A Multi-Touch ER Diagram Editor to Capture Students' Design Rationale

R.G.Stone, F.Batmaz, T.M. Rickards

***Abstract*—The increased presence of diagram-type student work in higher education has recently attracted researchers to look into the automation of diagram marking. Research into the semi-automatic diagram assessment at Loughborough University has identified the requirements of a diagram editor in order to capture the students' design rationale. To fulfill these requirements, several experimental diagram editors have been developed. This paper introduces an ER diagram editor which uses multi touch technology. The initial experiments and findings for the editor are described in the paper.**

***Index Terms*—e-Assessment, Graph Diagrams, Human Computer Interface, Self-Explanation Systems, multi-touch input.**

I. Introduction

Online assessment is becoming more popular. More educational organizations are considering online assessment rather than traditional paper-and-pencil exams since it reduces the overall marking time as compared to dealing with paper-based assessment and improves the marking consistency [1].

Online assessment implies two main concepts. Firstly, it means that students give their responses online, secondly, that these responses are marked by an automated system [2]. However automated marking systems have quite limited proven forms consisting of multiple-choice questions, matching questions or simple 'fill in the blank' questions. These forms of online testing are not sophisticated enough to examine students' understanding of complex content and thinking patterns [2]. Diagram-type student works (e.g. ER diagrams, UML) are required to assess students' knowledge in a more comprehensive way like essay-type questions. Although research into automatic essay and diagram marking has been carried out (e.g. DEAP Project [3]), full automation of the marking has not been achieved yet. We believe that human markers are required who manually assess the diagrams in a supportive online environment in this intermediate stage towards full automatic systems.

This research envisages a system with the main characteristics of online diagram drawing by students and online marking of these diagrams by a human marker using a specialised tool. A first version of the system has already been implemented and was used in the tutorial sessions in 2009. Based on this core system we have developed an interactive environment that allows students access to comprehensive feedback on class level [4]. Among hundreds of known diagrammatic notations in computer science, the research, in its initial stage, focuses on conceptual database diagrams (Entity Relationship diagrams).

The aim of the marking tool is to reduce the number of the components in the diagrams marked by the examiner which we call this approach semi-automatic marking. That requires finding the identical components in different student diagrams. The Assess by Computer (ABC) [5] project, which has got the same aim as this project, defines identical components by using those component's attributes (e.g. label, type, adjacent boxes). In our research, identical components are defined by the references to the text describing the scenario (design tracing). A similar approach is used for intelligent tutoring system in KERMIT [6] and ERM-VLE [7] projects. Design tracing and component matching are discussed in the semi-automatic approach section.

The KERMIT and ERM-VLE projects have developed their own online diagram editors to capture student diagrams. This research also requires its own diagram editor. The editor is a new version of the previous online diagram editor [4] which has been tested on large number of the students. The new version deals with students' interaction problems with the previous user interface. The tool and the new student-computer interaction are discussed in the diagramming tool section. A prototype of the editor has been tested on a small number of students. Results from this may be found in the experiment sections and further work is described in the final section.

II. Semi-Automatic Approach

The semi-automatic marking aims to reduce the number of diagrams marked by the examiner. The system groups identical components of the students' diagrams and then asks the assessor to approve the correctness of a representative from each of the different groups. Therefore the examiner would be involved in the marking process only for the number of diagram groups rather than the total number of student diagram components.

The correctness of the grouping depends on the criteria used to match the diagram components. This research uses the student design traces as the criteria for each component in their diagram. Design traces are links from scenario texts to diagram components. Figure 1 shows the design trace of

Manuscript entitled "A Multi-Touch ER Diagram Editor to Capture Students' Design Rationale" submitted to International Conference on Education and Information Technology 2010, July 13, 2010.

R. G. Stone is with Computer Science Department, Loughborough University, UK (e-mail: R.G.Stone@lboro.ac.uk).

F. Batmaz is with Computer Science Department, Loughborough University, UK (e-mail: F.Batmaz@lboro.ac.uk).

T. M. Rickards was with Computer Science Department, Loughborough University, UK (e-mail: T.Rickards-06@student.lboro.ac.uk).

"Staff" entity. 'Lecturer' and 'HoD' noun phrases appear in the scenario text and are initially mapped to the corresponding entities which are, at a later stage, merged into a 'Staff' entity. Thus there is a trace which links 'Lecturer' in the text to the entity 'Staff' in the diagram. "Lecture" and "HOD" entities in the figure have a direct reference to scenarios. This research calls them direct referenced (DR) component whereas "Staff" entity is called indirect referenced (IR) component.

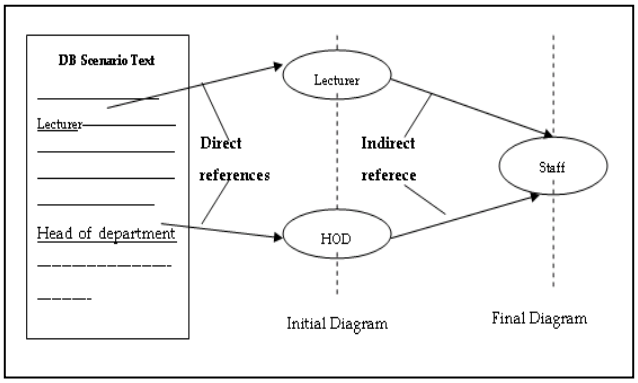

Figure 1 - Design Trace of "Staff" Entity

Components of student diagrams for the same scenario are grouped based on these traces in the semi-automatic approach. One component from each component group is represented with its design trace to the examiner. The examiner can see how the student created the components. It may enable the examiner to understand more about their design rationale and comment on the student's work.

Production of the design traces can be done in various ways. Pinhairo [8] looks into trace production in the requirement traceability research field. He describes two kinds of trace productions: off-line and on-line. The off-line production performs capturing process after the student activity has finished. For example, students submit their works with their design activities and then the activities are analysed automatically and interpreted by the human marker to produce design traces. Thus the off-line method requires an additional interpretation task on top of marking the diagram components. Moreover, the examiner's interpretation may not be what the student intended.

In online production, the traces are captured as a result of performing design activities. For example students do the design in the special environment which enables the students create design traces as a part of design process. They explicitly mention their intention during the design. This method is called self–explanation in the literature [9].

Psychological studies [10] show that self-explanation (SE) is a very effective learning strategy resulting in deep knowledge. SE systems support students while they study solved examples or are asking for an explanation while solving problem. The main problem of self-explanation whilst solving the problem is the high cognitive load [9]. The proposed diagram editor is designed to reduce the cognitive load of the self-explanation. Section III looks at components of this diagram editor and examines how cognitive load may be reduced.

The central part of the semi-automatic approach is the process of grouping diagram components. Too many component groups to be marked could decrease the efficiency of the assessment process severely. The number of groups depends on the number of different design traces for a particular component. The reason for different traces is diversity in either the students' reasoning or their actions for the same reasoning. The reasoning diversity is restricted by the given scenario text. A method for scenario text writing was developed to control the reasoning diversity. The details of the method can be found in [11]. The action diversity can be controlled by the user interface of the diagram editor. User gestures for the editor are defined in Section III for this purpose.

The marking part of the semi-automatic approach is described in detail in the paper [12]. The developed marking tools deal with automatic marking of some of the component groups. It uses the design traces of previously marked components and domain independent rules.

III. DIAGRAMMING TOOL

The prototype diagram editor is based on multi-touch technology. The technology provides an ability to touch more than one point at once on a screen. Many useful applications using this technology have been developed for the Microsoft Surface and the Apple iPhone platforms since 2007.

The prototype editor provides an environment for students to draw their diagrams and it captures their diagramming steps. Students' multi-touch interaction with the editor enables the steps to be interpreted to understand students' design rationale. This interaction also helps students to be able to apply the design methodology they learn in the lecture.

The editor has a user interface without any toolbox on it. Figure 2 shows the diagram editor with a simple ER diagram on the canvas. It has two panes; Scenario and Diagram. The scenario pane displays the scenario text paragraph by paragraph so that the student considers the information in that section only. This method is called scaffolding in the self-explanation literature [9].

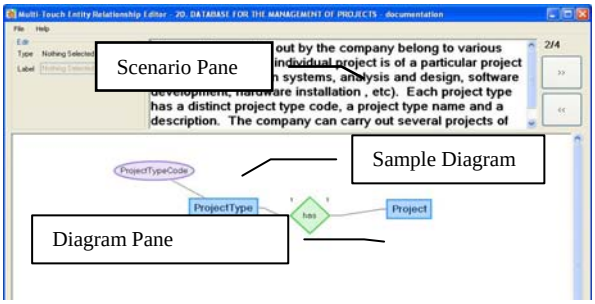

Figure 2 - The diagram editor

The interface works without a toolbox by assigning meaning to the gesture of dragging items to identifiable targets on the screen. The draggable items comprise every noun phrase (NP), scenario sentence (SS) in the scenario and every entity (E), attribute (A) and relationship (R) in the diagram. The targets are the components of the diagram (E, A, R) plus the Scenario Pane (SP) and (any unoccupied part of) the Diagram Pane (DP). Thus NP->DP is interpreted as a request to create a new entity (E) in the diagram while E->SP

is interpreted as the erasing of the entity. Obviously the topology of the diagram can be manipulated by dragging the items within the diagram pane. The selection of a scenario sentence for dragging is a multi-touch interaction whereas the other selections are mono-touch.

Not all the editing can be accomplished by the dragging gesture. The gesture of anchoring (described below) is used to identify a component for editing, meaning updating, not creation or deletion. Anchoring is a multi-touch interaction. The rest of this section describes the gestures of the editor in more detail.

Entity creation: Entities are the only element on the diagram which could be considered independent. They require no other elements in order to exist, whereas relationships and attributes cannot be created without entities to connect to. As such, a simple drag and drop gesture will be employed to create entities. A user selects and drags a (copy of the) noun phrase from the scenario and drops it directly where they want an entity to appear on the diagram. The process of dropping the noun phrase onto a blank canvas destroys the noun phrase and creates an entity (based on the noun phrase) in its place.

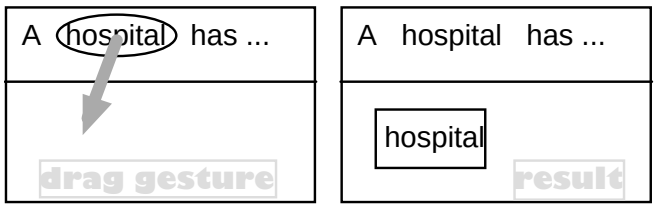

Figure 3 - Dragging a noun phrase from the scenario text to the diagram to create a new entity

Attribute creation: Attributes have one connection on the diagram, to their parent entity. To create an attribute a user needs to select and drag a noun phrase and drop it on top of an existing entity. This randomly places the new attribute around the edge of the entity.

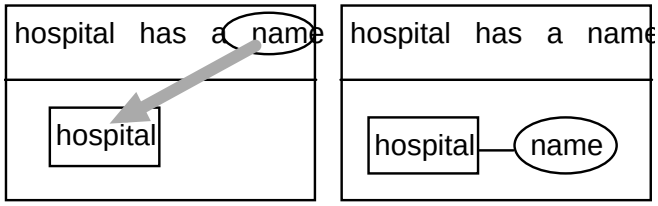

Figure 4 - Dragging a noun phrase from the scenario text to an entity on the diagram to add a new attribute to the entity

Relationship creation: A binary relationship inherently relates two entities to each other and as such requires two existing entities on the diagram in order to be created. The creation of a relationship will involve two points on the screen with no restriction as to which two fingers they should come from. A user is required to find a sentence from a scenario which relates two entities to one another and choose two points. By touching these two points, relationship diamonds will appear under them to signify they are involved in the gesture. These two diamonds can be dragged around just like noun phrases and should be dragged to the canvas and dropped on top of the two entities to relate. This will finalise the creation of the relationship, with the connecting lines drawn by the canvas.

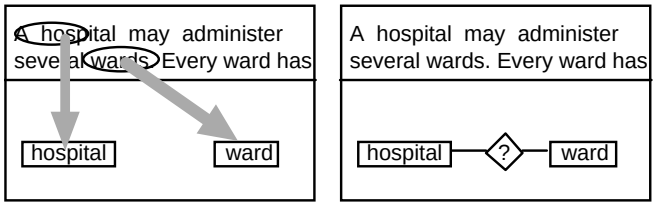

Figure 5 - Dragging a sentence (defined by two touch points) from the scenario text to two entities on the diagram to add a new relationship between the entities

Component Deletion: To delete components that exist on the diagram, the most intuitive solution which also coincidentally saved valuable diagramming space, is to drag the components back to the scenario pane. Attributes and relationships can simply be picked up and dragged away from the canvas to be deleted with no cascading effect. Deleting an entity, because of the need to enforce integrity, will also delete any connected relationships or attributes. While an entity is being held over the scenario to be deleted, the canvas will highlight any connecting elements that will be deleted via the cascading effect. If at any time after the element has been picked up and held over the scenario for deletion and the user would like to return it, it can be dropped anywhere on the canvas and will snap back to its original position.

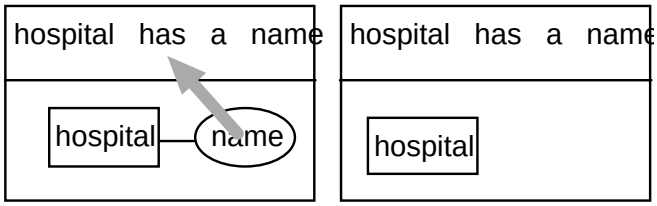

Figure 6 - Dragging a diagram component back to the scenario in order to erase it from the diagram

Anchoring: While deciding the gestures for various editing actions on the diagram, it became apparent that there needed to be a way to distinguish between movement of an element and a gesture on an element. This is where the concept of anchoring comes in, in order to signify that a certain section of the diagram has paused for editing. A user will be able to touch and hold a component in one location, for just under a second, in order to anchor it. This will allow for a whole new set of gestures to be created while the touch point is held on the component. Even outside of creating an entity relationship diagramming application the ability to anchor will appear in any number of situations where both movement and editing are required.

Editing a relationship's label: A relationship is based upon a specific sentence so that a user can use appropriate words in that sentence to represent the connection. Once a relationship has been created with its default label, a user will be able to edit it. When a user anchors a relationship, the current words in the label will appear above it, each in their own movable rectangle. The words can be deleted by dragging them to the scenario or rearranged by moving them in front of each other. Every time the user lets go of a word,

ensuring the anchor is still applied, the words will snap back to a particular arrangement. New words can be added by dragging non-noun phrases from the sentence the relationship was created from, which is automatically highlighted when the relationship is selected. Releasing the anchor commits the words to the label in their current order.

Editing a relationship's cardinality: Each side of a relationship has a cardinality attached to it which is required to be variable. A user is able to cycle through the possible values (1, 0/1, n, 0/n) by anchoring the relationship in the same way as editing the label and then tapping the corresponding entity once for each step through the cycle. When the anchor is removed, the elements will become movable again and the cardinality will be committed.

Editing an attribute's primary: Attributes are grouped by their parent entity and are flagged as primary of that entity. As such the entity will be the anchor point for changing primary key status. Once an entity has been anchored, any of it's connected attributes can be tapped to scroll through the combinations. When released the values are committed for that attribute.

The way editor has been designed allows for concurrent interaction with components on the screen. The scenario control has been designed so multiple noun phrases can be selected at once, which inherently allows multiple entities and attributes to be created simultaneously. Multiple anchors can also be placed, allowing for the editing of various properties at the same time. For the majority of gestures, a group of people standing in front of the screen can interactively design a diagram together. This allows for decisions to be made collaboratively allowing relationships to be visualised immediately as opposed to after a meeting or discussion.

The gestures enable students to be able implicitly explain their design steps to themselves and the examiner. The Students can see how the scenario and their diagram are related in this way. Although the gestures do not need any additional text input currently, students may be asked to write extra text for an explicit explanation in the future.

The gestures, which the editor provides, are not the complete list of gestures needed to capture all students design rationale. For example, the editor does not currently support merge and split gestures which result in DR-components. Additional gestures can be defined for better diagram drawing and editing. Research in the future will define new and alternative gestures for better interaction with the multi-touch screen.

IV. Experiment and Results

The purpose of the experiment was to see how normal end users, with no experience of the software, manage when creating diagram elements and using the multi-touch interaction it provides. The multi-touch device used was a 42inch G2 screen from PQ-Labs overlaying an LCD monitor. Participants were volunteers chosen at random from a class of students studying a Database Module at Loughborough University, common among first year students sitting Computer Science and IT related degrees. They were split into 5 groups who first watched a demonstration, before being asked to create a small diagram with what they have seen. After around 5 minutes of observed test time they were asked to fill in the feedback form.

Groups 1 and 2 were students taken from the databases module studying for an Information Technology Management for Business (ITMB) degree, whereas groups 3, 4 and 5 were studying for a Computer Science (CS) degree. Every group managed to successfully perform all of the gestures with only occasional pointers needed regarding the finer details of their execution.

Two questions on the feedback form were based on a five item Likert scale, giving the user options on how much they would agree with the statements given. A Likert scale can be quantitatively analysed by assigning a number to each of the possible values. Figure 7 shows the average results across all groups for the question which asked “How much you agree with each gesture being well suited to its purpose?”. Figure 7 shows that the students’ perceptions are very positive for the implemented gestures. Both creating entities and attributes saw a 96% agreement with the question, indicating users felt this was the best way to implement such an interaction. Creating relationships and gestures involving anchoring received mid to high 80's which also is a very encouraging number for the first attempt at designing gestures for this kind of application. The lowest scorer was for the deletion of elements on the canvas which received an 82% agreement, possibly suggesting users might like to see other ways of performing the task.

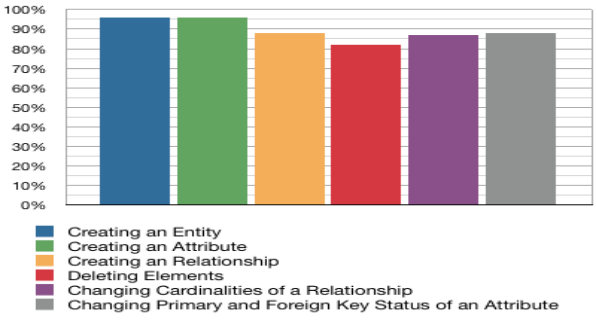

Figure 7 - Each gesture being well suited to its purpose?

Figure 8 shows the average results across all groups for the question which asks “How much you agree with each gesture being smooth and well implemented?”. The overall average for this question is slightly lower but still very promising.

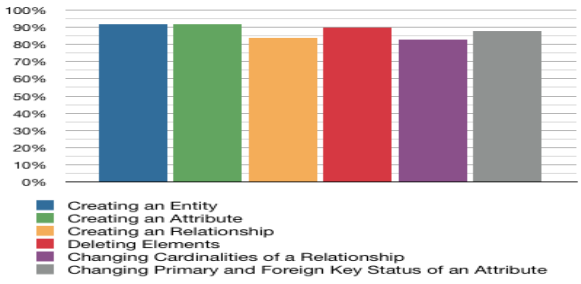

Figure 8 - Each gesture being smooth and well implemented?

The results show a similar trend with creating entities and attributes as the highest scorer but this time creating and changing cardinalities of relationships came in lowest at 84% and 82.8% respectively. This gesture could be modified made to improve this score to match those with the highest values.

Comparison of ITMB to CS Students shows another interesting trend that CS students answered consistently lower than ITMB students. This may be due to the fact CS students have been exposed to other multi-touch devices, possibly those with the more forgiving capacitive technology as opposed to the infra-red used in the G2 overlay.

It is also possible that the values could increase with continued use. The results were taken from users with no prior exposure to the gestures or this type of multi-touch desktop application. Users would be expected to increase productivity over time as they get used to the interactions required and get more experienced with the hardware by using it on a regular basis.

The feedback form also contained questions regarding suggestions for new gestures and possible new applications in which this technology could be implemented. A particularly interesting suggestion was made regarding the swapping of entities. To avoid the cascading deletion of large sections of a diagram just to change an entity's label, noun phrases could be picked up and swapped with an entity instead. It could be implemented using the existing anchoring technique to keep consistency and would still enforce integrity of a diagram. Other ideas such as double tapping to change the relationship's cardinality and selecting entities to relate prior to dragging a relationship down from the scenario could be looked into further. The observations during testing indicated there may be some room for slight improvements on the gestures implemented, specifically for creating relationships where some users struggled on the first attempt.

All groups answered that they would consider using software like this again and the verbal responses reinforced this result.

V. CONCLUSION AND FURTHER WORK

A group of students working on a large multi-touch screen has potential to increase the collaboration among the students. However the size and cost of such screens may currently prevent them from being commonplace in a computer laboratory. Additionally, the examiner can only mark the group's design rational rather than individual work.

The semi-automatic approach is a very powerful method of dealing with the marking of ever-increasing numbers of student assignments. It can provide detailed personalised feedback independent from the number of students. The group use of a large multi-touch screen does not demonstrate this potential of the approach. Therefore, the research has already planned to use personal multi-touch devices such as the iPad from Apple. The defined and tested gestures in the large screen environment can be used in these personal devices without major modification.

Using gestures to capture student design rational can be applied to many different graph-based diagrams in computer science and many other engineering programmes, since the gestures are not specific to one type of graph diagram. The research's findings about the gestures can also be used in requirement engineering field in order to develop the industrial application for online trace production.

Self-explanation systems have been developed for school students to improve their learning in some subject areas. We believe that gestures and multi-touch technology can also be used to improve the student self-explanation skills.

REFERENCES

[1] K. Christie, "Online assessment: Moving beyond 'Gotcha'". Phi Delta Kappan, 83(6), 2002, p426.

[2] E. Heinrich, Y. Wang, "Online marking of essay-type assignments". Proc. ED-MEDIA 2003, Honolulu, Hawaii, USA, 2003, 768-772.

[3] P.Thomas, K.Waugh, and N. Smith, "Experiments in the automatic marking of ER-diagrams", SIGCSE Bull. 37, 3 (Sep. 2005), 158-162.

[4] Batmaz, F., Stone, R.G. and Hinde, C.J., "Personal Feedback with Semi-Automated Assessment Tool for Conceptual Database Model", 10th Annual Conference of HEA-ICS, University of Kent, Canterbury, UK, August 2009, pp.115-119.

[5] C. Tselonis, J. Sargeant, M. McGee Wood, "Diagram matching for HCC assessment", Proc. 9th CAA Conference, Loughborough UK, 2005

[6] P. Suraweera, A. Mitrovic, 2002, "KERMIT: A constraint-based tutor for database modelling". Proc ITS'2002, LCNS 2363, 2002, 377-387.

[7] L.Hall, and A. Gordon, 1998, "A virtual learning environment for entity relationship modelling". Proc 29th SIGCSE, Atlanta, Georgia, US, 1998, 345-349.

[8] F. Pinheiro. "Design of a Hyper-Environment for Tracing Object-Oriented Requirements". Doctoral dissertation, University of Oxford, 1996.

[9] Chi,M.T.H. , M. Bassok, M.Lewis, P. Reinmann and R. Glaser, "Self-Explanations: How students study and use examples in learning to solve problems". Cognitive Science, 13(2) 1989, 145-182.

[10] A, Bunt, C Conati, and K, Muldner, "Scaffolding self-explanation to improve learning In exploratory learning environments", Proc 7th ITS, LNCS 3220, 2004, 656-667.

[11] Batmaz, F. and Hinde, C.J., "A Method for Controlling the Scenario Writing for the Assessment of Conceptual Database Model", Proceedings of the 11th IASTED International Conference on Computers and Advanced Technology in Education, Uskov, V., Acta Press, Cate2008, Crete, Greece, October 2008, pp. 327-332.

[12] Batmaz, F. and Hinde, C.J., "A Web-Based Semi-Automatic Assessment Tool For Conceptual Database Diagram", Proceedings of the sixth IASTED Internation Conference Web-Based Education, Chamonix, France, March 2007, pp. 427-432.