In Proceedings of the 2011 IEEE/ACM International Conference on Automated Software Engineering, (c) IEEE 2011

Generating Essential User Interface Prototypes to Validate Requirements

Massila Kamalrudin
Department of Electrical and Computer Engineering
University of Auckland
Private bag 92019 Auckland 1142, New Zealand
mkam032@aucklanduni.ac.nz

John Grundy
Centre for Computing and Engineering Software Systems
Swinburne University of Technology
PO Box 218, Hawthorn, Victoria 3122, Australia
jgrundy@swin.edu.au

***Abstract*—Requirements need to be validated at an early stage of analysis to address inconsistency and incompleteness issues. Capturing requirements usually involves natural language analysis, which is often imprecise and error prone, or translation into formal models, which are difficult for non-technical stakeholders to understand and use. Users often best understand proposed software systems from the likely user interface they will present. To this end we describe novel automated tool support for capturing requirements as Essential Use Cases and translating these into "Essential User Interface" low-fidelity rapid prototypes. We describe our automated tool supporting requirements capture, lo-fi user interface prototype generation and consistency management, and outline a user evaluation of our tool.**

Keywords-requirements validation; rapid prototyping

I. INTRODUCTION

Requirements capture from clients is often difficult, time consuming and error prone [1, 2]. Late validation, in particular, causes requirements quality to suffer [3]. This has placed a focus on techniques for early client feedback such as use of formal and semi-formal models and heuristic algorithms [4],[5] plus techniques for translating natural language requirements into such models. While beneficial, these approaches are often difficult to use and require much effort [6] [7]. Rapid prototyping can be one of the best ways for early validation of requirements from both a requirements engineer (RE) and a client's view [8]. Using prototypes, clients gain a much clearer understanding of a proposed system via an intuitive representation, or mock-up, of the target system. This helps to very early on identify missing or incorrect requirements [9],[10].

For early stage requirements analysis, low-fidelity or abstract prototypes are useful [11]. However, developing such prototypes requires effort [10] and is poorly supported by toolsets [11]. In previous work we have developed a technique and toolset for checking consistency of requirements based on Essential Use Case (EUC) diagrams [12-14].These EUCs are semi-formal models which we automatically extract from natural language requirements and validate against known EUC patterns. Here, we describe a significant extension of this work providing end to end rapid prototyping support. EUC models are mapped to an abstract Essential User Interface (EUI) prototype model. From there they are mapped to concrete User Interface (UI) views in the form of form-based UIs. This allows the RE and client to walk-through the formalized requirements together and to validate and confirm the consistency of these requirements. We have implemented a set of EUI patterns as an extension to our existing Marama AI EUC extraction and modeling tool and have conducted a formal user evaluation of both the EUI extension and its resulting end to end rapid prototyping and requirements validation support.

II. BACKGROUND AND OUR APPROACH

EUI prototyping is a low fidelity prototyping approach [15]. It provides the general idea behind the UI but not its exact details. It focuses on the requirements and not the design, representing UI requirements without the need for prototyping tools or widgets to draw the UI [16]. EUI prototyping extends from and works in tandem with the semi-formal representation of EUCs, both focusing on users and their usage of the system, rather than system features [2]. It thus helps to avoid clients and REs being misled or confused by chaotic, rapidly evolving and distracting details.

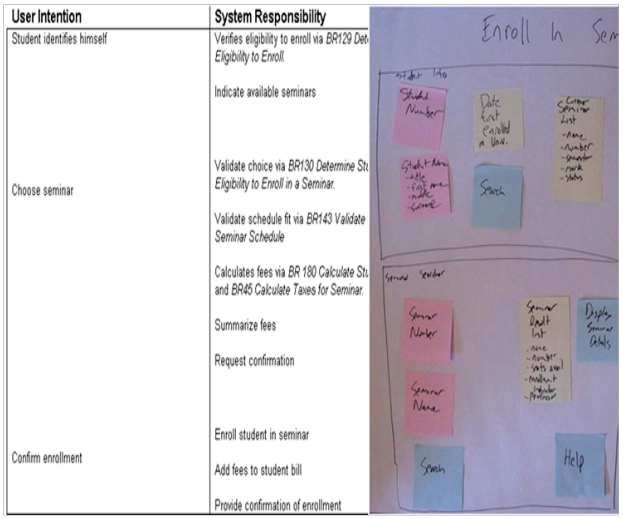

Figure 1 Example of EUI prototype from an EUC model (Ambler [2][16]))

Figure 1 shows an example of an EUI prototype being developed from an EUC. The post-it notes represent abstractions of user interfaces. Here the requirements engineer is capturing the user intention/system responsibility dialogue represented in the EUC as possible UI functionality, at a high level of abstraction. Although EUI prototyping has advantages, it has not been rigorously applied in practice as no tool support is available. Being a whiteboard/paper technique, it does not integrate well with other tools [15]. Previous work has shown that the application of low fidelity techniques in practice proves challenging [10].

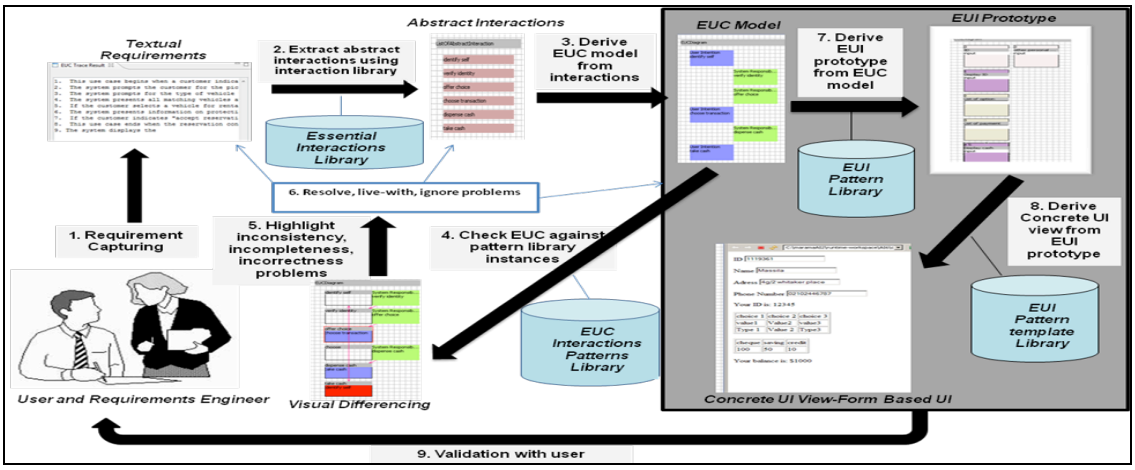

Figure 2 End-to-end EUC and EUI prototyping approach

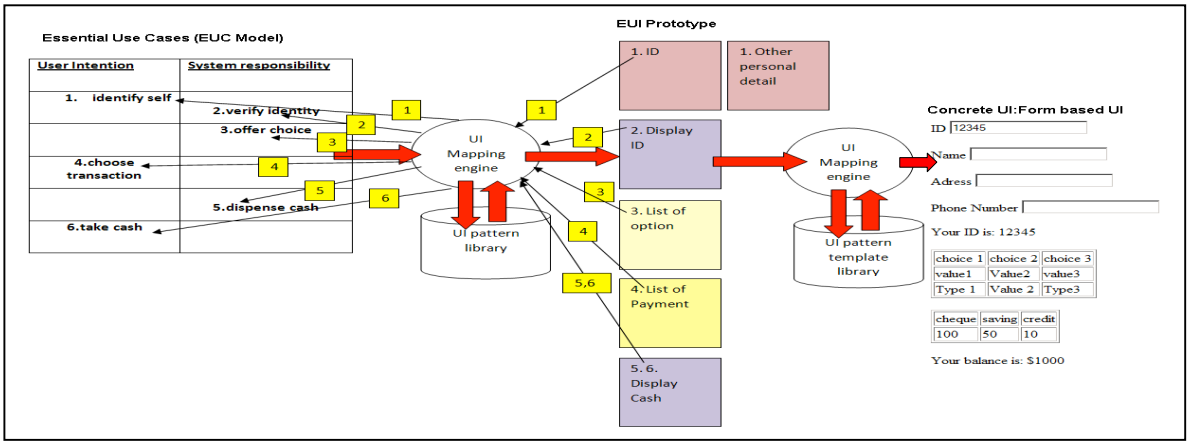

Figure 3. An example of performing mapping of EUC model to EUI prototype using the UI Pattern library with trace-forward/ trace-back

We have developed an approach and supporting tool to enable REs to more effectively capture or confirm requirements with clients, as shown in Figure 2. REs elicit/capture requirements from users (1) as textual natural language. These are translated to a list of abstract interactions (essential requirements) using an essential interaction library (2). These abstract interactions are transformed to an EUC model (3) capturing the sequence and interactions of a requirement. REs can validate requirements consistency between the various models and also against known, valid patterns of both Essential interactions and EUCs (4, 5 and 6) [13]. Our new work, in the grey box, allows the RE to automatically and traceably transform EUC models to EUI prototypes (7). Combined with our earlier toolset, this means traceability is provided throughout the process allowing any of the EUI components to be traced forward/back from/to the EUC model, abstract interaction or textual natural language requirement. The EUI prototype can also be translated to a more concrete HTML web form (8). Simple interaction with the generated HTML form is supported to illustrate how target system information input and output could work. This EUI model and concrete UI is reviewed by the RE with end users to validate and confirm consistency of the original textual requirements (9).

Figure 3 shows the mapping and tracing process between the EUC, EUI prototype and Concrete UI view, using Constantine and Lockwood's "getting cash" scenario. The numbers indicate mapped elements between the models. The EUC model is mapped to/from the EUI prototype using the UI mapping engine. This maps each of the abstract interaction components which have a relevant EUI pattern in the EUI pattern library. For example, the abstract interaction "identify self" will be searched for in the EUI pattern library and its related EUI pattern found. This results in abstract UI elements "ID" and "Other personal detail" being added to the EUI model. More than one abstract interaction may share the same UI pattern. For example, the abstract interactions "dispense cash" and "take cash" share the same UI pattern "Display cash". Here, only one UI pattern "Display Cash" is included in the model. The EUI patterns were developed by us using an adaptation of the brainstorming methodology of Constantine and Lockwood [17]. Our adaptation generalised their approach using simpler and more generic EUI patterns. Our generalized EUI patterns comprise four types of EUI pattern category: List, Display, Input and Action. The main aim of these EUI Patterns is to assist REs to rapidly model a user interface based on the requirements captured and modelled earlier in the EUC model.

III. TOOL SUPPORT

We have extended our prototype tool, MaramaAI [12, 18], to additionally and automatically map EUC models to EUI prototypes and concrete UIs. The EUI prototype is modelled in a Marama editor, MaramaEUI. The concrete UI is presented in the form of an HTML page, both realised in the Eclipse IDE. Several screen dumps of the tool in use are shown in Figure 4. From a set of natural language requirements (1) semi-formal EUC models are extracted (2) and then mapped to a low-fidelity Essential User interface model in a MaramaEUI editor (3). Each EUI prototype component is associated with an EUC model abstract interaction component and, via that, the original natural language textual requirement it was derived from. Any EUI component can be selected and its associated EUC component and related textual natural requirement can be shown using a "trace back" menu item which highlights the relevant components. For example in Figure 4 (section A), the "List of Option" EUI Component (3) is traced back to a "system responsibility: offer choice" EUC component (2) which in turn is traced back to the textual requirement (1), both of which have been highlighted. One EUI component might associate with more than one abstract interaction in the EUC model. Figure 4 (section B) shows that the EUI component " Display cash" (4), traces back to two abstract interaction components of the EUC model "dispense cash" and "take cash" (5) and the associated textual requirement (6) "dispenses the requested amount" and "receives cash".

Figure 5 shows the view for both the Marama EUI (A) and concrete HTML form-based UI view (B). Both views allow the RE and client to walk-through the requirement and its UI in order to validate the consistency of the requirement. The Marama EUI component is editable allowing the RE and client to add input detail that they think is required or delete any UI pattern that they think is not necessary.

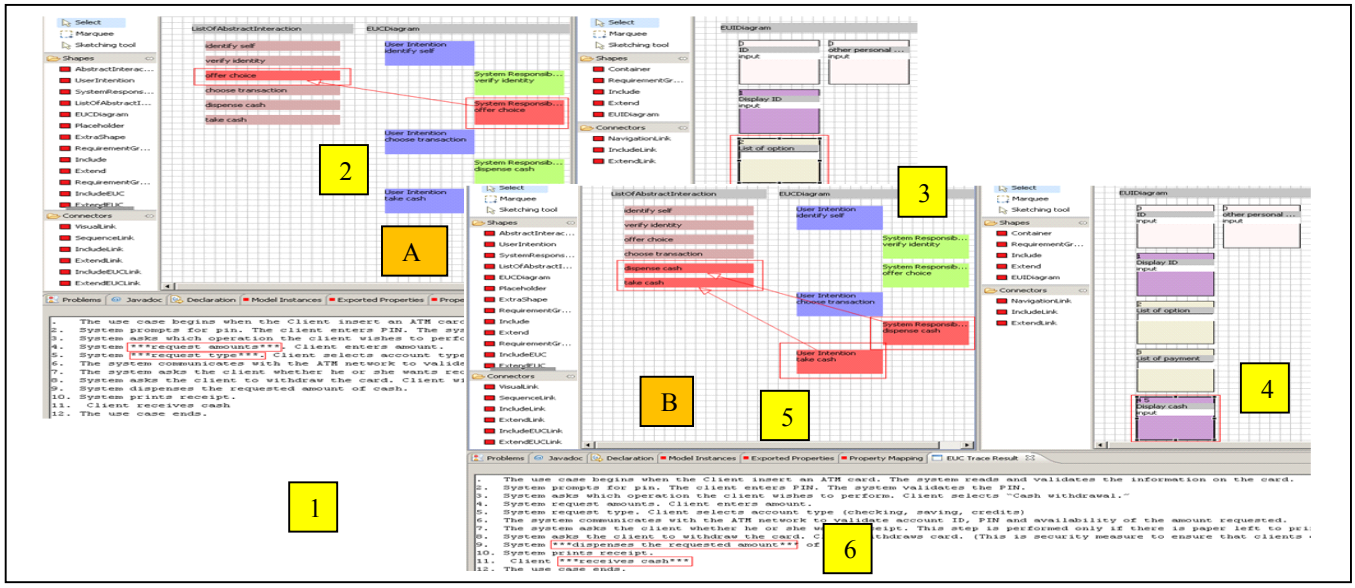

Figure 4.Trace forward and Trace-back from EUC model to EUI prototype.

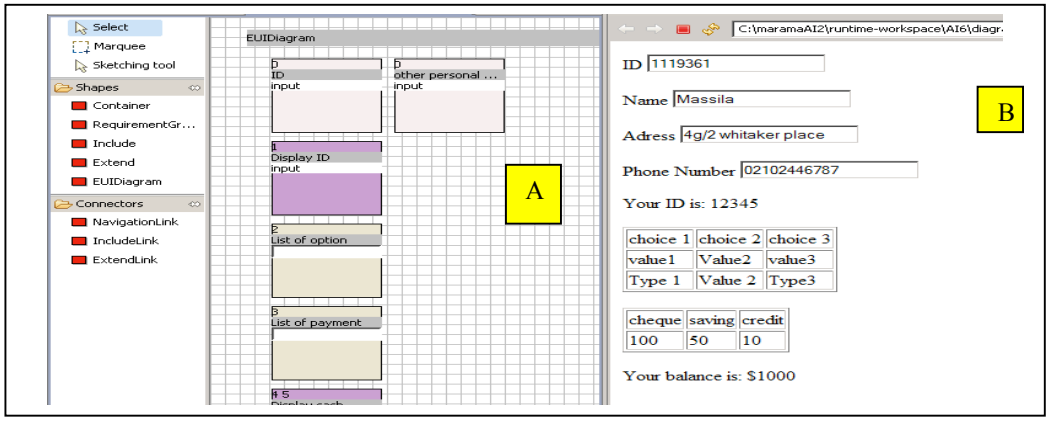

Figure 5. Marama EUI and concrete UI view in a form of form- based UI

IV. RELATED WORK

Ogata and Matsuura proposed a method for automatic generation of user interface prototypes for web-based business applications based on requirement specifications defined in UML [20]. Their work guarantees consistency of the data and flow between the RA model and prototype, and decreases the time taken to conduct requirements analysis. In contrast, our work supports the consistency of requirements

using end-to-end prototyping from natural language requirement to semi formal models in the form of Essential Use Cases (EUCs), abstract prototypes (EUI prototypes) and form-based UIs for various domains, not just business applications, complementing their approach. Work on non tool-based techniques includes Vijayan and Raju, who propose a paper prototyping approach for eliciting requirements [21]. Examining the captured paper prototype to identify omissions, ambiguities and other requirement quality problems validates requirements gathered. Our work is complementary to theirs. Combining the two approaches would provide an interesting approach to elicitation and validation by allowing comparison of the two different types of lo-fidelity prototype for consistency. Molina et al. have developed a model and graphical notation for specification of abstract user interfaces based on a conceptual pattern [22]. Their Just UI approach identifies patterns for UIs and abstracts them to work with problem domains specifically for presentation and navigation issues. It extends OO methods to capture UI requirements and presents a set of patterns that can be used as building blocks to create UI specifications for information systems manually. We do not focus on UI issues specifically, rather using UI prototypes as a means of understanding requirements.

V. SUMMARY

We have described an approach supporting the confirmation and verification of requirements from both and RE and client perspective using end-to-end rapid prototyping. We allow requirements captured earlier by an RE to be verified by the client by visualizing low-fidelity prototypes in a form of Essential User Interface prototypes and a more concrete form-based UI in order to validate requirements. We have developed automated tool support for our approach and evaluated our prototype tool using an end user study. The results of this evaluation are promising. As future work, we plan to enhance our EUI library with more EUI patterns to support wider domains of application. We plan to develop domain specific GUI templates for the form based prototype. It would be useful to integrate our EUI library with UML models. We are planning an industrial example evaluation with real software clients.

ACKNOWLEDGEMENTS

We thank John Hosking and Jun Huh for their assistance. This research is funded by the: PReSS Account of University of Auckland; FRST Software Process and Product Improvement Project; Ministry Of Higher Education Malaysia & Universiti Teknikal Malaysia Melaka.

REFERENCES

[1] M.Kamalrudin, J.G., John Hosking. *Tool Support for Essential Use Cases to Better Capture Software Requirements,* Proc. 25th IEEE/ACM International Conference on Automated Software Engineering (ASE'10), ACM. 2010. Antwerp, Belgium.

[2] S.W. Ambler, *The Object Primer: Agile Model-Driven Development with UML 2.0 (3rd ed.)*. 2004, New York Cambridge University Press.

[3] K.Schneider, *Generating Fast Feedback in Requirements Elicitation*, in *Requirements Engineering: Foundation for Software Quality*. 2007. p. 160-174.

[4] V.Gervasi, and D. Zowghi, *Reasoning about inconsistencies in natural language requirements.* ACM Trans. Softw. Eng. Methodol., 2005. **14**(3): p. 277-330.

[5] A.Kozlenkov and A. Zisman, *Are their design specifications consistent with our requirements?*, Proc. IEEE Joint International Conference on Requirements Engineering,(ICRE02) IEEE Press, 2002. pp. 145-154.

[6] G.Engels, R.heckel, and M.Kuster, *The Consistency Workbench: A tool for consistency management in UML-based development*, in *UML 2003—the Unified Modeling Language*, P. Stevens, Whittle, J., Booch, G, Editor. 2003, Springer, Berlin Heidelberg New York. p. 356–359.

[7] A.Katasonov and M. Sakkinen, *Requirements quality control: a unifying framework.* Requirements Engineering, 2006. **11**(1): p. 42-57.

[8] Z.Jia, C.K. Chang, and C. Jen-Yao. *Mockup-driven fast-prototyping methodology for Web requirements engineering*. in *Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International*. 2003.

[9] Xiping, S. *S-RaP: A Concurrent Prototyping Process for Refining Workflow-Oriented Requirements*. 2005.

[10] S.Robertson, J.Robertson, *Mastering the Requirements Process (2nd Edition)*. 2006: Addison-Wesley Professional.

[11] N.Sukaviriya et al., *User-Centered Design and Business Process Modeling: Cross Road in Rapid Prototyping Tools*, in *Human-Computer Interaction – INTERACT 2007*, C. Baranauskas, et al., Editors. 2007, Springer Berlin / Heidelberg. p. 165-178.

[12] M.Kamalrudin, J.Grundy, J.Hosking, "*Managing consistency between textual requirements, abstract interactions and Essential Use Cases"*, Proc. 34th Annual IEEE International Computer Software and Applications (COMPSAC 2010). IEEE. Seoul, Korea.

[13] M.Kamalrudin, J.Hosking, J.Grundy. "*Improving Requirements Quality using Essential Use Case Interaction Patterns"*. in Proc.33rd International Conference of Software Engineering (*ICSE'11)*. 2011. ACM. Honolulu, Hawaii, USA.

[14] M.Kamalrudin, J. Hosking, and J. Grundy. *MaramaAI: Automated and Visual Approach for Inconsistency Checking of Requirements*. in Proc. 18th IEEE International Requirements Engineering Conference (RE2010), 2010. IEEE. Sydney,Australia.

[15] S.W.Ambler. *Essential (Low Fidelity) User Interface prototypes*. 2003-2009 Available from: www.agilemodeling.com/artifacts/essentialUI.htm.

[16] L.L.Constantine. and A.D.L. Lockwood, "Usage-centered software engineering: an agile approach to integrating users, user interfaces, and usability into software engineering practice", in Proc. 25th International Conference on Software Engineering (ICSE'03). 2003, IEEE Computer Society, Portland, Oregon.

[17] L.L.Constantine. and A.D.L. Lockwood, "*Software for use: a practical guide to the models and methods of usage-centered design"*. 1999, ACM Press/Addison-Wesley Publishing Co. 579.

[18] M.Kamalrudin, J.Hosking, J.Grundy. *MaramaAI: Tool support for capturing requirement and checking the inconsistency, Proc. 21st Australian Software Engineering Conference (ASWEC'10)*, 2010. Auckland, New Zealand: IEEE Computer society.

[19] J.Grundy, J.Hosking,J Huh,. and N.Li, *Marama: an Eclipse meta-toolset for generating multi-view environments*. Proc. *IEEE/ACM International Conference on Software Engineering 2008(ICSE'08)*. May 2008, Liepzig, Germany: ACM Press.

[20] S.Ogata and S. Matsuura, *Evaluation of a use-case-driven requirements analysis tool employing web UI prototype generation.* WSEAS Trans. Info. Sci. and App., 2010. **7**(2): p. 273-282.

[21] J.Vijayan and G. Raju, *Requirements Elicitation Using Paper Prototype*, in *Advances in Software Engineering*, T.-h. Kim, et al., Editors. 2010, Springer Berlin Heidelberg. p. 30-37.

[22] Pedro J.Molina, S.M., O.Pastor, "*JUST-UI: A User Interface Specification Model*, in *Computer-Aided Design of User Interfaces III"*, Proc. 4th International Conference on Computer-Aided Design of User Interfaces (CADUI'2002).2002. Valenciennes, France.