07081 Abstracts Collection
End-User Software Engineering
— Dagstuhl Seminar —

Margaret M. Burnett[1], Gregor Engels[2], Brad A. Myers[3] and Gregg Rothermel[4]

[1] Oregon State University, US
burnett@cs.orst.edu
[2] University of Paderborn, DE
engels@upb.de
[3] Carnegie Mellon University - Pittsburgh, US
bam@cs.cmu.edu
[4] University of Nebraska - Lincoln, US
grother@cse.unl.edu

Abstract. From 18.01.07 to 23.02.07, the Dagstuhl Seminar 07081 "End-User Software Engineering" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. End user software engineering, end-user programming, human-computer interaction, programming language design

07081 Executive Summary – End-User Software Engineering

From 18.01.07 to 23.02.07, the Dagstuhl Seminar 07081, "End-User Software Engineering", was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. This document summarizes the event.

Keywords: End user software engineering, end-user programming, human-computer interaction, programming language design

Joint work of: Burnett, Margaret M.; Engels, Gregor; Myers, Brad A.; Rothermel, Gregg

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1098

Gender HCI Issues in End-User Software Engineering Environments

Laura Beckwith (Oregon State University, USA)

Although gender differences in a technological world are receiving significant research attention, much of the research and practice has aimed at how society and education can impact the successes and retention of female computer science professionals. The possibility of gender issues within software, however, has received almost no attention. We hypothesize that factors within software have a strong impact on how well female problem solvers can make use of the software. Evidence from other fields and investigations of our own have revealed evidence supporting this hypothesis.

Keywords: End-user software engineering, gender, self-efficacy

Joint work of: Beckwith, Laura; Burnett, Margaret; Wiedenbeck, Susan

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1076

End User Programming for Scientists: Modeling Complex Systems

Andrew Begel (Microsoft Research - Redmond, USA)

Towards the end of the 20th century, a paradigm shift took place in many scientific labs. Scientists embarked on a new form of scientific inquiry seeking to understand the behavior of complex adaptive systems that increasingly defied traditional reductive analysis. By combining experimental methodology with computer-based simulation tools, scientists gain greater understanding of the behavior of systems such as forest ecologies, global economies, climate modeling, and beach erosion. This improved understanding is already being used to influence policy in critical areas that will affect our nation's future, and the world's.

Keywords: StarLogo, Science, Modeling, Complex Systems

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1077

Empirical Foundations for EUSE and Interdisciplinary Design for EUSE

Alan Blackwell (Cambridge University, GB)

How does EUSE research build on empirical studies of programmers, and what kinds of empirical research might provide foundations for future EUSE research?

My own work on interdisciplinary design draws comparisons across academic and professional boundaries, applying the results to the design of new technologies, and the critical assessment of technology.

Keywords: Interdisciplinary design, Empirical Studies of Programmers, Psychology of Programming, Real World Research

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1078

Empirical Studies in End-User Software Engineering and Viewing Scientific Programmers as End-Users – POSITION STATEMENT –

Jeffrey Carver (Mississippi State Univ., USA)

My work has two relationships with End User Software Engineering. First, as an Empirical Software Engineer, I am interested in meeting with people who do research into techniques for improving end-user software engineering. All of these techniques need to have some type of empirical validation. In many cases this validation is performed by the researcher, but in other cases it is not. Regardless, an independent validation of a new approach is vital. Second, an area where I have done a fair amount of work is in software engineering for scientific software (typically written for a parallel supercomputer). These programmers are typically scientists who have little or no training in formal software engineering. Yet, to accomplish their work, they often write very complex simulation and computation software. I believe these programmers are a unique class of End-Users that must be addressed

Keywords: Empirical Studies

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1079

What is an end user software engineer?

Steven Clarke (Microsoft Research - Redmond, USA)

The group of people described as end user software engineers are a very large and diverse group. For example, research scientists building simulations of complex processes are described as end user software engineers as are school teachers who create spreadsheets to track the progress of their students. Given the difference in background and domains in which different end user software engineers work, I argue that it is important to distinguish between different categories of end user software engineers. Such distinctions will enable us to determine which tools and techniques are appropriate for which types of end user software engineers. Indeed, such distinctions will also make clear the differences and similarities between end user software engineers and so called professional software engineers.

Keywords: Personas, End user software engineer,Professional software engineer

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1080

Software environments for supporting End-User Development

Maria Francesca Costabile (University of Bari, I)

Our work on End-User Development primarily focuses on the needs of a specific community of users, namely professionals in diverse areas outside of computer science, such as engineers, physicians, geologists and physicist, who are not professional programmers. We refer to them as domain experts. We developed a participatory design methodology, called SSW (Software Shaping Workshop) methodology, aimed at designing software environments that support domain experts to become co-designers of their tools. The different stakeholders can contribute their own views on the problem to design, development and maintenance of an application, using their own languages and notations.We also proposed a model of the Interaction and Co-Evolution processes (ICE model) occurring between users and system. It extends a previous model of Human-Computer Interaction by considering an important phenomenon occurring during the use of interactive systems, called co-evolution of users and systems.

Keywords: Customized software environments, meta-design, participatory design, domain expert

Joint work of: Costabile, Maria Francesca; Piccinno, Antonio

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1081

Meta-UI for Ambient Spaces: Can MDE help?

Joelle Coutaz (Université de Grenoble, F)

This position paper introduces the concept of Meta-UI and outlines our technical approach. This approach draws upon the flexibility of MDE and SOA to allow users, by the way of a meta-UI, to control the ambient space in which they live.

Keywords: Meta-UI, MDE, SOA

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1082

Rethinking the Software Life Cycle: About the Interlace of Different Design and Development Activities

Yvonne Dittrich (IT University of Copenhagen, DK)

Software engineering research addresses professional ways of designing, developing and implementing software. So far, software engineering more or less takes for granted that software professionals have control over the material implementation of a piece of software. Though users might use the software innovatively or even customise it, neither end-user tailoring (EUT) nor end-user development (EUD) are treated systematically regarding the impact of deferring part of the design to the use context on software development technologies or processes. Especially the development, adaptation and configuration of software products, software that is used by more than one user in more than one organisation makes visible that different parallel ongoing development activities often distributed over more than two organisations have to be coordinated.

Keywords: End User Development, Software Processes

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1084

Requirements and Modeling for End-User Developers

Gregor Engels (Universität Paderborn, DE)

Eliciting the requirements and creating a model of a software system are standard activities in the development process of professional software development. The talk discusses whether these two development phases are also present in end-user software development and how they could look like. It is argued that one has to distinguish between at least two types of end-user software developers. Those, who are not professional software developers, but work in an engineering domain and follow stepwise development processes. They are used to have requirements specifications as well as models, too. But, non-professional, non-engineering end-users, e.g. spreadsheet developers, don't and would not like to distinguish between different steps in the development process. Therefore, we propose to hide the distinction between these different steps by closely interconnecting requirements specification, models and code, and by putting them into one development box. By offering appropriate interface functions like create, adapt, refine, etc. to the box, the end-user is supported in developing software without being aware that he is undergoing a stepwise refinement process from requirements specifications towards concrete code.

Keywords: End-User Modeling

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1085

Exploiting Domain-Specific Structures For End-User Programming Support Tools

Martin Erwig (Oregon State University, USA)

In previous work we have tried to transfer ideas that have been successful in general-purpose programming languages and mainstream software engineering into the realm of spreadsheets, which is one important example of an end-user programming environment.

More specifically, we have addressed the questions of how to employ the concepts of type checking, program generation and maintenance, and testing in spreadsheets. While the primary objective of our work has been to offer improvements for end-user productivity, we have tried to follow two particular principles to guide our research.

(1) Keep the number of new concepts to be learned by end users at a minimum.

(2) Exploit as much as possible information offered by the internal structure of spreadsheets.

In this short paper we will illustrate our research approach with several examples.

Keywords: Spreadsheet, program analysis

Joint work of: Abraham, Robin; Erwig, Martin

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1086

Meta-Design: A Conceptual Framework for End-User Software Engineering

Gerhard Fischer (University of Colorado, USA)

In a world that is not predictable, improvisation, evolution, and innovation are more than a luxury: they are a necessity. The challenge of design is not a matter of getting rid of the emergent, but rather of including it and making it an opportunity for more creative and more adequate solutions to problems.

Meta-design is an emerging conceptual framework aimed at defining and creating social and technical infrastructures in which new forms of collaborative design can take place. It extends the traditional notion of system design beyond the original development of a system. It is grounded in the basic assumption that future uses and problems cannot be completely anticipated at design time, when a system is developed. Users, at use time, will discover mismatches between their needs and the support that an existing system can provide for them. These mismatches will lead to breakdowns that serve as potential sources of new insights, new knowledge, and new understanding.

Keywords: Meta-design, consumers and designers, unself-conscious cultures of design

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1087

Dependability in Web Software

Marc Fisher (University of Nebraska, USA)

The web is an increasingly important platform used for a wide variety of tasks on a regular basis. And as the web becomes more important, the ways in which it is used grows increasingly sophisticated. End users build web pages and applications, use web applications in new and unexpected ways and use web macro tools to automate web-based tasks. All of these tasks are error-prone. In addition, they often depend on external components outside of the control of the developer or end user. Therefore we have been developing tools and methodologies to assist users with these

Keywords: Web Applications, Dependability

Joint work of: Elbaum, Sebastian; Fisher II, Marc; Rothermel, Gregg

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1089

A Methodology to Improve Dependability in Spreadsheets

Marc Fisher (University of Nebraska, USA)

Spreadsheets are one of the most commonly used end user programming environments. As such, there has been significant effort on the part of researchers and practitioners to develop methodologies and tools to improve the dependability of spreadsheets. Our work has focused on the development of the "What You See Is What You Test" (WYSIWYT) family of techniques. WYSIWYT is designed to be seamlessly integrated into a spreadsheet environment and the user's development processes. It uses visual devices that are integrated into the user's spreadsheet to guide the process of finding and fixing problems with the spreadsheet.

Keywords: Spreadsheets, Dependability, Testing

Joint work of: Burnett, Margaret; Fisher II, Marc; Rothermel, Gregg

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1088

Designers Need End-User Software Engineering

Mark Gross (Carnegie Mellon Univ. - Pittsburgh, USA)

This position paper for the End-User Software Engineering workshop outlines three systems that employ end user programming for designers: a constraint-based design environment; a sketch recognition interface for knowledge based systems, and a physical programming environment for building modular robots.

Keywords: Design, end-user, programming, physical, graphics, constraints

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1090

Barriers to Successful End-User Programming

Andrew J. Ko (Carnegie Mellon University, USA)

In my research and my personal life, I have come to know numerous people that our research community might call end-user programmers. Some of them are scientists, some are artists, others are educators and other types of professionals. One thing that all of these people have in common is that their goals are entirely unrelated to producing code. In some cases, programming may be a necessary part of accomplishing their goals, such as a physicist writing a simulation in C or an interaction designer creating an interactive prototype. In other cases, programming may simply be the more efficient alternative to manually solving a problem: one might find duplicate entries in an address book by visual search or by writing a short Perl script.

Keywords: End-user programming, learning, empirical studies

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1091

End-User Software Engineering Position Paper

Henry Lieberman (MIT - Cambridge, USA)

This position paper outlines work on making programming easier, and its relationship to end-user software engineering.

Keywords: End-user programming

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1092

End Users Creating More Effective Software

Brad Myers (Carnegie Mellon University, USA)

This position paper briefly summarizes paradigms used to create end-user software.

Keywords: End user software engineering

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1093

End-User Design

Alexander Repenning (University of Lugano, CH)

Are UML diagrams a good tool to teach middle school students how to make video games? Probably not, but what kinds end-user design aids such as mental models, scaffolding structures, examples or other kinds of objects to think we can we give to end-users in order to gradually introduce them to good programming practice?

Keywords: End-user programming, end-user development, computers in education, programming environment for kids

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1099

Position paper for EUSE 2007 at Dagstuhl

Mary Beth Rosson (Penn State University, USA)

This brief position paper summarizes several facets of research underway by the Informal Learning in Software Development group at Pennsylvania State University. The focus of the work reported is on end user web development, with discussion of user needs and tools that might help to meet these needs.

Keywords: End user web development

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1092

End-User Software Engineering and Professional End-User Developers

Judith Segal (The Open University, GB)

There is a great variety of end user developers and a great variety of contexts within which they develop. End user developers may have little or no experience of using computers or may be adept coders in general purpose programming languages.

They may develop their software on their own over a few minutes or in groups over years. The software produced may be for their own use only or for a large community of users. It may be inconsequential or the consequences of its failure may be great. In this paper, we identify and discuss the problems of one particular group of end user developers – professional end user developers – who have no fear of coding and who develop software which plays a vital part in furthering their professional goals.

Keywords: Professional end user developers, scientific computing

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1095

Helping Everday Users Establish Confidence for Everyday Applications

Mary Shaw (CMU - Pittsburgh, USA)

End users obtain their desired results by combining elements of information and computation from different applications. Software engineering provides little support for identifying, selecting, or combining these elements – that is, for helping end users to design computational support for their own tasks. Software engineering provides even less support to help end users to decide whether the resulting system is sufficiently dependable – whether it will meet their expectations. Many users, especially end users, base judgments about software on informal and undependable information, and they draw conclusions with informal rather than rational decision methods. We have been developing support for everyday dependability, with an emphasis on expressing expectations in abstractions familiar to the user and on obtaining software behavior that reasonably satisfies those expectations. In this Dagstuhl I would like to explore the differences between everyday informal reasoning and the rational processes of computer science in order to develop means for establishing credible indications of confidence for end users.

Keywords: Everyday users, everyday dependability, data feeds, task level of abstraction, topes

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1096

End-User Development Techniques for Enterprise Resource Planning Software Systems

Michael Spahn (SAP Research - Darmstadt, D)

The intent of this position paper is to present the focus of interest of our end-user development (EUD) related research at SAP Research CEC Darmstadt.

As we are in an early phase of research, research topics will be presented rather than detailed results. We focus on investigating and applying EUD techniques suitable for enterprise resource planning (ERP) software systems, especially for small and medium-sized enterprises (SMEs). Our current research addresses the sub-domains of workflow management and business intelligence.

Keywords: End-User Development (EUD), Enterprise Resource Planning (ERP), Workflow Management, Business Intelligence (BI)

Joint work of: Spahn, Michael; Scheidl, Stefan; Stoitsev, Todor

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1097

End-user (further) development: A case for negotiated semiotic engineering

Clarisse de Souza (PUC-Rio de Janeiro, BR)

Semiotic Engineering views human-computer interaction as a special case of computer-mediated communication. In it, designers communicate to users their design vision about computer artifacts: what they are meant to do and how, but also what benefits they bring to users' lives (as perceived by designers). The semiotic theories of meaning subscribed by Semiotic Engineering postulate that meanings always evolve - there are no fixed meanings. Thus, the meanings users assign to computer artifacts and the meaningful situations in which they expect such artifacts to be valuable evolve. End user development is thus a requirement for "useful artifacts". Users should be able to encode new meanings and meaningfulness in them. However, there are limitations for this encoding: helping designers and users negotiate new meanings at interaction time, through the mediation of systems' interfaces, is thus a key issue for Semiotic Engineering. In this scenario, explanations - especially those about what artifacts cannot do, have not done, and why - are cruacially important.

Keywords: Semiotic engineering, explanations, representation codes

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2007/1083

End-user Programming of Ambient Narratives

Mark van Doorn (Philips Research Lab. - Eindhoven, NL)

Ambient Intelligence is a vision on the future of consumer electronics, telecommunications and computing in which devices move into the background while at the same time placing the user experience in the foreground. Producing Ambient Intelligent environments on a large scale is problematic however. First, it is technologically not possible in the foreseeable future to mass produce a product

or service that generates Ambient Intelligence, given the current state-of-the-art in machine learning and artificial intelligence. Second, it is economically not feasible to manually design and produce Ambient Intelligence applications for each person individually. One of the main research questions in creating such environments is the design of a system capable of supporting mass customization of ambient experiences by means of end-user programming. A brief outline of the approach taken to address this question is described including future research.

Keywords: Ambient intelligence, storytelling, hypertext, end-user programming

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1075