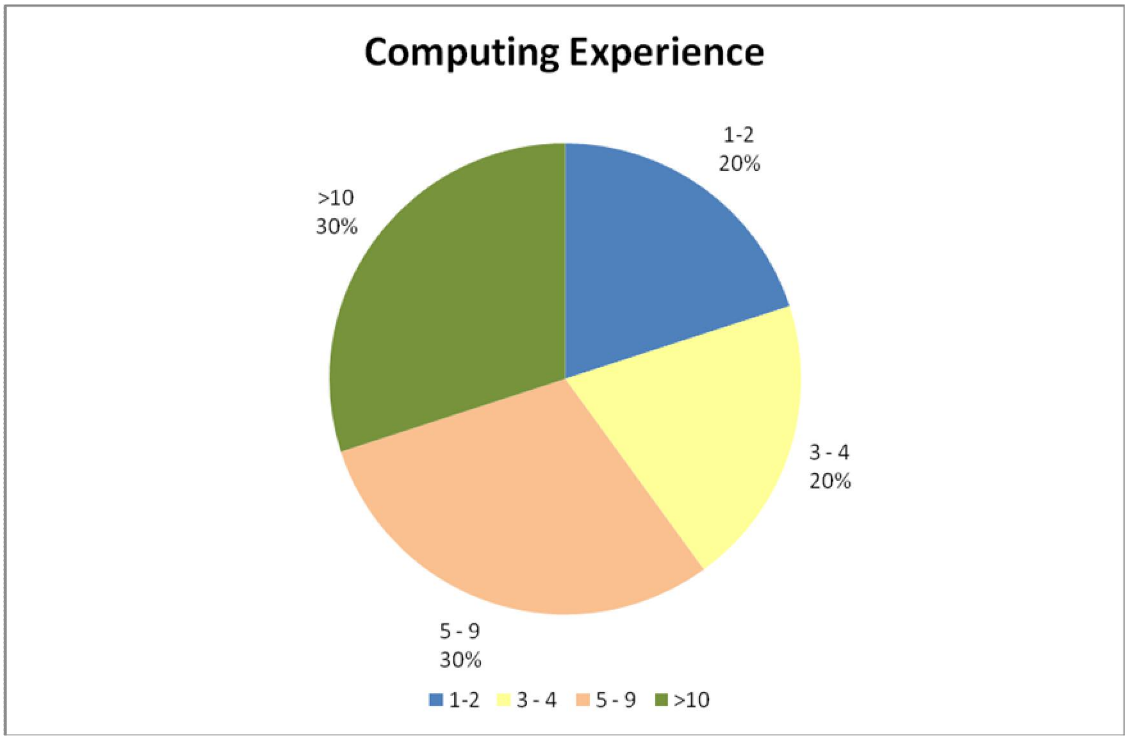


Figure 7.9: Computing Experience Profile of Test Participants (n=10)

7.4.3 Results

Analysis of the post-test questionnaires completed by the test participants was used to measure user satisfaction (Section 7.4.3.1). The post-questionnaire yielded both quantitative (Section 7.4.3.1.1) and qualitative data (Section 7.4.3.1.2). A detailed summary of the quantitative user satisfaction results is given in Appendix E. The results of the qualitative data yielded both positive and negative comments.

To support the user satisfaction results, data gathered from the eye-tracking was analysed and used to determine effectiveness. Comparisons were made in terms of fixation duration and observation count. These comparisons are presented in Section 7.4.3.2.

7.4.3.1 Satisfaction Results

User satisfaction of the IUI prototype was measured by analysing the ratings (Section 7.4.3.1.1) and the general comments (Section 7.4.3.1.2) from the post-test questionnaire (Appendix C).

7.4.3.1.1 Quantitative Analysis

Ratings were given on a five-point Likert scale and rated to each of the usability criteria in the post-test questionnaire. Overall ratings for each criterion were obtained by calculating the mean, median, mode and standard deviation values for all the participants. These values are illustrated in Appendix E. Using a five-point Likert scale, a rating of three would be a neutral rating and can be considered as satisfactory (Section 7.4.1.2.1). A rating above three indicates that there is no cause for concern. However, any ratings below three indicate that the participant was unsatisfied. These ratings are a cause for concern and require extra explanation if significantly different from the rest of the ratings.

Figure 7.10 illustrates the overall satisfaction ratings for the usability criteria specified in the post-test questionnaire. The mean, mode and median values were plotted and were all above four, indicating a high-level of user satisfaction. This shows that the test participants were highly satisfied with the use of the IUI prototype for the purposes of logging and resolving customer queries.

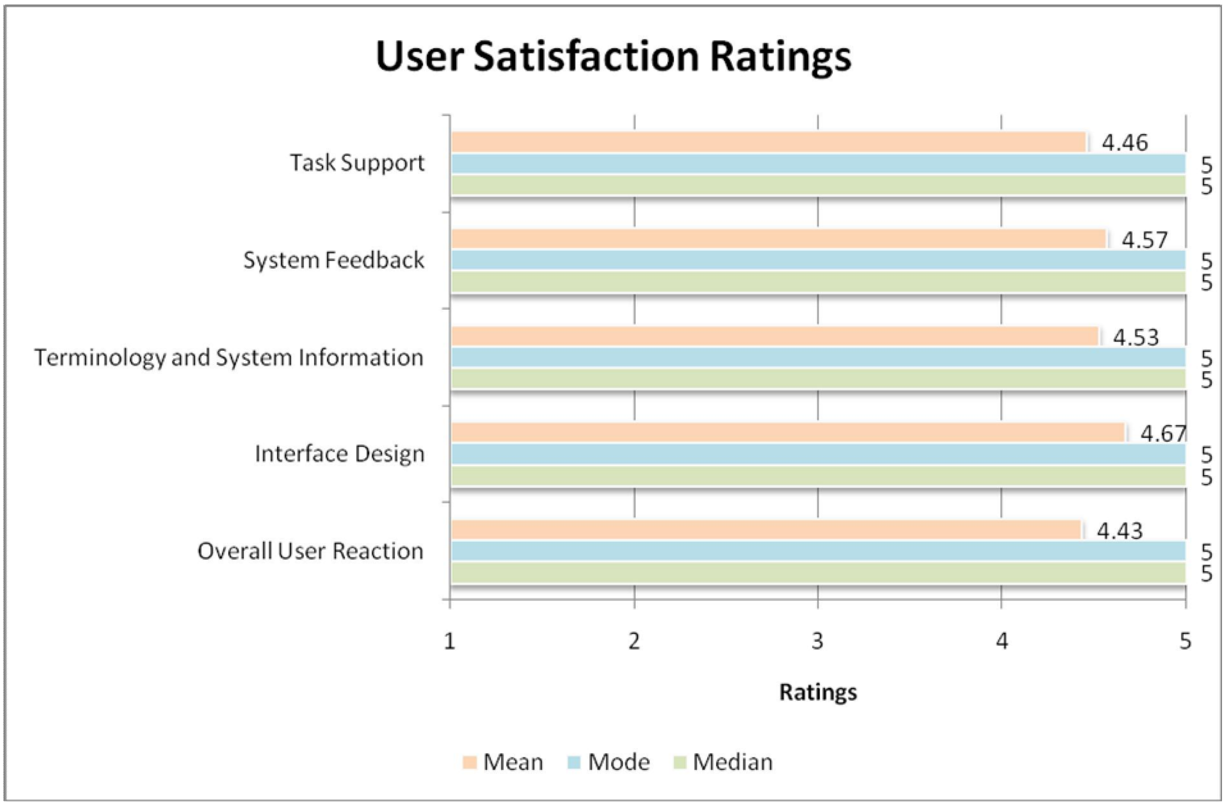


Figure 7.10: User Satisfaction Results (n=10)

The overall user reactions to the use of the IUI prototype are plotted in Figure 7.11. The level of satisfaction achieved by the test participants is indicated by the mean, medium and mode values. These values ranged between four and five indicating that the test participants were highly satisfied with the overall reaction to the use of the prototype. The participants found the IUI prototype flexible, easy to use, satisfying, and that it provided an adequate amount of functionality for the tasks which they had to perform. The test participants also rated the IUI prototype as fairly stimulating and wonderful.

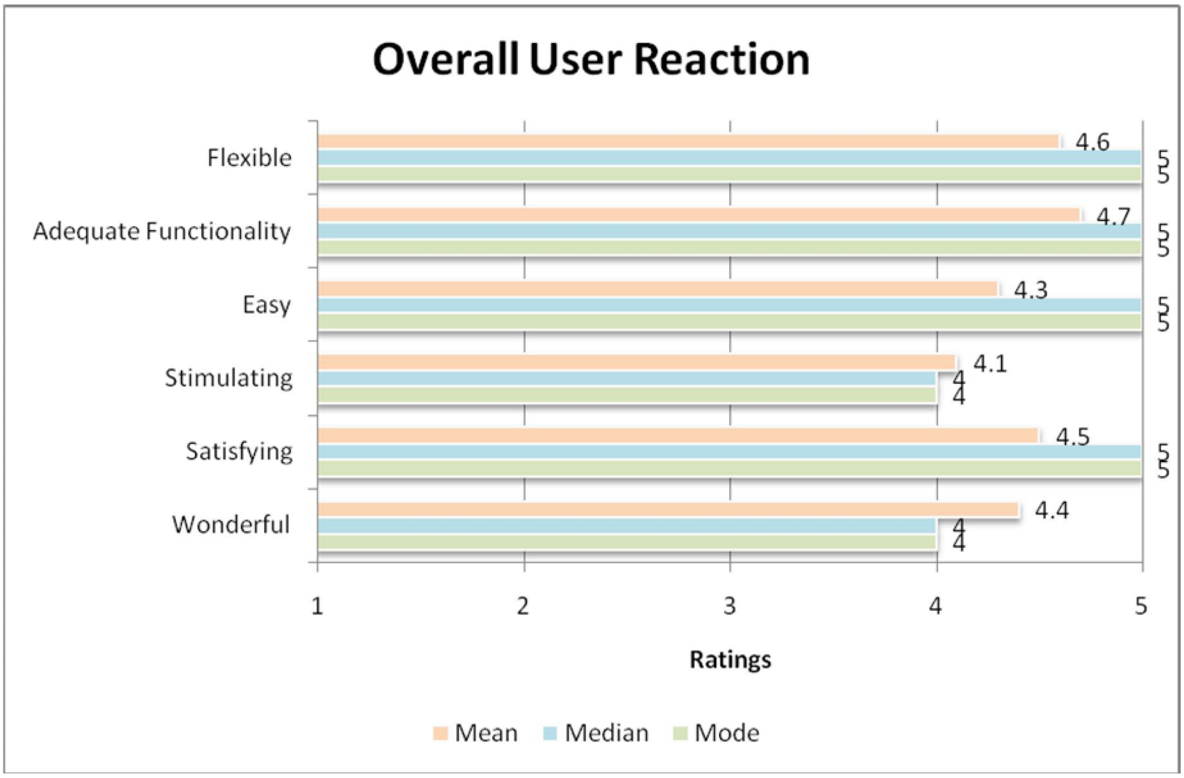


Figure 7.11: Overall User Reaction Results (n=10)

The Interface Design section of the post-test questionnaire aimed to determine whether the test participants were satisfied with the information provided by the IUI prototype, the way in which the interface was designed, the layout of the windows, the ability to switch between various windows, and the amount of customer, incident and problem resolution information displayed. The results of this are illustrated in Figure 7.12. All of the mean, median and mode values were greater than four, indicating that the test participants were highly satisfied with the interface design of the IUI prototype.

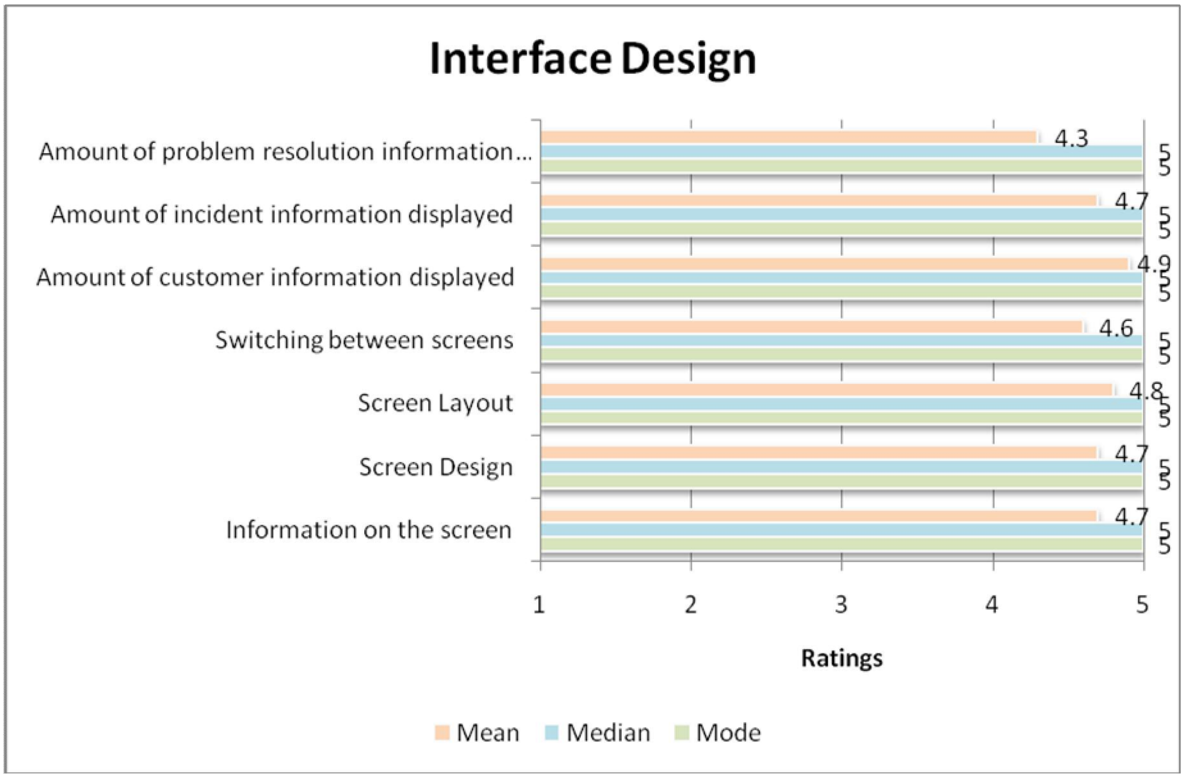


Figure 7.12: Interface Design Satisfaction Results (n=10)

Section D of the post-test questionnaire measured the terminology used and information provided by the IUI prototype. This was achieved by rating the wording of error messages, the obviousness of feedback, relation of the terminology used to the tasks performed and the consistent use of terms. Figure 7.13 indicates the mean, median and mode values for these criteria. All of these values were greater than four, indicating that the test participants were highly satisfied with the use of terminology and the type of system information provided. The mean and median values for the obviousness of feedback provided were slightly lower than the mean and median values of the other criteria. These slightly lower values are explained in the values obtained from the System Feedback Section of the post-test questionnaire (Figure 7.14).

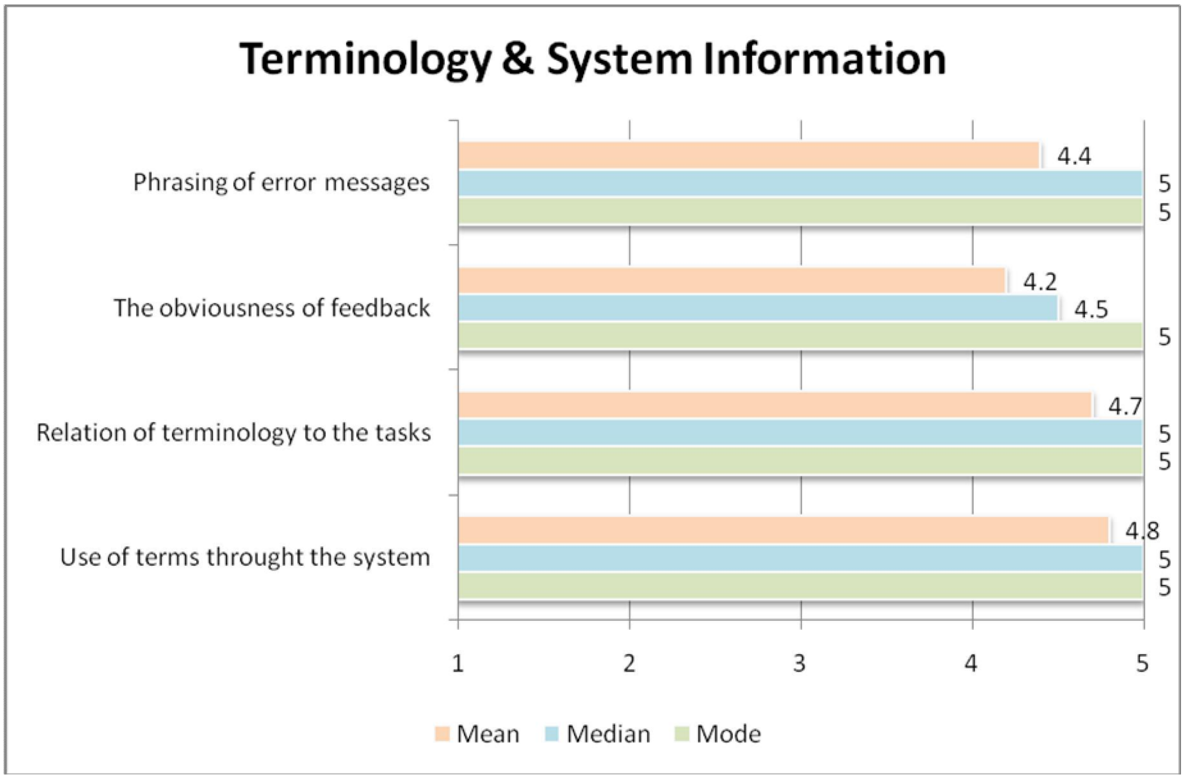


Figure 7.13: Terminology and System Information Satisfaction Results (n=10)

One of the goals of the evaluation was to determine to what extent the IUI prototype supports plan recognition. Plan recognition was provided in the IUI prototype by means of system feedback. Section E of the post-test questionnaire presented various criteria for evaluating the usefulness of the system feedback. The criteria aimed to establish whether the users were satisfied with the reliability of the system feedback, the amount of feedback provided, the ability to view more details on demand and the ability to provide assistance in searching for a customer, categorising a problem and finding a possible solution. Figure 7.14 represents the mean, median and mode values for these criteria. The ratings ranged from four to five, which indicate that the test participants were highly satisfied with the system feedback provided. Slightly lower ratings were obtained for the reliability of the system feedback and the amount of system feedback provided. These ratings are linked to the slightly lower mean and median ratings obtained in the Terminology and System Information Section. The reason for the slightly lower values was due to the limited time in which the test participants interacted with the IUI prototype. The test participants felt that they would become more accustomed to the IUI prototype over a period of time and would be able to develop a greater sense of confidence in the feedback provided. They also felt that the feedback should provide more options in terms of dynamically categorising a customer's query and the number of possible solutions generated.

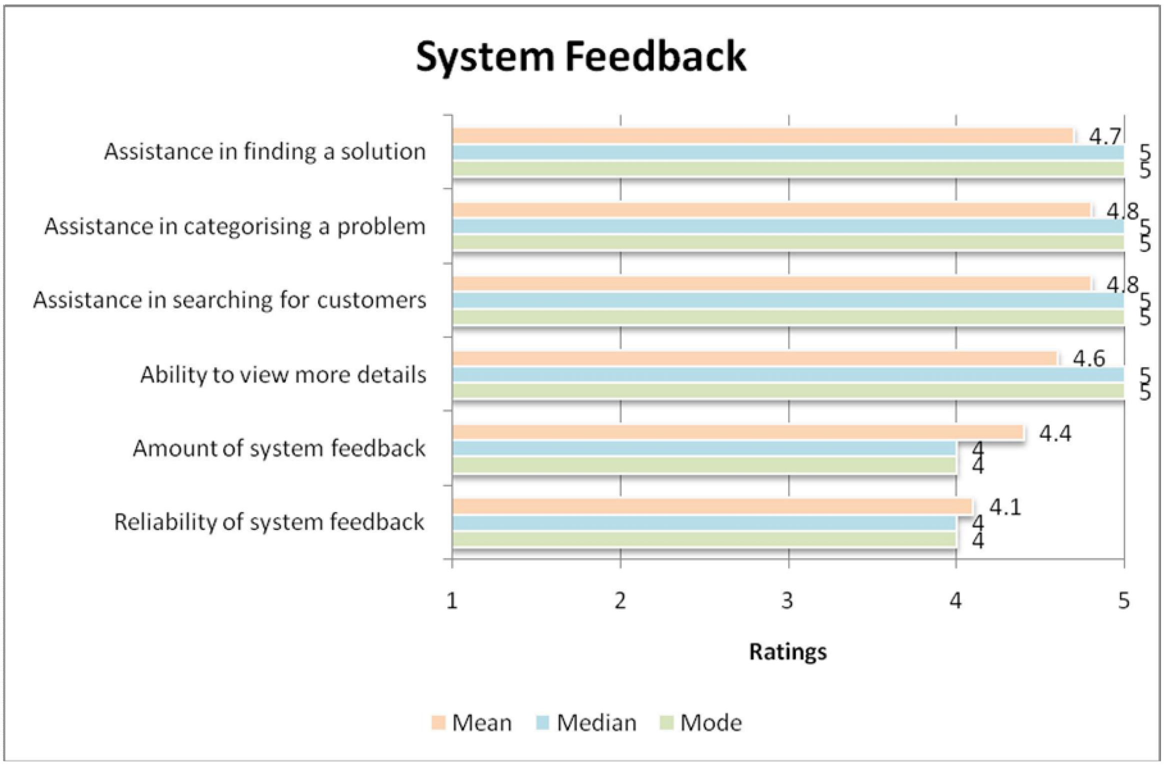


Figure 7.14: System Feedback Satisfaction Results (n=10)

The Task Support section of the post-test questionnaire attempted to determine how satisfied the test participants were with the usefulness of the task-based information provided and the assistance provided in terms of task completion. Other criteria that this section evaluated were the ease of use of the Visual Step Indicator, the indication of task progress and the extent to which the task support was able to facilitate call logging and resolution. The results of this analysis are illustrated in Figure 7.15. All of the mean, median and mode values were greater than four, indicating that the test participants were highly satisfied with the task support provided.

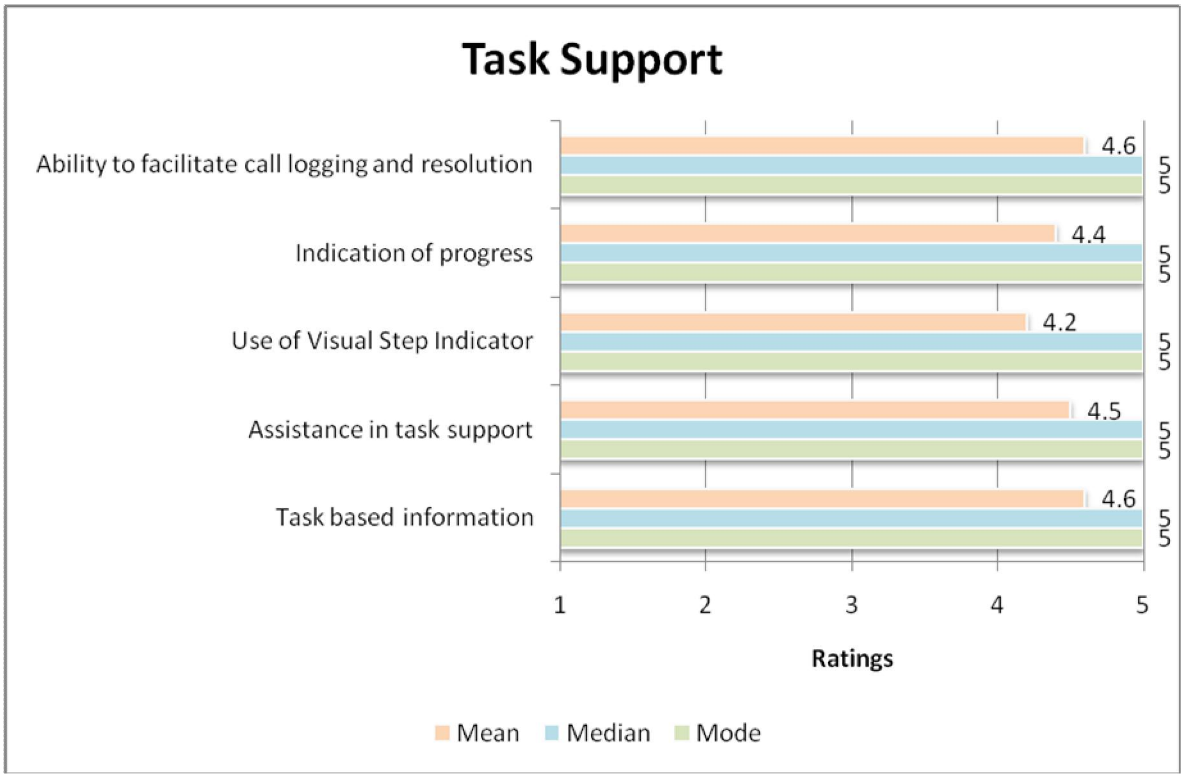


Figure 7.15: Task Support Satisfaction Results (n=10)

7.4.3.1.2 Qualitative Analysis

An analysis was performed on the qualitative data obtained from the post-test questionnaire. This qualitative data was in the form of General Comments (Section G) obtained from the test participants. The analysis resulted in the identification of both negative and positive comments.

7.4.3.1.2.1 Negative Comments

The negative comments received from the test participants related to the time in which they had to interact with the IUI prototype. Participants felt that more time was required before the evaluation in order for the interaction to be smoother and easier during the evaluation.

One of the participants commented that the tooltip, which displays auxiliary information for the candidate results in the System Feedback Section, is not displayed long enough. There are two possible solutions to this: first, the duration for which the tooltip appears could be increased or, secondly, the tooltip could be set to only disappear when the user selects another candidate result or moves away from the System Feedback Section.

Some of the participants felt that more guidance was required, other than the Visual Step Indicator, when the user is stuck and does not know what to do next. A possible solution to this would be to allow the Agent Manager to track the time in which it takes the user to provide a mouse or keyboard event. Based on the length of this time period, assistance could be provided. This could be considered as future work in order to improve the effectiveness of the prototype.

7.4.3.1.2.2 Positive Comments

All of the test participants preferred the prototype to log and resolve customer queries as opposed to the existing HEAT system used at the NMMU ICT Service Desk. This was because they felt that it was easy and practical to use. Some of the positive comments that confirmed this were:

- *"Very easy to use";*
- *"Very user friendly";*
- *"I enjoyed working on this program as it was easy to use"; and*
- *"Program is well presented, easy to use and very practical".*

A key feature that made it very useful was the Query Resolution window (Figure 6.14). Five of the ten participants (50%) commented on the usefulness of this feature. Some of the remarks made about this feature were:

- *"Q n A very useful";*
- *"Liked possible Q n A";*
- *"...especially liked when it gave questions to ask the user..."; and*
- *"Very helpful with possible problems and solutions".*

Both negative and postive comments reflected a positive reaction to the use of the IUI prototype to support query resolution. None of the negative comments criticised the use of task-based information or the ability to provide plan recognition. The ability to resolve customer queries through the use of the Query Resolution window was consistently mentioned as a useful feature of the prototype. The negative comments provided suggestions on how to make the prototype more usable and did not make any mention of the effectiveness of the prototype in terms of logging and solving customer queries. The positive comments highlighted the effectiveness of the prototype through its ease of use and practicality.

Based on the analysis of the feedback provided by the test participants, it can be deduced that a positive reaction was received to the use of the IUI prototype for the purposes of logging and resolving customer queries. This is supported by the quantitative data in Figures 7.10 and 7.11.

This section has discussed how user satisfaction was measured through the analysis of the ratings and general comments obtained from post-test questionnaire. The next section discusses the effectiveness of the IUI prototype. This was done by analysing the data gathered from the eye-tracker for each participant.

7.4.3.2 Effectiveness Results

The effectiveness of the prototype was measured by assessing the completion rate of the tasks for the evaluation scenarios (Appendix F). A second measure of effectiveness was obtained through the use of eye-tracking. The use of eye-tracking identified whether the intelligent components of the Query Diagnosis window (Figure 6.8) were used. These components are the Task Status Section and the System Feedback Section.

Test participants were initially asked to work through a tutorial in order to familiarise themselves with the IUI prototype. The tutorial was included as part of the test plan (Appendix B). There were three major tasks that needed to be completed in order for the test participant to successfully log and resolve a call (Section 2.6). These tasks were to:

a) Identify the customer;
b) Categorise and classify the customer’s query; and
c) Offer a possible solution and capture these details.

The participants were guided through each of these steps and required to answer questions based on the different tasks. The total task completion rate was 100% for all of the participants (Figure 7.16).

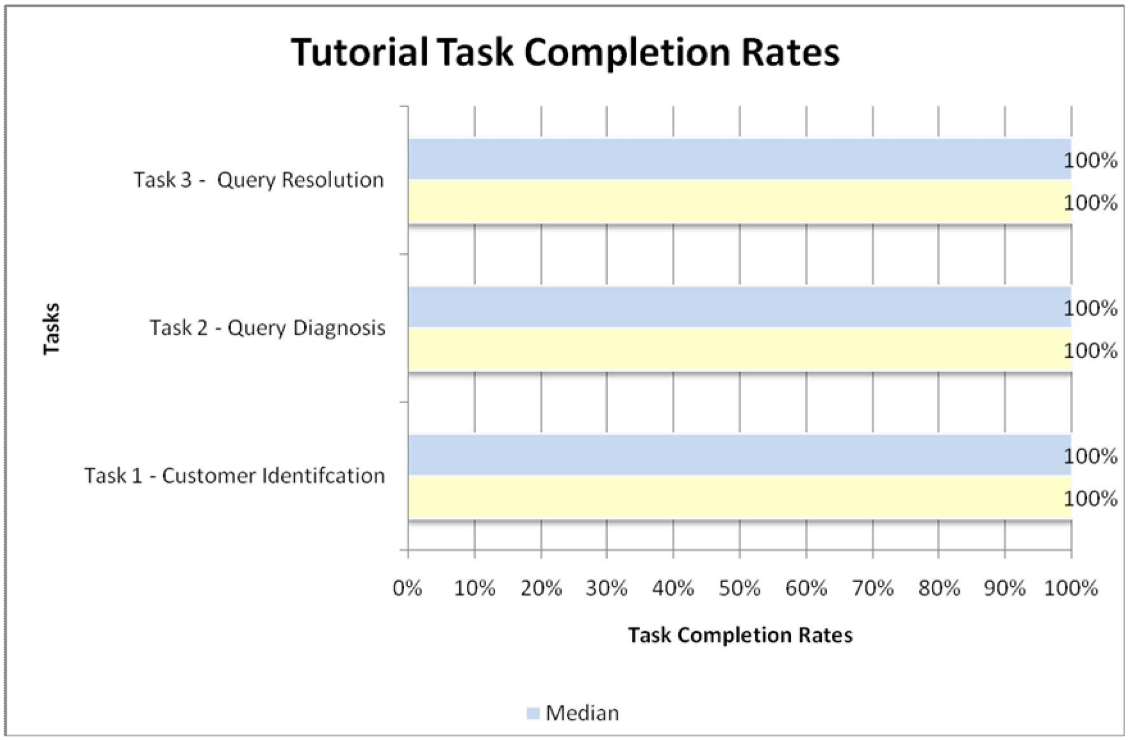


Figure 7.16: Tutorial Task Completion Rates Results (n=10)

The evaluation section of the test plan presented a scenario to the test participants. This scenario required that the test participants complete the required tasks without any guidance. The test participants were able to complete the evaluation scenario, resulting in a 100% completion rate. These results are illustrated in Figure 7.17.

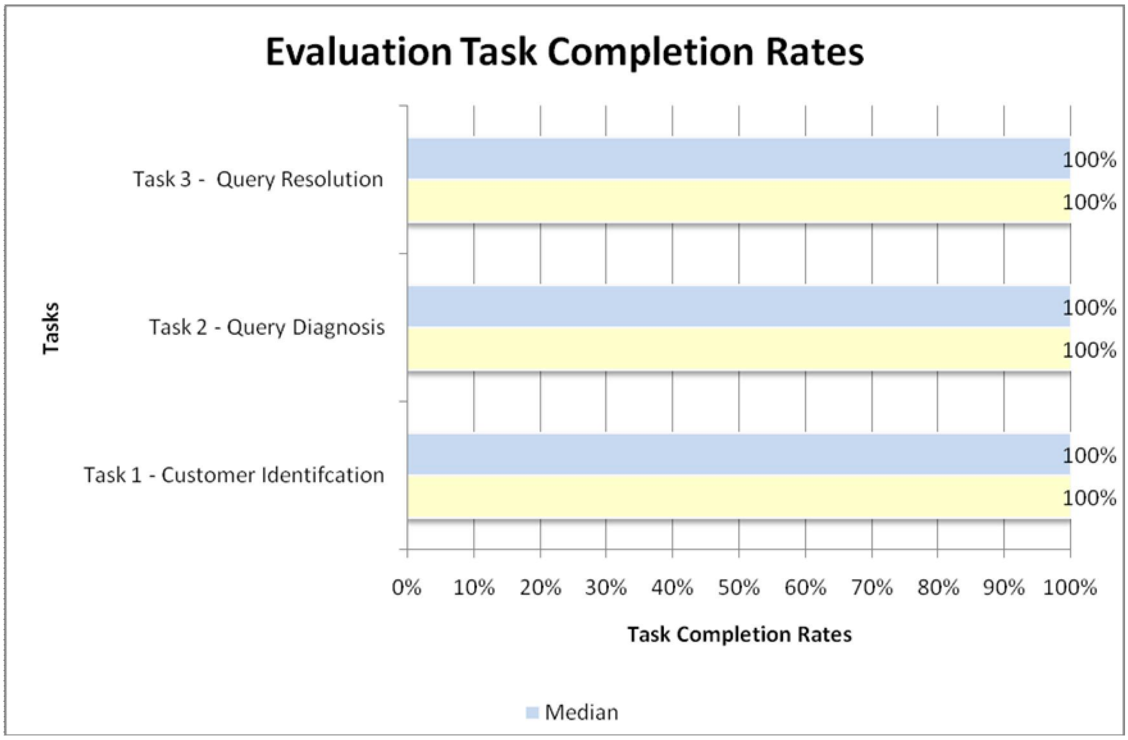


Figure 7.17: Evaluation Task Completion Rates Results (n=10)

Being able to successfully log and resolve a customer's query is essential in proving the effectiveness of the prototype. It is also one of the goals of this evaluation – to determine the extent to which a user can successfully log and resolve a customer's query. The task completion rates from the guided tutorial and the evaluation scenario were 100%; this show that the test participants were able to:

a) Identify the customer;
b) Categorise and classify the customer's query; and
c) Offer a possible solution and capture these details.

This proves that the IUI prototype effectively supports the tasks of call logging and resolution.

Measuring effectiveness was also done through the use of eye-tracking. Eye-tracking was used to determine to what extent the intelligent components namely, the Task Status Section and the System Feedback Section of the Query Diagnosis window (Figure 6.8) were used. All of the results presented are based on the test participants working through the test plan (Appendix B). AOIs needed to be defined before any analysis could be done. These areas are illustrated in Figure 7.5.

Fixation duration illustrates how long the user looked at a particular AOI. Through the use of a heatmap, fixation duration can also determine whether an AOI was used or not. The duration of the fixation is represented through the use of colour. As the duration increases, the colour changes from a light green to a bright red. Figure 7.18 represents the heatmap for the tutorial.

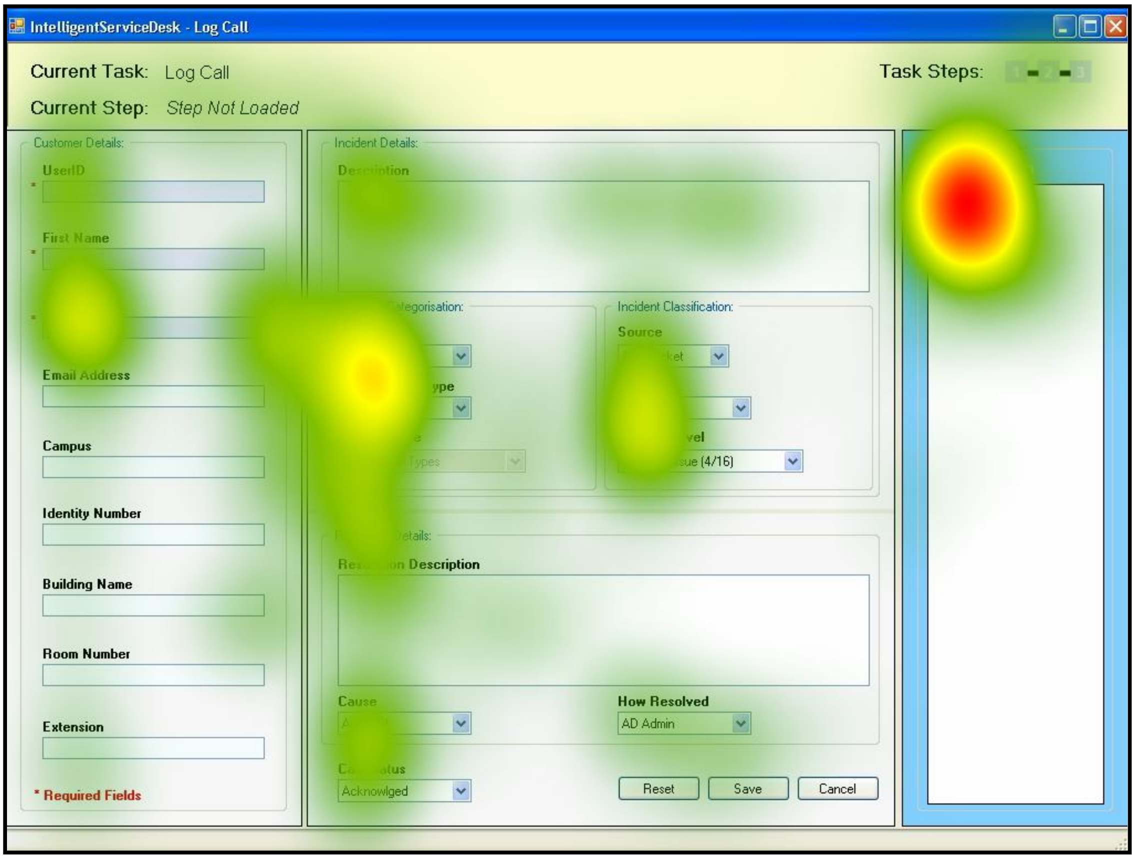


Figure 7.18: Tutorial Heatmap – Total Fixation Time (n=10)

The heatmap reveals that the test participants looked at all of the AOIs during the tutorial. According to the heatmap, more time was spent in the User Input Section as opposed to System Feedback and Task Status sections. This assumption is supported in Figure 7.19 which represents the mean and median fixation duration times for the various AOIs. The results show that the User Input Section has the highest mean values and the Task Status Section the lowest. This implies that the test participants spent more time trying to categorise and classify the query opposed to receiving task support. These values are tabulated in Appendix G.

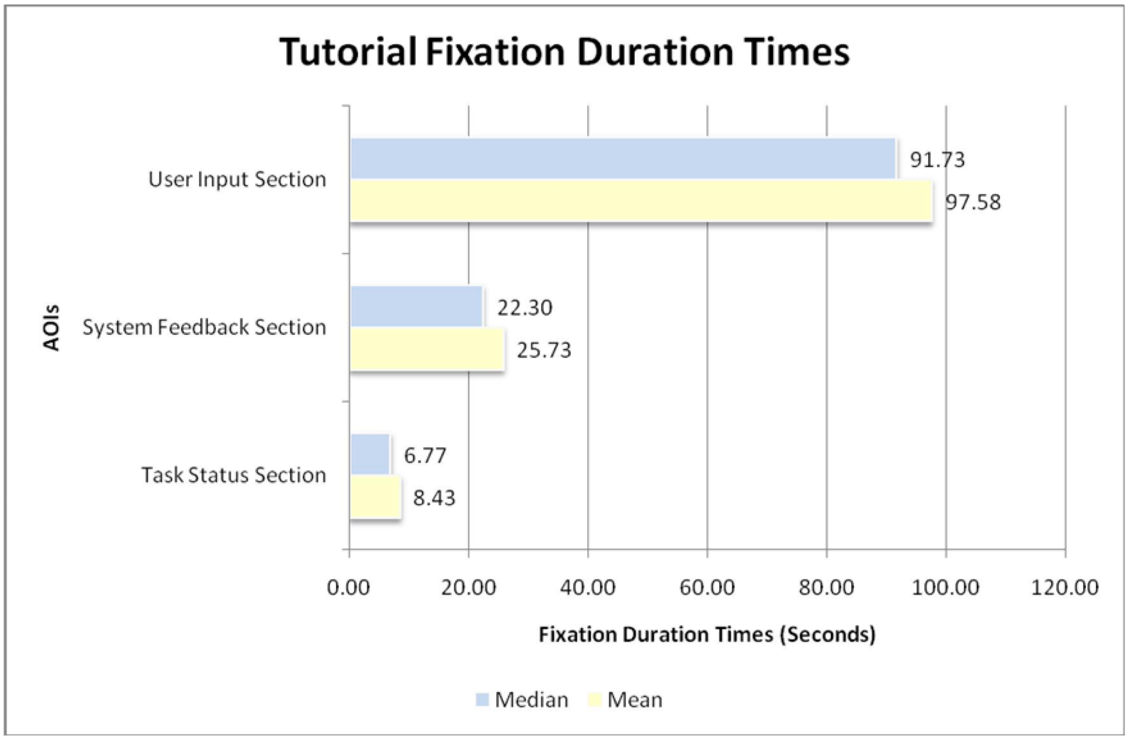


Figure 7.19: Tutorial Fixation Duration (n=10)

To further support these results, the observation count data of the various AOIs were analysed (Appendix G). Although directed through the tutorial, these results could provide an indication of how many times during the tutorial the test participants looked at an AOI. These results are illustrated in Figure 7.20.

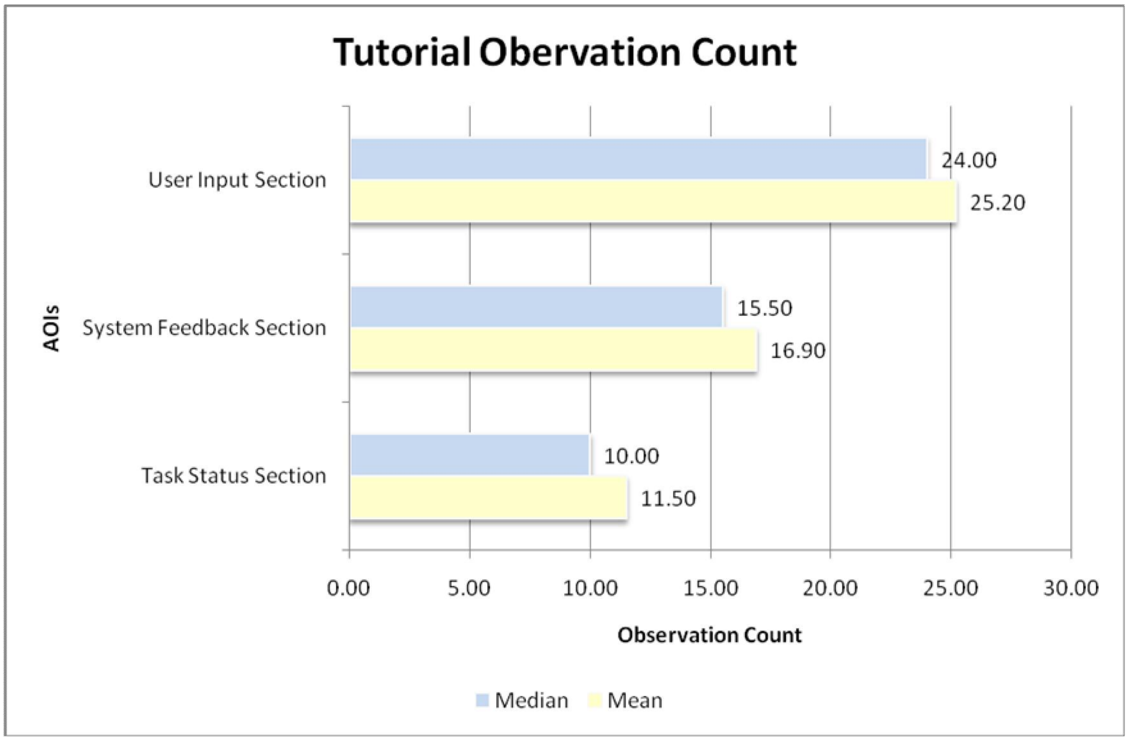


Figure 7.20: Tutorial Observation Count (n=10)

The mean and median values (Figure 7.20) of the Task Status Section supports the intensity of the hotspot in Figure 7.18. The average number of times the Task Status Section was looked at by the test participants (throughout the duration of the tutorial) was 11.5. Further analysis of the data revealed that the test participants looked more at the System Feedback Section as opposed to the Task Status Section. The section which received the most attention was the User Input Section.

The next step is to report on the eye-tracking results for the evaluation scenario. The evaluation scenario did not guide the test participants through each of the required tasks. Test participants were required to read the scenario and complete the various tasks in a similar manner to that of the tutorial without guidance.

Figure 7.21 is a heatmap of the evaluation scenario. It illustrates that, in comparison to the tutorial heatmap (Figure 7.18), less time was spent looking at the Task Status Section. More emphasis was spent on the User Input Section, with respect to categorising and resolving the problem.

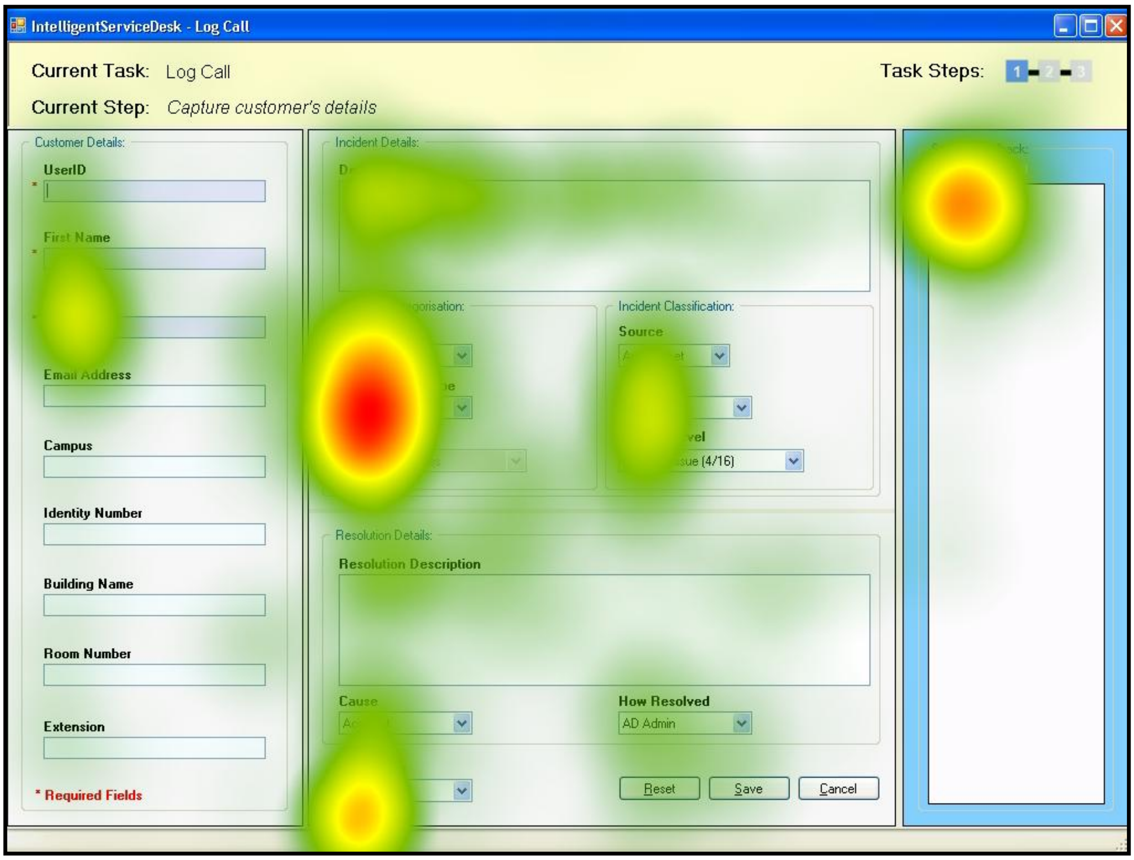


Figure 7.21: Evaluation Heatmap – Total Fixation Time (n=10)

A comparison of the mean and median values in Figure 7.19 and Figure 7.22 reveals a significant change in fixation duration. There was a 65% reduction in the time at which the user was fixated on the User Input Section, 38% for the System Feedback Section and a 7% reduction in Task Status Section. These values were calculated as following:

$$\%Change = \frac{\text{Evaluation Mean}}{\text{Tutorial Mean}} \text{ x } 100$$

The low mean values in Figure 7.22 indicates that the test participants spent little or no time looking at the Task Status Section. These values are tabulated in Appendix G. The low mean rating also supports the little or no evidence of hotspots in the Task Status Section in Figure 7.21.

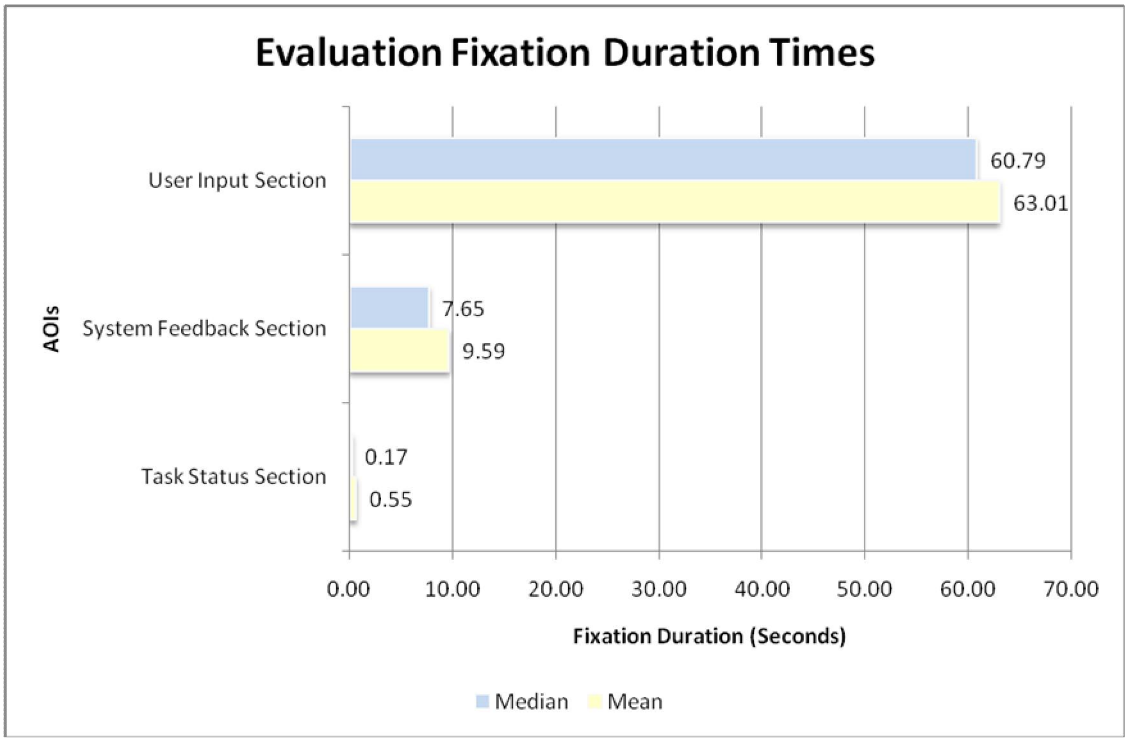


Figure 7.22: Evaluation Fixation Duration (n=10)

The total observation counts of the AOIs were lower in the evaluation scenario than the tutorial. These results are illustrated in Figure 7.23. The mean and median observation count values in Figure 7.23 are also lower than those for of the tutorial in Figure 7.20. The Task Status Section was observed, on average, 15% less in the evaluation as opposed to the tutorial times. A decrease in the observation count of the System Feedback Section by 84% shows that the test participants were familiar with its purpose and knew how it could be used to assist with task completion. There was also a 45% reduction in the total number of times the User Input Section was looked at in the evaluation as opposed to the tutorial. These values are tabulated in Appendix G.

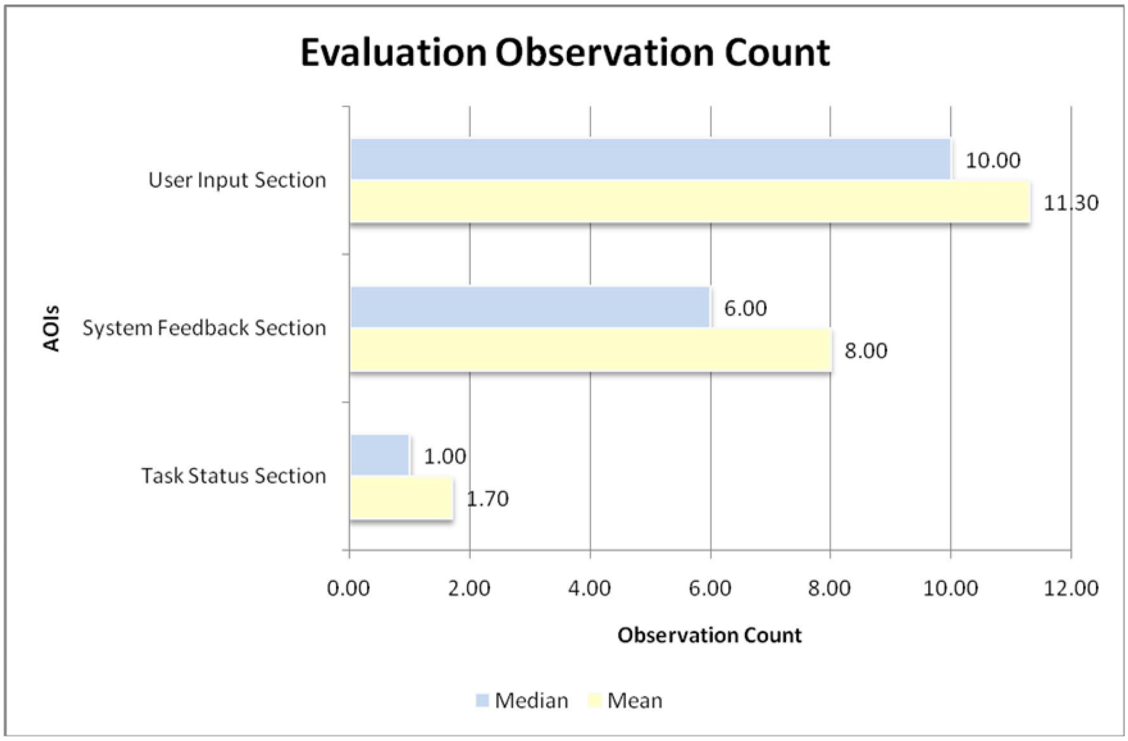


Figure 7.23: Evaluation Observation Count (n=10)

The Task Status Section proved to be of little use for the test participants. Figure 7.23 shows a mean value of 1.7. This implies that the test participants looked at the Task Status Section at most twice, during the evaluation. This shows that the Task Status Section had a minimal influence on task completion. These results compliment the lack of hotspots present in the Task Status Section in Figure 7.21.

Analysis of the eye-tracking data showed that the test participants looked at all of the AOIs when completing the tutorial and the evaluation scenario. Although directed, the tutorial produced results that were inflated as the test participants were instructed to look at a particular AOI in order to complete the task. The results of the evaluation scenario were unbiased and the test participants could complete the scenario in any manner. Analysing the results of the evaluation scenario highlighted that the Task Status Section was seldom referenced. This means that the ability to deliver task-related information did not affect task completion rates. Extensive use of the System Feedback Section was evident. However, a lot more time was spent in the User Input Section

7.5 Implications

The results of the model evaluation (Section 7.3) and the user testing (Section 7.4) were discussed. These results suggested changes that could improve the model and the prototype. This section will outline and discuss these suggestions.

7.5.1 Model Evaluation Suggestions

Analysis of the model evaluation results revealed that the IUI prototype lacks feedback when a particular step has been completed. This is suggested as the IUI prototype was not able to answer the question "***When** did I/We/You do?*" (Table 3.1). A possible solution address this shortcoming is to append the completion date and time to the tooltip for each step in the Visual Step Indicator. This information could be stored in the User Model and monitored by the Agent Manager.

The ability for the user to provide feedback as to why a step has not been completed is not currently present in the IUI prototype. Having this functionality would address the question "***Why** did You/We (not) do?*" (Table 3.1). A popup could be provided to address this issue. The information provided by the user could be saved to the User Model and could be analysed later for in order to improve completion rates.

7.5.2 User Testing Suggestions

An analysis of the qualitative data revealed that there is a need for intelligent assistance. This assistance would need to be in the form of additional guidance. The guidance would be able to track when a user is taking too long on a particular step or field and offer assistance. This functionality could be provided in the form of an interface agent (i.e. Clippy).

7.6 Conclusions

The evaluation outlined and discussed in this chapter showed that through the IUI prototype implemented in Chapter 5, the proposed IUI model (Chapter 4) could aid the development of IUIs for CCs. The IUI prototype's ability to support plan recognition, task support and query resolution was also shown.

Currently no formal methods and metrics exist for evaluating IUIs. To overcome this limitation, an evaluation of the proposed IUI model (Section 7.3) and user testing (Section 7.4) were conducted.

Results from the model evaluation showed that the interface of the prototype is intelligent. The prototype was able to answer most of the questions specified in Table 3.1 and displayed the characteristics of a typical IUI.

The second objective of this chapter was to determine to what extent the prototype supported plan recognition, task support and query resolution. User testing (Section 7.4) was used to evaluate the overall user satisfaction (Section 7.4.3.1.1) obtained from using the prototype. The effectiveness (Section 7.4.3.2) of the prototype, in terms of being able to log and resolve customer queries, was also measured and whether the intelligent components of the interface were used or not.

Data for the user testing was gathered by allowing the test participants to work through a test plan (Appendix B) and then answer the questions in a post-test questionnaire (Appendix C). The post-test questionnaire was modified to address the characteristics of an IUI and to acquire feedback that could assist in determining whether the IUI prototype supported plan recognition, task support, and query resolution.

Participants for the user testing comprised the NMMU ICT Service Desk Staff and IT Support Staff at the NMMU (Section 7.4.2). The NMMU ICT Service Desk Staff represents the first-level support of a CC and the IT Support Staff is representative of the second- and third-level support.

Results of the user testing showed that the test participants were highly satisfied with their interaction with the prototype. Test participants were able to complete all of the required tasks for both the tutorial and the evaluation scenarios. The use of eye-tracking allowed for determining the effectiveness of the intelligent components (the Task Status Section and the System Feedback Section) of the interface. An analysis of this data showed that the Task Status Section was used the least and did not influence task completion rates. Plan recognition, through the use of the System Feedback Section, was extensively used and welcomed. This was supported by the users being highly satisfied with the System Feedback Section's ability to assist with task completion.

Qualitative feedback in the form of general comments supported the quantitative results (Section 7.4.3.1.2). The request for additional guidance was also identified through the analysis of the qualitative data (Section 7.4.3.1.2). Test participants felt that additional guidance could serve as a secondary instructor when a user does not know how to complete a task. Positive comments from the qualitative analysis revealed that the test participants enjoyed their experience with the IUI prototype and found it very easy to use.

The results of the evaluation did not reflect any implications that would require a change or modification of the model. Several suggestions were made that could possible improve the usability of the IUI prototype (Section 7.5).

The next section concludes this dissertation, summarising its contribution and achievements, as well as outlining possibilities for future research.

Chapter 8:

Conclusions and Recommendations

8.1 Introduction

The aim of this dissertation was to develop an IUI model for CCs that could potentially improve CRR at the first level of support. A model was proposed and evaluated - through the implementation of an IUI prototype, an evaluation of the model and an usability evaluation of the IUI prototype. The objective of this chapter is to revisit the objectives of this research, discuss the achievements and contributions made and to highlight any recommendations that could lead to future research.

8.2 Research Objectives Revisited

Chapter 1 outlined the objectives for this research (Section 1.3.3). The main objective was to develop an IUI model for CCs that could assist in improving CRR at the first level of support. Achieving this objective required that secondary research objectives be established. These secondary objectives were:

- Identification of the factors influencing CRR and the current process of logging and resolving a customer's query (Chapter 2);
- Determining how IUIs could be used to address these problems (Chapter 3);
- Evaluation of existing IUI models and how they could be specialised to the domain of CCs (Chapter 4);
- Specialising an existing IUI model for the domain of CCs (Chapter 5);
- Determining how the proposed IUI model could be implemented (Chapter 6) and evaluated (Chapter 7); and
- Highlighting the achievements and contributions made by this research and outlining any recommendations that could lead to future work (Chapter 8).

Each of these objectives were decomposed into research questions (Table 1.1). All of the secondary objectives were met and were discussed in the previous chapters.

8.3 Research Achievements

The goal of this research was to develop an IUI model for CCs. This model would assist in potentially improving the CRR at the first-level of support. This goal was realised, both theoretically and practically. Theoretically this goal was met by designing and proposing an IUI model that could be used to develop software solutions for CCs (Section 8.3.1). Practically, an IUI prototype was implemented and evaluated in order to determine whether the proposed IUI model met its theoretical objectives (Section 8.3.2). These objectives were to determine to what extent the proposed model aids the development of IUIs for CCs, and the extent to which the IUI prototype supports plan recognition, task support and query resolution. Both of these objectives were met.

8.3.1 Theoretical Achievements

A core component of this research was to develop a model. Model-based design of IUIs was researched in order to determine the requirements of a model and how it could be evaluated (Chapter 4). The first stage to proposing an IUI model for CCs was to determine if any existing IUI models could be specialised. A set of criteria was established in order to evaluate existing IUI models and architectures (Table 4.3). The IUI Architecture (Tyler *et al.* 1991) was selected as it was the most generalised model that could be specialised for the domain of CCs. The second stage to proposing the model was to specialise it for the domain of CCs (Chapter 5). Specialising the architecture to the domain of CCs required that the knowledge residing within the knowledge base be specific to the domain of CCs.

The selected IUI architecture (Tyler *et al.* 1991) possessed some limitations (Chapter 5). These limitations were that user's low-level goals could not be inferred, knowledge base models were specified without any formal standard or structure and no IUI design template was specified. These limitations were overcome. Furthermore, the Adaptor component of the selected architecture was renamed to the Agent Manager, due to its autonomous nature. A new model was added to the Knowledge Base, namely the Solution Model. This model stores solutions in order to deliver query resolution capabilities at the first-level of support. The design of an IUI model for CCs was discussed in terms of its three elements (Chapter 5):

- An Architectural Design element;
- A Component-Level Design element; and
- An Interface Design element.

8.3.2 Practical Achievements

An IUI prototype was constructed based on the design on the proposed IUI model. Only three of the five components, namely the Plan Manager, Agent Manager and Knowledge Base were implemented. This is because the remaining two components (the Input/Output Manager and Presentation Manager) do not directly support the diagnosis and resolution of customer queries. Implementation of the proposed model showed that the prototype possessed plan recognition capabilities and the ability to provide intelligent feedback. Further testing in the form of user testing was performed in order to determine the usability of the IUI prototype.

Currently, no formal methods or metrics exist to evaluate IUIs. To overcome this limitation, a model evaluation and user testing were performed. The purpose of the model evaluation was to determine whether the IUI prototype could be regarded as an IUI. The results of this evaluation were positive and it can be concluded that an interface designed using the proposed model can be regarded as an IUI. User testing showed that the users were highly satisfied with the prototype. Eye-tracking identified that users utilised the System Feedback Section (intelligent feedback) but did not pay much attention to the Task Status Section (task-based information). A qualitative analysis of user feedback showed that users felt that the IUI prototype was easy to use and very practical.

8.4 Research Contributions

Accomplishing the objectives set out by this research resulted in various contributions. These contributions can be divided into theoretical and practical contributions. The theoretical contributions apply directly to the field of IUIs. Practical contributions resulting from this research apply to the domain of CCs.

8.4.1 Theoretical Contributions

The first major theoretical contribution is an IUI model for CCs (Figure 8.1). This model consists of three elements, namely the architecture (Section 5.3.1), component-level design (Section 5.3.2) and interface design (Section 5.3.3). Figure 8.1 is repeated for ease of reference.

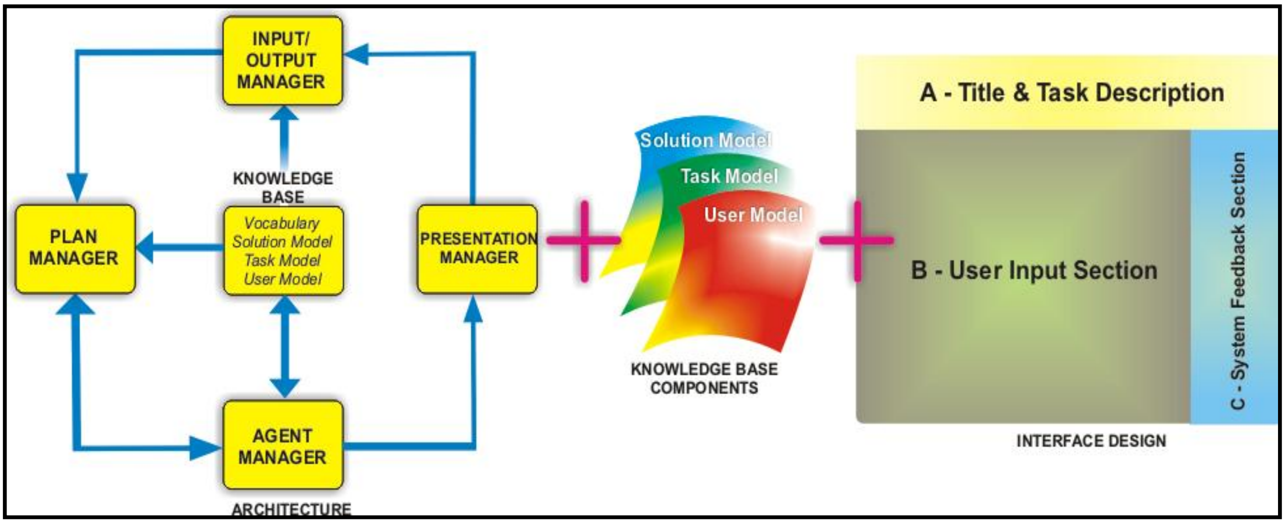


Figure 8.1 : IUI Model for CCs

The architecture of the proposed model (Figure 8.1) is a modified version of the selected IUI Architecture (Figure 5.2) proposed by Tyler *et al.* (1991). The selected IUI architecture possessed the limitation of being unable to infer the user's low-level goals. This limitation was overcome by allowing the Plan Manager to access the Task Model directly from the Knowledge Base. A second modification included the renaming of the Adaptor component to the Agent Manager due to the autonomous nature of this component. The Solution Model was added to the architecture in order to facilitate query resolution at the first-level of support. The Solution Model contained questions and answers to specific types of queries. These questions could be asked to the customer and the answers provided in an attempt to resolve the customer's query.

The IUI Architecture (Tyler *et al.*1991) described the roles of the various models residing within the Knowledge Base. However, no formal specifications or structures for these models were described. Therefore, the second theoretical contribution of this research included a specification for the User Model (Section 5.3.2.2), Task Model (Section 5.3.2.1) and the Solution Model (Section 5.3.2.3). In order for the proposed IUI model to be specific to the domain of CCs, the knowledge residing within the Knowledge Base needed to be specific to CCs. Therefore, the information contained with the Task and Solution models is specific to the domain of CCs.

A third contribution that was made was the interface design template (Section 5.3.3) of a typical IUI. This template included a section for task-based information, IA feedback and a direct manipulation section (Figure 5.7). The IUI Architecture (Tyler *et al.*1991) did not specify the layout of a typical IUI.

An investigation into existing methods and instruments for evaluating IUIs revealed that no formal methods and instruments existed. A model evaluation was proposed in order to evaluate the interface of the IUI prototype possessed intelligence (Section 7.3). This evaluation required that an IUI be answerable to the questions in Table 3.1 and possess the typical characteristics of an IUI (Section 3.3). User testing was also conducted as part of the evaluation (Section 7.4).

The last theoretical contribution that was made was a set of evaluation criteria for IUI models (Table 4.3).

8.4.2 Practical Contributions

Evaluation of the proposed IUI model was done through the implementation of an IUI prototype and a usability evaluation. Only three of the five components of the model were implemented. These were the Plan Manager, Agent Manager and Knowledge Base. The remaining two components were not implemented as these components do not directly support the processing of logging and resolving customer queries.

Implementation of the IUI prototype was done to show that the proposed model could be used to develop IUIs for CCs. This was evaluated through a model evaluation (Section 7.3). Results of the model evaluation were positive and confirmed that that the interface design of the prototype possessed intelligence and that it could therefore be regarded as an IUI.

The interface design of the IUI prototype consisted of two windows: a Query Diagnosis window (Figure 6.8) and a Query Resolution window (Figure 6.14). These windows were necessary in order to support the key processes of logging and resolving a customer's query within a CC. The two windows were designed in accordance to the proposed IUI design template (Section 5.4). Intelligence added to the windows facilitated the delivery of task-based support in the Task Status Section (Section 6.3.1.1) and intelligent feedback in the System Feedback Section (Section 6.3.1.3).

Intelligent feedback was offered in order to reduce the total time it takes to identify a customer, categorise and classify the customer's query and to offer a possible resolution. This was made possible through the use of pattern-matching. Pattern-matching was used to match partial strings entered by the user to keywords stored in the Knowledge Base. This provided support for dynamically categorising the customer's query (Section 6.3.1.3). CCAs could enter the query description as given by the customer and the IA (running in the background) will try and match words from the description to those stored in the Knowledge Base. Based on the categorisation of the query, possible solutions are offered.

In order to provide problem-solving capabilities for CCAs at the first-level of support, a Query Resolution window was provided (Figure 6.14). The purpose of this window was to provide the CCA with a list of questions for the customer. These questions are retrieved from the Solution Model. Based on the response provided by the customer, either the next question is enabled or the answer for current question is displayed.

Implementation of the various Knowledge Base components was done in XML. This allowed for a degree of extensibility and flexibility. The Knowledge Base components can be integrated into existing applications or into the design of new IUI applications for CCs.

Quantitative data, from the usability evaluation, revealed that the users were highly satisfied with the use of the IUI prototype. Qualitative data revealed that the users found the IUI prototype easy and practical to use. The ability to ask customers questions and to receive feedback was rated very highly as this is lacking in current CC software. The use of eye-tracking showed that the users found the intelligent feedback provided in the System Feedback Section very useful. These results supplemented the results from the analysis of the post-test questionnaire.

8.5 Recommendations

8.5.1 Recommendations for Theory

Literature identified that the slow rate of progress in the field of IUIs is a result of the lack of emphasis on the study of IUI models (Section 4.7). This research has specified an IUI model for the domain of CCs (Chapter 5). This model provides an architecture to build IUI applications and the specifications of User and Task models which are generalised and could be applied to any domain. The low-fidelity IUI design template is also generic and could be used to build IUIs for any domain.

Very little has been published on the evaluation of IUIs (Section 7.2). This research proposed an evaluation method comprising of a model evaluation and user testing in order to evaluate an IUI. This evaluation methodology could be adopted to evaluate the usefulness and usability of IUIs in future.

8.5.2 Recommendations for Practice

The proposed model provides developers with a complete set of specifications which can be used to build IUI applications for CCs (Chapter 5).

8.5.3 Recommendations for Future Work

Several opportunities exist for future research. Some of these possibilities are discussed below.

The analysis of the User Model to assist in routing a query to the most qualified and available CCA could be investigated. This could reduce the time in which it takes to resolve a query. Developing an adaptive UI based on the use of User Model could also be investigated. This could provide interfaces that are personalised and customised according to the user's preferences and behaviour.

Only three of the five model components were implemented. Future work could be to implement the remaining two components and investigate the benefits these would present to CCs. The remaining two components are the Input/Output Manager and the Presentation Manager. The roles of these components are to provide multimodal and adaptive capabilities. Providing multimodal capabilities could increase the accessibility of the application. Furthermore, the use of multiple modalities could deliver interaction that is more natural. Implementation of the Presentation Manager would allow for outgoing data to be examined and to determine the most appropriate modality for that data. The Presentation Manager would determine these modalities based on the preferences and behaviour of the individual user (stored in the User Model).

Future work could also include the investigation of other ways of presenting system feedback in order to make the feedback more obvious. The same could be applied to the

Task Status Section considering that this section was the least used during the usability evaluation.

An investigation into other ways of dynamically building and updating the Solution Model could be conducted. This could provide a greater amount of flexibility when trying to address a customer's query. A learning engine could assist is this regard by referring to both the customer's history and the current query to dynamically build a customer-specific solution.

A longitudinal study could be performed. This study could investigate the results of extended use of an IUI within a CC environment. Results emanating from this study could reveal whether IUIs have a long-term positive impact on the user satisfaction, operational costs and CRR of a CC.

8.6 Summary

This research has successfully met its objectives. The outcomes of this research have resulted in a theoretical contribution to the field of IUIs and a practical contribution to the domain of CCs. Future work has outlined various possibilities which could provide improvements to the model and IUI prototype.

References

Aagedal, J. and Mohagheghi, P. (2007): Evaluating Quality in Model-Driven Engineering. *International Workshop on Modeling in Software Engineering (MISE)*. IEEE Computer Society. 2007. 6-11.

Almskog and Frydman Communications (1996): Call Centre 2000: The future is here [online]. Available at: http://www.callcentres.com.au/DMA2000.htm (Last Accessed: 9 March 2006).

Analysis of Customer Metrics (ACM) Diagnostics (2005): Help Desk–Call Centre or Dispatch Centre [online]. Available at: http://www.acmdiagnostics.com/links/download/wp_3_helpdesk.pdf (Last Accessed: 20 July 2006).

Anton, J. (2004): Combining Service Excellence with Profitability [online]. Available at: http://www.asponline.com/egain_anton.pdf (Last Accessed: 10 September 2006).

Aspect Communications Corporation (2004): How Do Your Customers Rate Your Call Centre? [online]. Available at: http://www.aspect.com/mm/pdf/products/pstn/Aspect_Cust_Rate_CC_wp.pdf (Last Accessed: 14 March 2006).

Australian National Audit Office (1996): Telephone Call Centres [online]. Available at: http://www.anao.gov.au/WebSite.nsf/Publications/4A256AE90015F69B4A25691200 06FCD1 (Last Accessed: 14 March 2006).

Axio Systems (2006): Assyst Help Desk & Incident Management [online]. Availabe at: http://www.axiossystems.com/six/shared/downloads/pdf/Help_Desk.pdf (Last Accessed: 10 September 2006).

Avaya, Inc. (2006): Call Center vs. Contact Center [online]. Available at: http://www.avaya.com/gcm/master-usa/en-us/wowtopics/callctrvscontactctr/callctrvscontactctr.htm (Last Accessed: 14 March 2006).

Barnum, C.M. (2002): *Usability Testing and Research*. Longman.

Berenbach, B. and Konrad, S. (2007): Putting the "Engineering" into Software Engineering with Models. *International Workshop on Modeling in Software Engineering (MISE)*. IEEE Computer Society. 2007. 4-9.

Chai, J. Y., Hong, P. and Zhou, M. X. (2004): A probabilistic approach to reference resolution in multimodal user interfaces. *Proceedings of the 9th International Conference on Intelligent User Interfaces,* Funchal, Madeira, Portugal, January 13-16, 2004. IUI '04. New York, NY, ACM Press. 70-77.

Chin, J.P., Diehl, V.A. and Norman, K.L. (1988): Development of an instrument measure user satisfaction of the human computer interface. *Proceedings of SIGCHI'88,* New York, NY, ACM Press. 213-218.

Cohen, P. R. (1992): The role of natural language in a multimodal interface. *Proceedings of the 5th Annual ACM Symposium on User Interface Software and Technology,* Monteray, California, United States, November 15-18, 1992. UIST '92. New York, NY, ACM Press. 143-149.

Coninx, K. and Van den Bergh, J. (2004): Model-Based Design of Context-Sensitive Interactive Applications: a discussion of Notations. *TAMODIA 2004,* Prague, Czech Republic, November 15-16. ACM. 2004.

Contact Centre University (2000): Glossary of Contact Centre Terms [online]. Available at: http://www.ctifourm.com/technology/tcs/glossary/pdf (Last Accessed: 14 March 2006).

Davis, R. and Maxwell, S. (2004): The comprehensive IT help desk: a single point-of-contact for the university community. *Proceedings of the 32nd Annual ACM SIGUCCS Conference on User Services,* Baltimore, MD, USA, October 10-13, 2004. SIGUCCS '04. New York, NY, ACM Press. 37-39.

Dimension Data (2006): Merchants Global Contact Centre Benchmarking Report 2006. Dimension Data: 83–202.

E-Communication Advantage (ECA) (2006): Call Center Terminology [online]. Available at: http://www.eca.com/index.php?option=com_content&task=view&id=51&Itemid=85 (Last Accessed: 21 March 2006).

Edmonds, E. (1993): The future of intelligent interfaces: not just “how?”, but “what?” and “why?”. *Proceedings of the 1st International Conference on Intelligent User Interfaces*, Orlando, Florida, United States, January 04-07, 1993. W. D. Gray, W. E. Hefley, and D. Murray, (Eds.). IUI '93. New York, NY, ACM Press. 13-17.

eGain (2005a): 7 Habits of Highly Effective Contact Centers and Help Desks [online]. Available at: http://bacs.uq.edu.au/intranet/ICT/egainwhitepaper7habits.pdf (Last Accessed: 10 September 2006).

eGain (2005b): 5 Contact Center Megatrends and How to Ride Them [online]. Available at: http://www.ccng.com/files/public/eGain_Whitepaper_Contact_Centers_Megatrends.pdf (Last Accessed: 10 Septmber 2006).

Frantz, M.G. (2006): The Evolution of the IT Help Desk to the Service Desk [online]. Available at: http://www.compucom.com/wwwimages/Corp/Images/CompuComServiceDeskWhitePaper0106.pdf (Last Accessed: 10 July 2006).

FrontRange Solutions (2006): HEAT Product Suite [online]. Available at: http://www.frontrange.co.za/ProductsSolutions/SubCategory.aspx?id=35&ccid=6 (Last Accessed: 10 September 2006).

Fusion, J. (2004): Improving contact center effectiveness and agent performance through process integration [online]. Available at: http://www.jacada.com/WhitePapers/pdfs/Frost-Sullivan.pdf (Last Accessed: 10 September 2006).

Galitz, W.O. (2002): *The Essential Guide to User Interface Design*. Second Edition, John Wiley & Sons, Inc.

Gans, N., Koole, G. and Mandelbaum, A. (2003): Commissioned Paper-Telephone Call Centres: Tutorial, Review and Research Prospects [online]. Manufacturing & Service Operations Management, **5**(2): 79-141. Available at: http://www.helpdesknews.net/CALLCENTRES/LIVE/Resources/Documents/TelephoneCallCenters.pdf (Last Accessed: 20 March 2006).

Gliedman, C. (2005): Thirty-One Best Practices for the Service Desk [online]. Available at: http://i.i.com.com/cnwk.1d/html/itp/31_Best_Practices_Service_Desk.pdf (Last Accessed: 10 September 2006).

Gomaa, H. and Hussein, M. (2007): Model-Based Software Design and Adaptation. *International Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE Computer Society. 2007. 7-16.

Goodrich, M.T. and Tamassia, R. (2002): *Algorithm Design*. John Wiley & Sons, Inc.

Hefley, W. E. and Murray, D. (1993): Intelligent user interfaces. *Proceedings of the 1st International Conference on Intelligent User Interfaces*, Orlando, Florida, United States, January 04-07, 1993. W. D. Gray, W. E. Hefley, and D. Murray, (Eds.). IUI '93. New York, NY, ACM Press. 3-10.

Höök, K. (2000): Steps to take Before Intelligent User Interfaces Become Real. *Journal of Interaction with Computers*, **12**(4):409-426.

InStranet, Inc. (2003): The Top 10 Information Challenges in Today’s Call Center [online]. Available at: http://www.inst-informatica.pt/v20/cid/biblioteca_digital/crm/CRM_The%20Top%2010%20Information%20Challenges%20in%20todays%20call%20center.pdf (Last Accessed: 30 May 2006).

ISO9241-11 (1998): Ergonomic Requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability, ISO. **Part 11**.

ITIL (2006): Service Desk [online]. Available at: http://www.itil.org/itil_e/index_e.html (Last Accessed: 30 May 2006).

Jameson, A. (2003): Adaptive interfaces and agents. In *the Human-Computer interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, J. A. Jacko and A. Sears, (Eds.). Human Factors and Ergonomics. Lawrence Erlbaum Associates, Mahwah, NJ, 305-330.

Jennings, N. R. and Wooldridge, M. J. (1998): Applications of Intelligent Agents [online]. Available at: http://www.cs.umbc.edu/agents/introduction/jennings98.pdf (Last Accessed: 25 September 2006).

Kaish, S. (2004): Challenges of the Customer Care Centre Enhancing Service and Reducing Cost with Cosmocall Universe [online]. Available at: http://www.cosmocom.com/WhitePapers/whitepaper-TeleCustomerCare.htm (Last Accessed: 14 March 2006).

Kass, R. and Finin, T. (1991): General user modelling: a facility to support intelligent interaction. In *Intelligent User Interfaces*, J. W. Sullivan and S. W. Tyler, (Eds.). New York, NY, ACM Press. 111-128.

Kobsa, A. and Wahlster, W. (1989): User Models in Dialog Systems. In *User Models in Dialog Systems*, Kobsa, A. and Wahlster, A. (Eds.). Springer-Verlag, Inc. New York. 4-34.

Koelle, D. (2004): Intelligent User Interfaces [online]. Availabe at: http://www.davekoelle.com/intint-outline.jsp (Last Accessed: 26 March 2006).

Lesh, M. , Rich, C. and Sider, C.L. (2001): Using Plan Recognition in Human-Computer Collaboration [online]. Available at: http://www.merl.com/papers/docs/TR98-23.pdf (Last Accessed: 25 September 2006).

Lieberman, H. (1997): Autonomous interface agents. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Atlanta, Georgia, United States, March 22-27, 1997. S. Pemberton, (Ed.). CHI '97. New York, NY, ACM Press. 67-74.

Lieberman, H. (2001): Interfaces That Give Advice. In *Human-Computer Interaction in the New Millenium*. Carroll, J.M (Ed.). ACM Press/Addison-Wesley Publishing Co. 475-485.

Lieberman, H and Selker, T. (2003): Agents for User Interfaces. In *Handbook of Agent Technology*. Bradshaw, J. (Ed.). MIT Press.

LiveTime (2006): Help Desk Overview [online]. Available at: http://www.livetime.com/webservicedesk/HelpDesk.html (Last Accessed: 10 September 2006).

Luck, M., Ashri, R. and D'Inverno, M. (2004): Agent – Based Software Development. Artech House, Inc, Norwood, MA.

Maybury, M.T. (1993): Introduction. In *Intelligent Multimedia Interfaces*, M.T. Maybury, (Ed.). Cambridge, MA, MIT Press. 1-7.

Maybury, M.T. (1998): Intelligent User Interfaces : An Introduction. M. T. Maybury and W. Wahlster, (Eds.). In *Readings in Intelligent User Interfaces*. Morgan Maybury Publishers Inc, 1-13.

Maybury, M.T. (1999): Intelligent user interfaces: an introduction. *Proceedings of the 4th International Conference on Intelligent User Interfaces,* Los Angeles, California, United States, January 05-08, 1999. IUI '99. New York, NY, ACM Press. 3-4.

Maybury, M.T. (2001): Intelligent Interfaces for Universal Access: Challenges and Promise [online]. Available at: www.mitre.org/work/tech_papers/tech_papers_01/maybury_intelligent/maybury_intelligent.pdf (Last Accessed: 30 August 2006).

Mehrotra, V. and Fama, J. (2003): Call Center Simulation Modeling: Methods, Challenges, and Opportunities.S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, (Eds.). *Proceedings of the 2003 Winter Simulation Conference*, Piscataway, New Jersey: Institute of Electrical and Electronics Engineers. 135-143.

Microsoft (2002): Service Management Functions: Service Desk [online]. Available at: http://www.microsoft.com/technet/itsolutions/cits/mo/smf/smfsvcdk.mspx (Last Accessed: 20 March 2006).

Nielsen, J. and Landauer, T.K. (1993): A Mathematical Model of the Finding of Usability Problems. *Proceedings of the Conference on Human Factors in Computing Systems*, Amsterdam, The Netherlands. New York, NY, ACM Press. 206-213.

NIST (1999): Common Industry Format for Usability Testing Reports.

http://www.nist.gov/iusr (Last Accessed: 14th October 2007).

Olivier, M.S. (2006): *Information Technology Research-A practical guide for Computer Science and Informatics.* Pretoria, Second Edition. Van Schaik Publishers.

Oviatt, S. (1999): Ten myths of multimodal interaction. In *Communications of the ACM* **42**(11):74-81. November 1999.

Oviatt, S., Coulston, R. and Lunsford, R. (2004): When do we interact multimodally?: cognitive load and multimodal communication patterns. *Proceedings of the 6th International Conference on Multimodal Interfaces,* State College, PA, USA, October 13 - 15, 2004. ICMI '04. New York, NY, ACM Press. 129-136.

Oviatt, S. and Cohen, P. (2000): Perceptual user interfaces: multimodal interfaces that process what comes naturally. In *Communications of the ACM* **43**(3):45-53.March 2000.

Parbhoo, S. (2002): CCMA's National Call Centre [online]. Available at: http://www.ccma.org.za/DisplayNews2.asp?ID=8 (Last Accessed: 9 March 2006).

Polar Software (2006): Polar Help Desk : Comparison [online]. Available at: http://www.polarsoftware.com/products/servicedesk/comparisons.asp (Last Accessed: 10 September 2006).

Preece, J., Rogers, Y., and Sharp, H. (2007): *Interaction Design: beyond human-computer interaction*. Section Edition, John Wiley & Sons, Inc.

Pressman, R.S. (2005): *Software Engineering-A Practitioner's Approach.* Sixth Edition. McGraw-Hill.

Pretorius, M.C., Calitz, A.P. and van Greunen, D. (2005): The added value of eye-tracking in the usability evaluation of a network management tool. *Proceedings of the 2005 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologist,* White River, South Africa, September 20-22, 2005. 1-10.

Puerta, A. R. (1993): The study of models of intelligent interfaces. *Proceedings of the 1st International Conference on Intelligent User Interfaces*, Orlando, Florida, United States, January 04-07, 1993. W. D. Gray, W. E. Hefley, and D. Murray, (Eds.). IUI '93. New York, NY, ACM Press. 71-78.

Ramachandran, A. and Young, R. M. (2005): Providing intelligent help across applications in dynamic user and environment contexts. *Proceedings of the 10th International Conference on Intelligent User Interfaces,* San Diego, California, USA, January 10-13, 2005. IUI '05. New York, NY, ACM Press. 269-271.

Rich, E. (1999): Users are individuals: individualizing user models. *Int. J. Hum.-Comput. Stud.* **51**(2):323-338. August 1999.

Rich, C., Sidner, C.L., and Lesh, N. (2001): Collagen: applying collaborative discource theory to human-computer interaction. *AI Mag,* **22**(4):15-25.

Robertson, J. (2005): Two types of in-bound call centres [online]. Available at: http://www.steptwo.com.au/papers/cmb_twocallcentres/pdf/CMb_TwoCallCentres.pdf (Last Accessed: 14 March 2006).

Ross, E. (2000): Intelligent User Interfaces: Survey and Research Directions [online]. Available at: www.cs.bris.ac.uk/Publications/Papers/1000447.pdf (Last Accessed: 24 July 2006).

Sanderson, S. (2003): *Introduction to Help Desk Concepts and Skills*. Osborne McGRaw-Hill.

Serio (2006): Serio Service Desk Software [online]. Available at: http://www.seriosoft.com/service_desk_software.htm (Last Accessed: 10 September 2006).

Shneiderman, B. and Maes, P. (1997): Direct manipulation vs. interface agents. In *Interactions* **4**(6):42-61. November 1997.

Siebel Systems (2005): Contact Center Analytics: Driving Greater Efficiency, Higher Revenue, and Increased Customer Satisfaction [online]. Available at: http://www.03-ibm.com/solutions/businesssolutions/sible/doc/content/bin/sible_Contact_Center_March2005_4.pdf (Last Accessed: 9 March 2006).

SiteHelpDesk.com (2006): SiteHelpDesk-IT [online]. Available at: http://www.sitehelpdesk.com/IT/Service-Management.htm (Last Accessed: 10 September 2006).

Steel, A. (2003): Understanding and Enhancing Call Centre Computer – Human – Human Interaction. *Proceedings of Conference on Human Factors in Computing Systems*, Florida. New York, NY, ACM Press. 2003. 690-691.

Trompf, M. (2001): Human-Centered Computational Intelligence in Future Telecommunications [online]. Available at: http://www.erudit.de/erudit/events/tc-d/16_04_99/3_trompf.pdf (Last Accessed: 15 August 2006).

Tyler, S. W., Schlossberg, J. L., Gargan, R. A., Cook, L. K. and Sullivan, J. W. (1991): An intelligent interface architecture for adaptive interaction. In *Intelligent User Interfaces*, J. W. Sullivan and S. W. Tyler, (Eds.). New York, NY, ACM Press. 85-109.

de Vrieze, P.T., van Bommel, P. and van der Weide, Th. P. (2004): A Generic Engine for User Model Based Adaptation [online]. *Proceedings of the 8th ERCIM Workshop – User Interfaces for All,* Palais Eschenbach, Vienna, Austraia, June 28-29, 2004. Available at: http://ui4all.ics.forth.gr/workshop2004/files/ui4all_proceedings/adjunct/methodologies/88.pdf (Last Accessed: 10 September 2006).

Waern, A. (1997): What is an Intelligent Interface? [online]. Available at: http://www.sics.se/~annika/papers/intint.html (Last Accessed: 29 July 2006).

Wahlster, W. (1991): User and discourse models for multimodal communication. In *Intelligent User Interfaces*, J. W. Sullivan and S. W. Tyler, (Eds.). New York, NY, ACM Press. 45-67.

Wooldridge, M. J. and Jennings, N. R. (1995): Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, **10**(2):115-152. Available at: http://eprints.ecs.soton.ac.uk/2102/01/KE-REVIEW-95.pdf (Last Accessed: 10 September 2006).

Appendix A:

Field Study Questionnaire

Department of Computer Science and Information Systems

Contact Centre Questionnaire

Student: Akash Singh **Supervisor: Prof. Janet Wesson**

{Akash.Singh , Janet.Wesson}@nmmu.ac.za

First Name: ____________________________________

Surname: _____________________________________

Company: _____________________________________

Position: ______________________________________

Date: ____

1. Roles (User) Questions:

- Describe your role at this contact centre (agent)?

- Describe the role of the agents at this contact centre (management).

- What other stakeholders, beside management and agents, have a direct influence on this contact centre (management)?

2. How (Daily Activities) Questions:

- What do your daily activities entail at this particular contact centre?

- Briefly describe how customer queries are handled at this contact centre?

- Briefly describe how software is used to support the management of this contact centre and its daily operations (management)?

- What are the goals that this contact centre tries to achieve on a daily basis?

3. Computer Assistance & Support Questions:

- What software is available to assist with resolving customers queries?

- How many different modes of interaction are currently employed in this contact centre (management)?

____

____

____

____

____

- In your opinion, how do you find the role of computers beneficial in your job?

____

____

____

____

____

- How many different software packages do you work with on a daily basis?

____

____

____

____

____

- Describe the purpose of these software packages?

____

____

____

____

____

- Why were these software packages chosen (management)?

____

____

____

____

____

4. Types of Information:

- What types of information is required by software for the management of this contact centre?

____

____

____

____

____

- Discuss the various types of feedback that one can receive from these software?

____

____

____

____

____

- How is information captured and by whom?

____

____

____

____

____

5. Problems Experienced:

- How difficult was it to learn how to use these software packages?

- How long did it take you to learn how to use the software available?

- In your opinion what benefits and disadvantages that these software packages provide?

- What happens when there is a problem with the software; who resolves it and how long does it take to resolve?

- What happens when you make a when using the software?

- How can these mistakes be corrected?

- At what stage is are software related problem escalated and to whom are they escalated to?

- As the manager of this contact centre, what problems are you are faced with (manager)?

- Do you feel that this contact centre is under or overstaffed? And why?

6. Recommended Solutions:

- What additional functionality would you like the software to provide that would make your job more efficient and satisfying?

- What do you think is required that will make this particular contact centre more effective and efficient as a solutions provider to your company's customers'?

7. General Comments:

Appendix B:

Usability Evaluation Test Plan

1. Overview

This test plan is divided into 3 sections:

I. An introduction to the evaluation and the application you will be using
II. A tutorial that will illustrate how to use the application
III. A scenario that needs to be completed in order to complete the evaluation

2. Introduction

The goal of this evaluation is to evaluate how easy it is to log and resolve a customer's query using Intelligent Service Desk (ISD).

During the evaluation the usefulness of the application will be tested. Usefulness will be determined by the easy in which:

- A customer is identified
- His/her query is categorised and classified
- How efficiently a possible solution could be offered in order to resolve the current query

ISD consists of 2 screens. The first screen (Figure B.1) in which the customer's query is categorised and classified, and the second screen (Figure B.2) where a list of possible questions are displayed that could be used to resolve the customer's query.

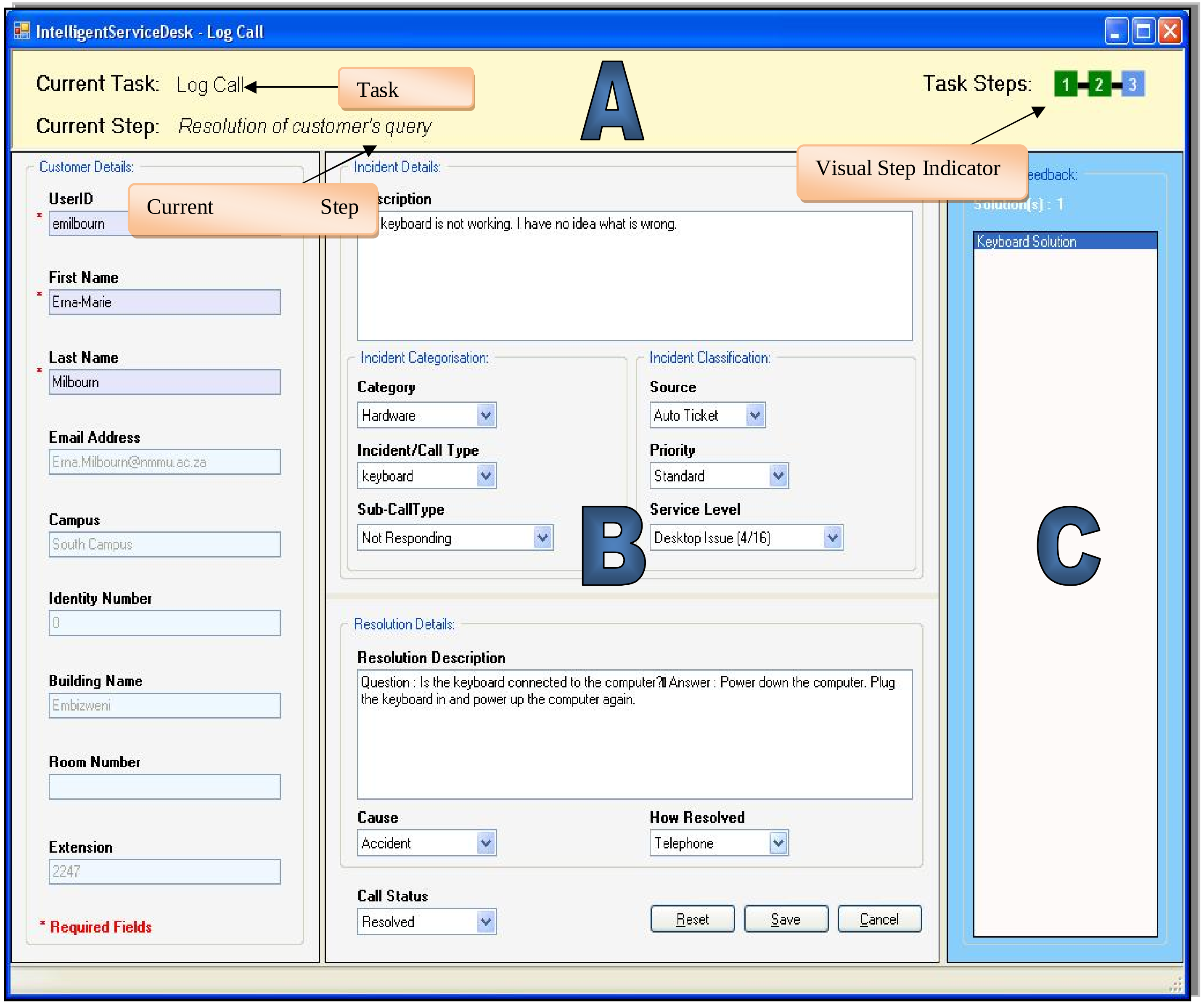


Figure B.1: IUI Prototype – Query Diagnosis Window

Figure B.1 is divided into 3 sections. Section A, B and C.

- Section A contains task status information related to the task at hand
- Section B is the section in which user input is provided in order to identify the customer, log his/her query and provide resolution details.
- Section C is the System Feedback section. This section is responsible for delivering dynamic feedback based on your input (Section B) in order to assist in enhancing the efficiency of the call resolution process.

The colours for the Visual Step Indicator are either:

- **Grey** – Indicates that the step has not yet been attempted
- **Blue** – Indicates the step you are currently busy with
- **Orange** – Indicates an incomplete step
- **Green** – Indicates that the step has been completed

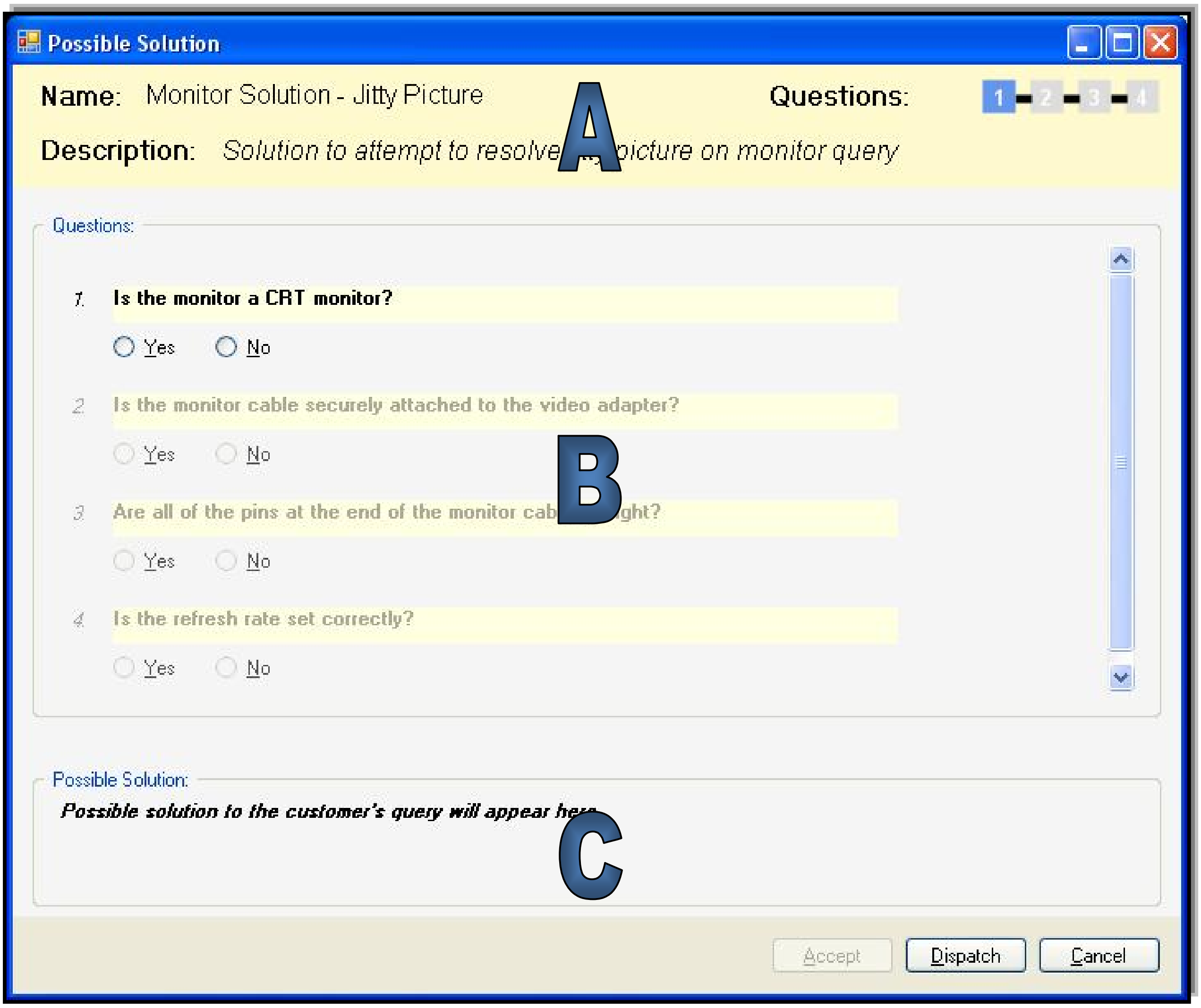


Figure B.2: IUI Prototype – Query Resolution Window

Figure B.2 is also divided into 3 sections. Section A, B and C.

- Section A contains information relating to the possible solution
- Section B contains a list of questions that could be asked to the customer
- Based on the customer's response (Yes or No) the possible solution will be displayed in Section C or the next question will be enabled

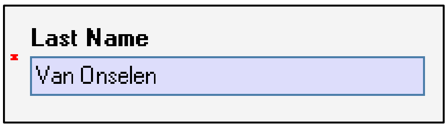
3. Tutorial

In order to successfully log a call there are 3 tasks that need to be completed:

d) Identify the customer
e) Categorise and classify the customer's query
f) Offer a possible solution and capture these details

Task A – Identify the Customer

1.1 Enter the customer's last name (*Van Onselen*) in the **Last Name** field



Customer Details Screen – Last Name field

1.2 Click once on the appropriate option in the System Feedback section in order to provide *More Details* about the selected customer.

1.3 If the selected customer is the correct one then double click on that customer.

1.4 Using the Visual Step Indicator, what is current status of the first step?

Answer: ____

Task B – Diagnose the Customer's Query

2.1 Click on the **Description** box in the *Incident Details* section

2.2 What is the current status of the first step?

Answer: ____

2.3 Enter the following description experienced by Ms Van Onselen in the **Description** box.

"I am currently experiencing some problems with my monitor. The picture is fuzzy."

2.4 Click once on the available option in the System Feedback Section to reveal *More Details*

2.5 What is the category of the option selected?

Answer: ____

2.6 Double click this option in order to select it as the appropriate classification.

2.7 Select the option *Fuzzy* from the Sub-Call Type drop down list.

2.8 Under the Incident Classification Section contained with in the Incident Details Section. Select *Phone* as the appropriate source.

2.9 The priority of the problem is *Standard* and the service level is that of a *Desktop Issue (4/16)*.

2.10 Using the Visual Step Indicator, what is current status of the second step?

Answer: ____

Task C – Offering a Possible Solution

Click on the Resolution Description box in the Resolution Details section

3.1 What is the current status of the second step?

Answer: ____

3.2 How many available possible resolutions are there in the System Feedback Section

Answer: ____

3.3 Click once on each of the available solutions for *More Details*.

3.4 Double click *Fuzzy Solution* in the System Feedback Section.

3.5 In the Possible Solutions form, how many questions are currently available for the current solution?

Answer: ____

3.6 Ms Van Onselen responds:

- Yes to the first question
- Yes to the second question
- No to the third question

3.7 What was the possible solution offered for the customer's query in the Possible Solutions section?

Answer: ____

3.8 Click the Accept button

3.9 Are the details of the possible solution completed in the Resolution Details section?

Answer:

	YES		NO

3.10 Set the Cause of the problem to *Cable*.

3.11 Set How Resolved to *Telephone*. This is because the query was resolved over the telephone.

3.12 Set the Call Status to Resolved.

3.13 Click the Save button to successfully log and save the call. Click the save button.

3.14 Click Yes on the Save Call Details dialog for the save.

4. Evaluation

A test scenario is described below. This scenario can be solved in a similar manner to that of the tutorial. Note that none of the questions need to be answered when solving this scenario. Try to successfully log and diagnose the customer's query in this scenario without making any reference to the tutorial.

Scenario:

Dr. Dieter Vogts is an employee at the Department of Computer Science and Information Systems. His email address is dieter.vogts@nmmu.ac.za and his office extension is 2089. Currently he is experiencing some problems with the Internet. The pages continuously load as *Page Not Found*. He calls the help desk to receive some assistance that could possibly resolve the problem.

> **Dieter**: *"I am currently experiencing some problems with the internet. The pages are not loading and the Page Not Found error is being displayed."*

The *Source* of the call was from a *Phone* and the *Priority* of the call was identified as *Standard*. The *Service Level* was recognised as a *Standard Desktop Issue (4/16).* According to the customer, the network card was connected to the computer and there was activity on the card (Green LED was blinking). The problem was a result of incorrect Internet settings and the possible solution was to go to the intranet and download the appropriate settings. The *Cause* of the problem was *Settings*, it was *Resolved* over the *Telephone* and the *Status* of the problem is that it was *Resolved.*

Solve the above scenario by attempting the following tasks:

Tasks:

- *d) Identify customer details*
- *e) Enter incident details to categorise and classify the query*
- *f) Enter resolution details and save the query*

Appendix C:

Post-Test Questionnaire

Intelligent Service Desk Questionnaire

Please answer all the questions by ticking the category that most accurately reflects your view.

<u>We take your comments very seriously, so please fill in the form seriously.</u>

A. Biographical Details							
Age:							
	20-34		35-50		> 50		
Gender:							
	Male		Female				
Department							
Occupation							
Computer Experience (years):							
	1-2		3-4		5-9		>10

Please circle the numbers which most appropriately reflect your impressions about using ISD. There is space at the end of the questionnaire for your written comments.

B. Overall User Reactions						
Overall reactions to the system:						
1		Terrible				Wonderful
		1	2	3	4	5
2		Frustrating				Satisfying
		1	2	3	4	5
3		Dull				Stimulating
		1	2	3	4	5
4		Difficult				Easy
		1	2	3	4	5

5		Inadequate Functionality				Adequate Functionality
		1	2	3	4	5
6		Rigid				Flexible
		1	2	3	4	5
C. Interface Design						
1	Information on the screen	Hard to Read				Easy to Read
		1	2	3	4	5
2	Screen Design	Frustrating				Satisfying
		1	2	3	4	5
3	The layouts of the screens	Confusing				Clear
		1	2	3	4	5
4	Switching between screens	Insufficient				Sufficient
		1	2	3	4	5
5	The amount of customer information displayed	Insufficient				Sufficient
		1	2	3	4	5
6	The amount of incident information displayed	Insufficient				Sufficient
		1	2	3	4	5
7	The amount of problem resolution information displayed	Insufficient				Sufficient
		1	2	3	4	5
D. Terminology and System Information						
1	Use of terms throughout system	Inconsistent				Consistent
		1	2	3	4	5
2	The relation of the terminology to the tasks	Unrelated				Well Related
		1	2	3	4	5
3	The obviousness of feedback (Error messages, confirmation, warnings) obvious?	Not at All				Very much
		1	2	3	4	5
4	Phrasing of error messages	Unpleasant				Pleasant
		1	2	3	4	5
E. System Feedback						
1	Reliability of System Feedback	Unreliable				Reliable
		1	2	3	4	5
2	Amount of System Feedback	Insufficient				Sufficient
		1	2	3	4	5
3	Ability to view More Details	Difficult				Easy
		1	2	3	4	5

4	Assistance in searching for customers	Not useful				Useful
		1	2	3	4	5
5	Assistance in categorising a problem	Not useful				Useful
		1	2	3	4	5
6	Assistance in finding a solution	Not useful				Useful
		1	2	3	4	5
F. Task Support						
1	Task based information	Not useful				Useful
		1	2	3	4	5
2	Assistance in task support	Not useful				Useful
		1	2	3	4	5
3	Use of Visual Step Indicator	Unhelpful				Helpful
		1	2	3	4	5
4	Indication of progress	Ambiguous				Clear
		1	2	3	4	5
5	Ability to facilitate call logging and resolution	Not at all				Very Much
		1	2	3	4	5
G. General Comments						

Appendix D:

Biographical Profile of Test Participants

Biographical Data												
Age (Years)			**Gender**		**Occupation**		**Department**		**Computing Experience (Years)**			
20-34	*35-50*	*>50*	*Male*	*Female*	*Help Desk Staff*	*IT Support Staff*	*ICT Services*	*CS & IS*	*1-2*	*3-4*	*5-9*	*>10*
8	2	0	6	4	4	6	8	2	2	2	3	3

Table D.1: Biographical Data of Test Participants (n=10)

Appendix E:
User Satisfaction Results

User Satisfaction Results					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Overall User Reaction*	5	5	4.43	0.77
2.	*Interface Design*	5	5	4.67	0.70
3.	*Terminology and System Information*	5	5	4.53	0.85
4.	*System Feedback*	5	5	4.57	0.65
5.	*Task Support*	5	5	4.46	0.86

Table E.1:Overall User Satisfaction - Summary Analysis (n=10)

Overall User Reaction					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Wonderful*	4	4	4.40	0.50
2.	*Satisfying*	5	5	4.50	0.69
3.	*Stimulating*	4	4	4.10	0.70
4.	*Easy*	5	5	4.30	1.21
5.	*Adequate Functionality*	5	5	4.70	0.47
6.	*Flexible*	5	5	4.60	0.67

Table E.2: Overall User Reaction - Detailed Analysis (n=10)

Interface Design					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Information on the screen*	5	5	4.70	0.65
2.	*Screen Design*	5	5	4.70	0.65
3.	*Screen Layout*	5	5	4.80	0.40
4.	*Switching between screens*	5	5	4.60	0.67
5.	*Amount of customer information displayed*	5	5	4.90	0.30
6.	*Amount of incident information displayed*	5	5	4.70	0.47
7.	*Amount of problem resolution information displayed*	5	5	4.30	1.21

Table E.3: Interface Design - Detailed Analysis (n=10)

Terminology & System Information					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Use of terms throughout the system*	5	5	4.80	0.40
2.	*Relation of terminology to the tasks*	5	5	4.70	0.47
3.	*The obviousness of feedback*	5	5	4.20	1.17
4.	*Phrasing of error messages*	5	5	4.40	0.93

Table E.4: Terminology & System Information - Detailed Analysis (n=10)

System Feedback					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Reliability of system feedback*	4	4	4.10	0.94
2.	*Amount of system feedback*	4	4	4.40	0.50
3.	*Ability to view more details*	5	5	4.60	0.67
4.	*Assistance in searching for customers*	5	5	4.80	0.40
5.	*Assistance in categorising a problem*	5	5	4.80	0.40
6.	*Assistance in finding a solution*	5	5	4.70	0.47

Table E.5: System Feedback - Detailed Analysis (n=10)

Task Support					
		Median	*Mode*	*Mean*	*Std Dev*
1.	*Task based information*	5	5	4.60	0.67
2.	*Assistance in task support*	5	5	4.50	0.69
3.	*Use of Visual Step Indicator*	5	5	4.20	1.27
4.	*Indication of progress*	5	5	4.40	0.82
5.	*Ability to facilitate call logging and resolution*	5	5	4.60	0.67

Table E.6: Task Support - Detailed Analysis (n=10)

Appendix F:

Task Completion Rates

ICT Service Desk Staff								
	Tutorial			**Evaluation**				
Participants	*Task 1*	*Task 2*	*Task 3*	*Task 1*	*Task 2*	*Task 3*	***Mean***	***Median***
1.	100%	100%	100%	100%	100%	100%	100%	100%
2.	100%	100%	100%	100%	100%	100%	100%	100%
3.	100%	100%	100%	100%	100%	100%	100%	100%
4.	100%	100%	100%	100%	100%	100%	100%	100%
5.	100%	100%	100%	100%	100%	100%	100%	100%
6.	100%	100%	100%	100%	100%	100%	100%	100%
7.	100%	100%	100%	100%	100%	100%	100%	100%
8.	100%	100%	100%	100%	100%	100%	100%	100%
9.	100%	100%	100%	100%	100%	100%	100%	100%
10.	100%	100%	100%	100%	100%	100%	100%	100%
Mean	100%	100%	100%	100%	100%	100%		
Median	100%	100%	100%	100%	100%	100%		

Table F.1: Tutorial and Evaluation Task Completion Rates (n=10)

Appendix G:

Eye-Tracking Data

ICT Service Desk Staff						
Participants	**Fixed Duration**			**Observation Count**		
	Task Status Section	*System Feedback Section*	*User Input Section*	*Task Status Section*	*System Feedback Section*	*User Input Section*
1.	1.48	22.70	202.06	3	16	22
2.	15.17	51.86	99.98	21	24	37
3.	4.01	19.82	68.82	10	15	22
4.	18.09	12.21	93.07	19	12	25
5.	8.76	28.11	71.57	9	15	22
6.	3.56	21.09	117.69	10	27	34
7.	9.38	21.90	116.09	17	16	27
8.	0.38	14.46	40.62	2	11	14
9.	18.67	33.73	90.39	18	14	26
10.	4.78	31.38	75.54	6	19	23
Mean	8.43	25.73	97.58	11.50	16.90	25.20
Median	6.77	22.30	91.73	10.00	15.50	24.00
StdDev	6.78	11.42	43.43	6.85	5.09	6.51

Table G.1: Tutorial Fixed Duration and Observation Count Data (n=10)

ICT Service Desk Staff						
Participants	**Fixed Duration**			**Observation Count**		
	Task Status Section	*System Feedback Section*	*User Input Section*	*Task Status Section*	*System Feedback Section*	*User Input Section*
1.	0.13	7.16	67.77	1	6	13
2.	0.80	13.74	82.24	2	5	7
3.	1.02	4.26	16.74	3	3	8
4.	0.00	4.66	41.60	0	6	9
5.	0.17	1.92	33.31	1	2	3
6.	0.32	16.17	89.90	2	17	16
7.	2.88	15.14	128.32	7	20	24
8.	0.18	3.06	53.81	1	5	7
9.	0.00	21.64	70.42	0	10	15
10.	0.00	8.14	45.94	0	6	11
Mean	0.55	9.59	63.01	1.70	8.00	11.30
Median	0.17	7.65	60.79	1.00	6.00	10.00
StdDev	0.89	6.65	32.13	2.11	5.96	5.98

Table G.2: Evaluation Fixed Duration and Observation Count Data (n=10)