Tool-based gradual User Modeling for Usability Engineering

Anna Hüttig
University of Luebeck
Ratzeburger Allee 160
D-23562 Luebeck, Germany
huettig@imis.uni-luebeck.de

Michael Herczeg
University of Luebeck
Ratzeburger Allee 160
D-23562 Luebeck, Germany
herczeg@imis.uni-luebeck.de

ABSTRACT
This contribution illustrates how software developers can be supported systematically in user analysis and user centered design. Particularly it has been explored how user models can be integrated in the entire development process in a reasonable and gainful manner. For this purpose, a module for user analysis within the Usability-Engineering-Repository (UsER) is presented. The system is based on an innovative concept of gradual user modeling with several levels of abstraction that is guiding and simplifying the process of practical user modeling. The design of the module was validated with the aid of formative expert evaluation and the realized application was evaluated summatively.

Author Keywords
HCI, Usability Engineering, UCD, User Analysis, User Modeling, User Classes, Personas, Archetypes

ACM Classification Keywords
D.2.2 Design Tools and Techniques; H.1.2 User/Machine Systems; H.5.2 User Interfaces

INTRODUCTION
Besides its functionality, the usability of a system as key factor contributes to corporate performance by better customer satisfaction and development of sales [23]. To reach higher grades of usability, it is essential to determine the relevant context of use before specifying the detailed software requirements. Especially, the analysis of the different users of the software solution, their needs and capabilities seem to be a central aspect. The development and distribution of methods to support this area of analysis evolved in the last years but still lacks a systematic and consequent transfer into development processes and the products developed.

ECCE 2015, July 01 - 03, 2015, Warsaw , Poland
 ISBN 978-1-4503-3612-3/15/07…$15.00
DOI: http://dx.doi.org/10.1145/2788412.2788423

Currently, usability tools add little to the success of usability in software companies [23]. However, supportive tooling may improve the quality of user interfaces. By providing flexible tools, the implemented methods stay adaptable to the development context and the available resources and existing processes can be complemented purposefully and enriched holistically [19, 20].

USER MODELING IN SOFTWARE DEVELOPMENT
User models in particular are suited to support software development in many ways. Serving as independent artifacts, they can be consulted as a foundation for discussion and combined with other usability methods. Therefore, tool support for user analysis and user modeling to create a user model should be a substantial component within frameworks for the development of interactive systems. Problems that have to be faced – besides the quite common total lack of user models in general – are the creation of unsuitable and ill-defined models and the insufficient transfer of user models into the ongoing system development process.

Modeling Techniques
Concerning the modeling of users, several techniques have been developed: While in *Usage-Centered Design* more abstract models like *Actors* or *Role Descriptions* have been used to analyze usage patterns [2], *User-Centered Design* aims for more realistic models. An obvious first step to handle the diversity of users is their division into *User Classes* [22]. Main distinctive features can be – depending on the product's context – the user's goals concerning the target system, the technical or use-oriented level of experience or the organizational role captured by the users [1, 9, 17]. Latter shows the tight interplay between role descriptions and user classes. Class descriptions can reach from simple, often assumption-based category depictions to detailed, well-grounded *User Profiles* [8].

However, user classes remain abstract and offer just a rough differentiation telling little about the real user needs. More vivid are user descriptions that depict a single concrete, fictive person as a representative for a specific group of users. Popular methods are different types of *Personas* [3]. They consist of detailed, mainly narrative descriptions of fictive persons often based on extensive studies and data of real people [17]. The potential and handling of Personas has been extensively described in literature [14].

Another form of modeling is the *Archetype* or – with rather negative connotation – the *Stereotype*. Here, the archetype shall be understood as a denomination for a putative precisely defined class of persons with specific characteristics that is – at least along general lines – inter-individually valid. The delimitation to user classes and to concrete user descriptions is smooth and often unclear. Archetypes represent, like user classes, a type of users, though these are more precisely and demonstratively defined. They are suited to mediate specific characters of a category (e.g. "the shuttling business man"), but also tend to caricatures so that the values of these descriptions are often limited [9]. However, an advantage of archetypes is their simple creation since only few or even no user data has to be collected [7, 21].

Furthermore, so-called *Extreme Characters* are described as narrative, extreme personalities with exaggerated emotional attitudes [5]. Aim here is to foster the design creativity by deviating from only regarding prototypical characters of a specific target group.

Gradual User Modeling from abstract to concrete

Practice reveals that user modeling often begins with abstract classifications that help to provide an overview of the users but do not give a vivid image of them. Refinement and differentiation of rough user models to individual user descriptions should be encouraged in particular. Therefore, a theoretical foundation that states the possible levels of refinement among the user models is required. One obvious feature of differentiation is the degree of abstraction: Different increments correspond to different levels of concretion with more and more details about the users. A hierarchical structure of the models is therefore self-evident. Regarding established modeling techniques allows a 3-stage division in user classes, archetypes and individual models, as a first approach [9]. However, a problem of this approach is the strict separation between user classes and archetypes that on the one hand cannot be precisely defined – especially concerning attribution – and on the other hand cannot easily be controlled by the modelers. In contrast to this, a basic 2-stage separation distinguishing a class level and an individual level is serving the purposes much better. However within the first level, user classes may be defined in arbitrary forms and refined into kinds of archetypes continuously. The individual layer below holds descriptions that picture individual and concrete users.

A remaining open question is the form in which the different user descriptions on the abstraction levels are correlating. Since user classes are often just organizational role descriptions, it appears that subclasses can be derived from several superclasses. Thus, pure single inheritance is not sufficient. Instead, references to the superior and derived models are introduced. The derivation can be handled in several ways. Since stepless refinement shall be possible within class level, the (decoupled) cloning in the sense of *Prototype-based Programming* [6] is a possible solution. When a model, e.g. a Persona, shall be created on the individual layer, this can be done directly or by derivation from a user class model. However, cloning of instances is not without problems since the attributes, respectively the used templates of a Persona, are quite different to those of a user class. Furthermore for derivations within the individual level, fixed rules concerning the inheritance of content and template cannot easily been given since the modeler's intentions in refining an individual model cannot be foreseen. Hence, further refinements within the individual layer should be permitted.

Usage of User Models within the development process

User models can only unfold their entire potential when they are integrated in the ongoing development process during and after their creation and refinement. Especially Personas seem to be helpful. Process models like the *Goal-Directed Design* [3, 4] or the *Concept Development Process* for requirements engineering [18] regard user descriptions within the development process and pursue the idea of not letting the models become fixed and more or less dead artifacts after their creation. Instead, they should permanently be present, accompany and support the development process together with other usability methods and also evolve and be enhanced themselves during the design and implementation phases of product development. Thus, the models can serve as vivid actors within scenarios or can be representatives for organizational entities. Furthermore, a close relationship between task analysis and requirements using links between entities is possible since tasks and features can serve as goals of a modeled user. The relationship of a user model to other entities like tasks or requirements has often been illustrated via matrix models or diagrams like in *Persona-Weighted Feature Matrix*, *Scenario Count Per Persona* [1] or *Personas-Viewpoints-Requirements Matrix* [18].

THE USABILITY-ENGINEERING-REPOSITORY USER

The web-based Usability-Engineering-Repository UsER supports software development processes by providing various modules [12, 16]. These mainly represent usability engineering methods. All information gathered is modeled as entities by one of the UsER modules and may be linked to entities of other modules. Using these semantic relationships, networks of knowledge about the work system can be constructed and by that supporting collaborative working of the system developers. Besides the versatile cross-references between entities, the analysts and developers can select the modules as needed. Usually a project will be organized in a classical linear structure as in conventional development documents. Every chapter in UsER provides the functionality of a specific method module. They can be dragged into the linear organized administration area of the current development document. By this, hierarchically structured specification documents can be constructed and the development document itself becomes starting and anchor point for the different development tools. This is an important change of

paradigm, leading from classical text- or table-centered specifications into active semantically modeled documents with a classical linear outer structure and an inner structure of semantic networks for holistic analysis, design and evaluation of interactive systems.

Apart from the user analysis module described here, UsER offers a variety of other modules, e.g. for collecting and managing requirements, analyzing organizational structures or describing scenarios [11]. Modules can be used in several phases within an engineering process (see Figure 1).

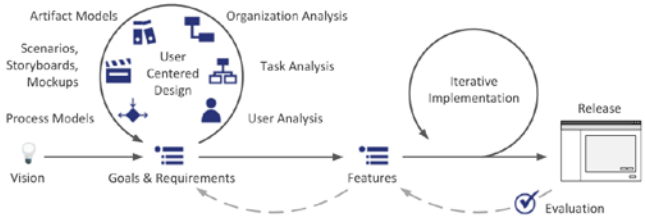

Figure 1. Possible engineering process supported by UsER.

User Modeling in UsER

The developed user analysis module for UsER supports software designers in user analysis by providing a systematic approach of gradual 2-stage user modeling and uses the integration into the framework to reuse the created models in the further development process. Initially, the module offers a graphical and therefore descriptive approach: On a canvas, entities for individual or class models can be generated und the relationships among them can be expressed. Thereby, user model entities can be created arbitrarily but can also be derived from each other. The theoretical analysis of gradual modeling led to the implemented concept as shown in Figure 2.

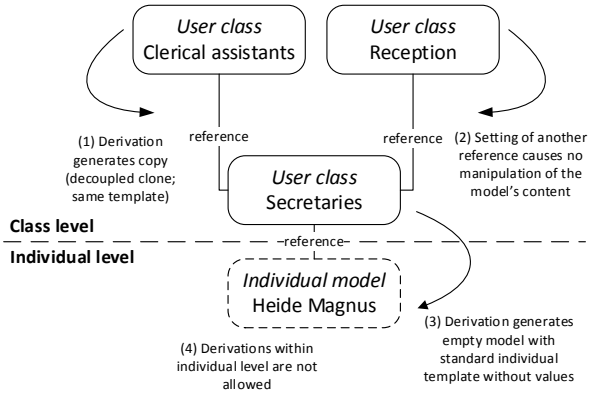

Figure 2. Realized concept for gradual user modeling.

Within the level of user classes, arbitrary refinements are possible whereby the derivation of a new individual model creates a decoupled clone. Thus, later changes in the superclasses will have no impact on the derived model or vice versa, but the relations of derivation still stay visible through the references. Additionally, the module offers templates for individual and class models. Templates can be created, designed and extended flexibly by the user. The suggested attributes and their arrangement in categories arose from extensive literature research [1, 2, 4, 8, 9, 15, 17, 22]. The concept of linking entities offered by UsER has been extended in a way that special kinds of attributes can reference specific entities in other modules. Construction and management of templates is done inside a special view and thereby separated from the use of the templates, supporting standards within the software company.

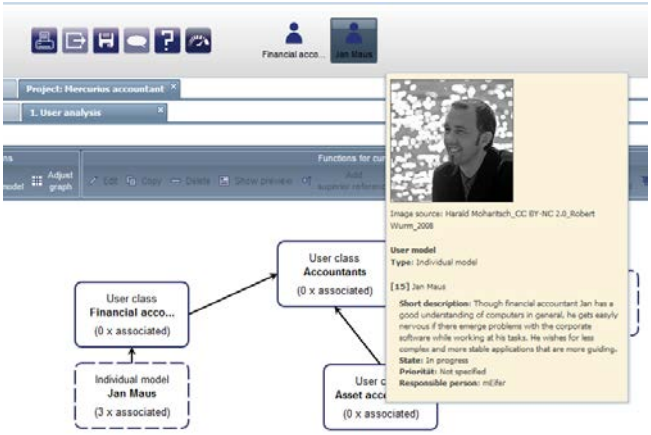

Figure 3. The pinning feature.

Additionally to linking, the pinning feature (see Figure 3) acts as another function that raises the presence of user models notably. Using this feature, models may permanently be visualized in the screen header of the current project in form of icons. Furthermore, the user models can be used outside of the framework via printing and export features.

CONCLUSIONS

The user analysis module within the UsER environment has been designed to offer a tool providing a flexible yet guiding method for user analysis and modeling. A 2-level reference model for gradual modeling with class and individual descriptions in combination with a template concept has been implemented. It supports an integrated access to different modeling techniques and allows the differentiation from abstract to concrete user descriptions. By the integration within the development platform UsER, the extension of explicit linking options between analysis and design entities and the intensified combination of different usability methods, the consideration of the user's needs within the development process can be strengthened.

The prototypical system offers a solution to cope with the challenges and lack of practical tool support for user modeling. Important features for future work are the support of the collection of user information as an inseparable aspect of user analysis as well as the stronger interlinking of implementation results (handled UsER-requirements) with the modeled user's goals and needs. Besides continuous formative evaluation, the module has been evaluated successfully summatively via expert interviews, resulting in valuable findings concerning the

practical benefits of the realized system. Its flexibility, the template concept and the interconnectedness of the user models were well received. Furthermore, first insights into the usability of the realized system could be collected but need to be analyzed further in ongoing user tests.

REFERENCES

1. T. Adlin and J. Pruitt. 2010. *The Essential Persona Lifecycle: Your Guide to Building and Using Personas*. San Francisco: Morgan Kaufmann Publishers Inc.
2. L. Constantine. 2006. Users, Roles, and Personas. In *The Persona Lifecycle*, J. Pruitt and T. Adlin (eds.). Amsterdam: Morgan Kaufmann/Elsevier. 499-519.
3. A. Cooper. 1999. *The Inmates are Running the Asylum*. Indianapolis: SAMS.
4. A. Cooper, R. Reimann, and D. Cronin. 2007. *About Face 3: The Essentials of Interaction Design* (3rd. ed.). Indianapolis: Wiley Publishing.
5. J. P. Djajadiningrat, W. Gaver, and J. W. Fres. 2000. Interaction relabelling and extreme characters: methods for exploring aesthetic interactions. In *Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*. New York: ACM. 66-71.
6. C. Dony, J. Malenfant, and D. Bardou. 1998. Classifying Prototype-based Programming Languages. In *Prototype-based Programming: Concepts, Languages and Applications*, J. Noble, A. Taivalsaari, and I. Moore (eds.). Singapore: Springer. 17-45.
7. J. Grudin. 2006. Why Personas work: the psychological evidence. In *The Persona Lifecycle*, J. Pruitt and T. Adlin (eds.). Amsterdam: Morgan Kaufmann/Elsevier. 643-663.
8. M. G. Helander, T. K. Landauer, and P. V. Prabhu. 1997. *Handbook of Human-Computer Interaction* (2nd. ed.). New York: Elsevier Science Inc.
9. M. Herczeg. 2009. *Software-Ergonomie. Theorien, Modelle und Kriterien für gebrauchstaugliche interaktive Computersysteme* (3rd. ed.). München: Oldenbourg.
10. M. Herczeg, M. Kammler, and A. Roenspieß. 2012. Prozessorientierte Entwicklung von aufgaben- und ereignisorientierten Benutzungsschnittstellen für die Prozessführung mit Hilfe eines Usability-Engineering-Repositories (UsER). In *Fortschrittliche Anzeigesysteme für die Fahrzeug- und Prozessführung: 54. Fachausschusssitzung Anthropotechnik der Deutschen Gesellschaft für Luft- und Raumfahrt*, M. Grandt and S. Schmerwitz (eds.). Koblenz: DGLR. 1-15.
11. M. Herczeg, M. Kammler, T. Mentler, and A. Roenspieß. 2013. The Usability Engineering Repository UsER for the Development of Task- and Event-based Human-Machine-Interfaces. In *Proc. 12th IFAC, IFIP, IFORS, IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems*. IFAC, 483-490.
12. M. Kammler, A. Roenspieß, and M. Herczeg. 2012. UsER: Ein modulares Usability-Engineering-Repository. In *Mensch & Computer 2012: interaktiv informiert – allgegenwärtig und allumfassend*. München: Oldenbourg Verlag. 333-336.
13. D. J. Mayhew. 1999. *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design*. San Francisco: Morgan Kaufmann.
14. L. Nielsen. 2013. *Personas - User Focused Design*. London: Springer.
15. G. Olsen. 2004. *Persona creation and usage toolkit*. Retrieved March 25, 2014 from http://www.interactionbydesign.com/presentations/olsen_persona_toolkit.pdf
16. M. Paul, A. Roenspieß, and M. Herczeg. 2013. UsER – Ein prozessorientiertes Entwicklungssystem für Usability-Engineering. In *Mensch & Computer 2013: Interaktive Vielfalt*, S. Boll, S. Maaß, and R. Malaka (eds.). München: Oldenbourg Verlag. 181-190.
17. J. Pruitt and T. Adlin. 2006. *The Persona Lifecycle: Keeping People in Mind Throughout Product Design*. San Francisco: Morgan Kaufmann.
18. W. Sim and P. Brouse. 2014. Empowering Requirements Engineering Activities with Personas. In *Procedia Computer Science*. 28. 237–246.
19. C. Stropp and S. Brandenburg. 2014. *Der Stellenwert von Usability in kleinen und mittelständischen Unternehmen*. Retrieved March 25, 2014 from http://www.usetree.de/wp-content/uploads/2014/03/Themenschwerpunkt_Juni_Usability_in_KMU.pdf
20. M. Thüring. 2013. *Herausforderungen für die Bereitstellung von Usability-Dienstleistungen bei KMU*. Retrieved November 21, 2014 from http://www.usability-in-germany.de/kos/WNetz?art=File.show&id=666.
21. P. Turner and S. Turner. 2011. Is stereotyping inevitable when designing with personas? In *Design Studies.* 32(1). 30-44.
22. X. Wang. 2011. *Personas in the User Interface Design*. Retrieved November 21, 2014 from http://pages.cpsc.ucalgary.ca/~saul/wiki/uploads/CPSC681/topic-wan-personas.pdf.
23. M. Woywode, A. Mädche, D. Wallach, and M. Plach. 2012. *Gebrauchstauglichkeit von Anwendungssoftware als Wettbewerbsfaktor für kleine und mittlere Unternehmen (KMU): Abschlussbericht*. Retrieved November 21, 2014 from https://ub-madoc.bib.uni-mannheim.de/29891.