Matters in End-User Development: Enculturing Qualities

Volker Wulf, University of Siegen, Germany
Volkmar Pipek, University of Siegen, Germany
Mary-Beth Rosson, Penn State University, USA
Boris de Ruyter, Philips Research, The Netherlands

ABSTRACT

End-User Computing, End-User Development, End-User Software Engineering and related approaches have experienced an enormous wave of attention currently. As modern Information and Communication Technologies (ICT) pervade more and more areas not only of our professional, but also of our private lives, it becomes necessary for end users to understand and monitor their technological environments as well as to be able to configure or even program them. The third wave of End-User Computing poses new challenges to research in several disciplines of Computer Science and Information systems. The contributions to this Special Issue on End-User Development (EUD) cover those parts of the challenges that extend the scope of EUD by acknowledging and overcoming cultural breaches between professional programming and EUD. In this contribution we give a brief overview of these contributions as well as some comments on future research on EUD.

Keywords: End-User Computing; End-User Development; End-User Software Engineering; Research Challenges

INTRODUCTION

When computers entered the modern world in the middle of the twentieth century, all users of computers were also programmers: The mathematician and the physicist developed and entered their own program code and used the results for their scientific work. This first age of end-user computing ended with the increasing capacities of computers: It was possible to write more complex programs, code reuse became a necessity, programming tasks were delegated, and a
new organization of code artefacts as well as programming processes emerged. The discipline of Software Engineering was born, and with a sounder methodology on Programming came the structures of professionalization we have today: developers, interface designers, database administrators, etc. – and the “end user” emerged, who has neither the interest nor the competencies to create or significantly change the software tools that specialized the ‘Universal Machine’ to the very purposes end users needed for their work. In this ‘second age’ of end-user computing, the programming end user became a niche phenomenon while software products targeted standardized usages (e.g. text processing applications) of a mass market or became local infrastructures (e.g. individual software applications, enterprise resource planning systems) that are maintained and improved by trained specialists. But at the same time, the ‘configuring’ end user emerged as a new species: individualization and tailoring of software to a specific use environment became a necessity, and the interfaces for tailoring reached a level similar to the complexity of domain-specific or visual programming languages.

A third wave of end-user computing is currently in the making. This wave has so many facets that it is hard to keep track of all of them:

- the World Wide Web offered new ways for the presentation of and navigation in information (that need to be ‘programmed’);
- the Web 2.0 movement made every user a potential information/application provider;
- household devices from a DVD recorder to a video surveillance system became more elaborate (with concomitant needs for configuration and programming) - with the Ubiquitous Computing movement promising further devices and applications ahead;
- the increasing connectivity of devices, particularly in private households (about 124 Million households in Western Europe will be online in 2013, Nutham 2007), leads to very heterogeneous, individual, and complex distributed application scenarios. One of the differentiators identified by Bodine (2007) is the option to allow people to customize. End users want to be in control – even in complex technological scenarios, and even if a certain level of programming skills remains necessary.
- the Open Source movement produced a base of very complex software systems that are principally open for everybody’s contribution and participation; and
- almost every product emerges with an ICT-based extension (e.g. support webpages or online user communities) that allows to air views and suggestions for tweaking, changing or enhancing the product at hand – with the ‘prosumer’ movement exploring new ways of user-/consumer-initiated production even beyond ICT.

The research to analyze and support all these tendencies spans many scientific communities in and around Computer Science. The researchers who have contributed to this special issue
have participated in different research discourses from Human Computer Interaction (HCI), Software Engineering (SE), Computer Supported Cooperative Work (CSCW), Participatory Design (PD), Information Systems (IS), and Artificial Intelligence (AI). Concepts such as tailorability, configurability, end-user programming, visual programming, appropriation support, meta-design, natural programming, and programming by example have already formed a fruitful base, but they need to be better integrated, and the synergy between them needs to be fully exploited. Liebermann et al. (2006) defined End-User Development as a “set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artefact”.

The Second International Symposium on End-User Development (IS-EUD) that took place in Siegen, Germany, in March 2009 (Pipek et al. 2009), aimed at gathering together the different open ends of end-user computing. Historically grounded in the collaboration of two large research efforts, the US-based research network on end-user software engineering (EUSES) and the follow-up initiatives of the European Network of Excellence on End-User Development (EUD-NET), the symposium established a discussion forum that unites the different research discourses mentioned above. This synergistic effort will be continued 2011 in Bari, Italy, with the third symposium in the series. This special issue covers revised and extended versions of seven talks that have been given at the Siegen meeting. Our selection covers a rich variety of domains and approaches, and indicates a number of future research challenges that we will now summarize and comment upon. This special issue covers three contributions that address cultural breaches between professional and end-user programmers in testing and debugging (Burnett, also covering Gender aspects, in ‘End-User Software Engineering and Why It Matters’), with regard to dealing with the documentation of application programming interfaces (APIs, contribution by Myers et al. called ‘Studying the Documentation of an API for Enterprise Service-Oriented Architecture’), and with regard to strategies to establish a culture of participation in technology development (Fischer on ‘End User Development and Meta-Design: Foundations for Cultures of Participation’). In four additional contributions in a second special issue in the April edition of the JOEUC, collaborative aspects of End-User development will be in the focus. There, Cabitza and Simone (‘Woad: A framework to enable the end-user development of coordination-oriented functionalities’) deal with formal descriptions of end-user collaboration, Kafai et al. (‘Entering the clubhouse: Case studies of young programmers joining the online scratch communities’) report on experiences with online communities of very young programmers, Morch and Andersen (‘Mutual development: The software engineering context of end-user development’) present a case study on interactions between professional and end-user programmers, and Stevens et al. (‘Appropriation infrastructure: Mediating appropriation and production work’) describe possible tool support options for
this interactions. We now concentrate on the contributions covered in this issue.

2 DOMAINS IN END-USER DEVELOPMENT

The application domain is always an important issue in EUD concepts (e.g. Domain-oriented Design Environments, Fischer, 1994): It is never just another example of the application of a standardized concept; it is always something new that needs to be conceptually integrated into the approaches.

The contributions in this special issue focus to a different extent on specific domains. In framing research issues in End-User Software Engineering, Burnett provides an overview of quality issues in End-User Computing. Myers et al. focus on business domains and look at Service-Oriented Architectures in enterprise software. The domains covered by Fischer, in which 3D-visualizations were relevant (e.g. urban planning, health informatics), represent research approaches between professional and private parts of life, and document the growing need to consider EUD issues for the domains of our private lives. This expanded scope raises specific challenges, as de Ruyter and van de Sluis (2006) describe for Ambient Intelligence applications, or Grinter et al. (2009) with regard to home networks. For example, new domains arising in home environments are support for the elderly and support for house and energy management. This variety of domains will be continued in the second special issue on collaborative aspects of EUD, although the focus remains on business applications. The research on seemingly simpler business scenarios, e.g. spreadsheet applications, which started with Nardi (1993), has been further continued and refined elsewhere (see more examples in Ko et al. 2009).

3 RESEARCH CHALLENGES

In addition to domain-specific considerations for EUD approaches, this special issue presents a number of different perspectives on current research challenges. The broader diversity of domains entrains a methodologically more intense look at them—in the contributions presented in this special issue, experimental settings dominate the contributions, but the second special issue on collaborative aspects shows that qualitative research approaches have gained relevance. Another frequent issue in end-user computing is the formality with which users are being confronted when manipulating their technological infrastructures (Shipman & Marshall, 1999). Burnett leverages formal aspects of programming languages for her approach, and Myers et al. describe how terminological breaches (here: business vs. technological) complicate the orientation of end-user programmers. Fischer (and also Burnett) illustrate the power of visualisations to overcome these problems (another example for the mutual influence of visual and process aspects can be found in Subrahmaniyan et al. 2008, another example for integrating user annotations of software visualizations in service environments has been described by Paterno et al. 2008). Following ideas
similar to Fischer’s ‘meta-design’, further contributions in the second special issue on collaborative aspects of EUD will show that collaboration helps alleviating many problems with the abstract nature of programming as a task for end users. In our eyes, the various approaches and perspectives afford the following five research themes for future End-User Development Research:

- **Formality and Visualization of Interfaces for End-User Development:** To benefit from the services of a computer, a certain level of formalization is necessary. But to what extent can it be hidden from end users to support the ‘ease of development’ or even the ‘fun of development’ that will become more relevant for home and leisure usages of ICT? When is it dangerous to hide formal issues that would help understanding and correctly manipulating ICT infrastructures? What additional visualizations do we need to provide in order to make the inner workings as well as the consequences of manipulations transparent for end users?

- **Individual and Collaborative Support for End-User Development:** Many research approaches thus far have addressed the individual user sitting in front of the computer screen, but the increasing number of interconnected devices and applications connects the users in more reliable and flexible ways with one another—and with the device and application designers. What are good collaboration scenarios and how can hardware and software artefacts designed to afford or support these collaborations?

- **Tools vs. Education for Software Quality in End-User Development:** Addressing the amateur programmer makes software quality management a crucial issue in End-User Development. One way to address it is to provide tools that help to guarantee certain standards of good programming—but maybe can a sound education in programming be not only helpful, but even ‘fun’ (see examples in Kafai et al.)? Is increasing the ‘ease of development’ a wholly misguided strategy to address the issue of being a human caught in a network of technological devices and applications—would it be better to accept that ‘programming’ needs to be regarded as a cultural technique of the modern man, comparable to reading and basic mathematical skills?

- **The Infrastructure Challenge - Ubiquity and Taken-For-Grantedness in End-User Development:** With the growing importance of Mobile and Ubiquitous Computing applications and devices, ICT pervades more and more areas of life, and at the same time becomes a complex infrastructure because applications may span several devices and software artefacts. Moreover, the resulting technological infrastructure becomes something that end users take for granted. They become at the same time less aware and more dependent on this infrastructure, and being able to detect, understand and correct perceived or real technological breakdowns becomes an important competency. How can we develop technologies and EUD support that addresses infrastructural aspects like
the issue of taken-for-grantedness (Pipek and Wulf 2009) of ubiquitous ICT infrastructures?

• **Software Engineering Meets User-Driven Innovation:** While Software Engineering still follows a ‘product’ metaphor (e.g. industrial software engineering, Greenfield & Short, 2004; Clements & Northrop, 2002), researchers at the borderline of technology development and technology use are slowly coming to recognize that software and ICT devices are a specific matter that cannot be directly compared with prior ‘products’. While delivering its service (e.g. text processing), our ‘product’ may not only provide us with help manuals and configuration support, but it may also connect us with any other user working on similar texts, with any designer who developed this application the user uses – and all of this within the very same artefact. These possible ties between users and developers allow not only new options for technological support, but also new business models that are open for feedback and further development of ICT artefacts with the help of end users. In the field of business administration, the issue of User-Driven Innovation already has significantly gained importance (von Hippel 2002).

Meeting these research themes will further stress the relevance of the research around End-User Development, and may also lead to important changes for Software Engineering and other technology development disciplines.

**REFERENCES**


Kafai, Y. B., Fields, D. A., & Burke, W. Q. (in press). Entering the clubhouse: Case
studies of young programmers joining the online scratch communities. *International Journal of Organizational and End User Computing*, 22(2),


