

EUD-Net's Roadmap to End-User Development

Markus Klann

Fraunhofer Institute for Applied Information Technology (FhG/FIT)

Schloss Birlinghoven

53754 Sankt Augustin, Germany

+49 2241 14 2152

markus.klann@fit.fraunhofer.de

ABSTRACT

The article gives an overview of aspects, current approaches and promising strands of research for the subject of End-User Development (EUD). It is a condensed version of the EUD roadmap which has been produced within the European research project EUD-Net whose aim is to foster research and development in this field. The article concludes that empowering end-users to carry out substantial adaptations of IT-systems is an important contribution to letting them become active members of the information society.

Keywords

End-User Development, co-evolution, requirements engineering, information society

INTRODUCTION

The subject of End-User Development (EUD) is the focus of the ongoing European research project EUD-Net [3, 7]. The project's definition of EUD is as follows [2]: "*End User Development is a set of activities or techniques that allow people, who are non-professional developers, at some point to create or modify a software artefact.*"

The goal of EUD-Net is to create a joint vision of researchers and industry partners in this field and to provide ideas and guidelines for future research and development. A first step in this process has been to create a roadmap for the field. This roadmap gives an introduction to the topic of EUD, a survey of current approaches, methods and areas of application and points at promising strands of research.

In order to make EUD-Net's ongoing work available to a wider audience, this article provides a condensed view of the central aspects presented in EUD-Net's roadmap.

WHAT IS EUD AND WHY IS IT IMPORTANT?

People want IT-systems to meet their requirements. Capturing these requirements and letting software-professionals implement them is a workable approach only if the requirements can be identified and remain stable over time. Very

much in contrast to this the current development in professional life, education and also in leisure time is characterized by an increasing amount of change and diversity. Changes in work and business practices, changes concerning individual qualifications and preferences and changes in the dynamic environment, organizations and individuals act in. Diversity concerning people with different skills, knowledge, cultural background and physical or cognitive abilities, as well as diversity related to different tasks, contexts and areas of work. As most of the work done in organizations, and an increasing amount of peoples' activities outside of organizations is supported by IT-systems, there is a need for substantially more flexible systems that can easily be adapted to meet the changing and diversified requirements.

This insight, which developed in various fields of human-computer-interaction (HCI) and software-engineering, has now become focused in the new research paradigm of End-User Development (EUD). The goal of EUD is to empower end-users to adapt IT-systems themselves as much as possible, thus letting them become the initiators of a fast, cheap and tight co-evolution between themselves and the systems they are using. To allow for this level of end-user development, IT-systems must be made considerably more flexible and they must support the demanding task of EUD in various ways: they must be easy to use, to teach, understand, and learn. Also, users should find it easy to test and assess their EUD activities.

EUD has now found its first widespread use in commercial software, and end-users have taken it up with some success: recording macros in word processors, setting up spreadsheets for calculations and defining e-mail-filters. While these applications only realize a fraction of the EUD potential and still involve many issues, they illustrate why empowering end-users to develop the systems they are using is an important contribution to letting them become active citizens of the information society. One example for future use of EUD technology is the field of home appliances, i.e., all sorts of electronic devices that people will use at home and that will become interconnected and very flexible in the near future. This creates a mass-market where people will want to adapt systems to their specific contexts

and requirements and where they will value personalized, adaptive and anticipatory systems.

Given estimates like that of Brad Myers [6] (Carnegie-Mellon-University) that in 2005 there will be 55 million end-users doing EUD while there will be only 2.75 million software professionals, the importance of research on EUD becomes apparent. Not only to limit the damage caused by erroneous EUD activities but also to fully exploit the potential benefits of quick and precise system adaptations that only end-users can perform at a reasonable cost.

ASPECTS OF END-USER DEVELOPMENT

Enhancing user-participation in the initial design process of IT-systems is one step towards better capturing user requirements. Research is done on providing domain-specific, possibly graphical modeling languages that users find easy to express their requirements in. Such modeling languages are considered an important means to bridge the 'communicational gap' between the technical view of the software professionals and the domain-expert view of the end-users.

But as stated above, end-user requirements are increasingly diversified and changing and even at a given point in time they may be difficult to specify. Consequently, an initial design tends to become outdated or insufficient fairly quickly. Going through conventional development cycles with software-professionals to keep up with evolving end-user requirements would be too slow, time-consuming and expensive. While end-users are generally neither skilled nor interested in adapting their systems at the same level as software professionals, it is very desirable to empower users to continuously adapt their systems at a level of complexity that is appropriate to their individual skills and situation. Challenging the conventional view of 'design-before-use', new approaches try to establish 'design-during-use', leading to a process that can be termed 'evolutionary application development' [5].

System changes during use might be brought about by either explicit EUD activities of the end-users or by the system automatically changing itself to better meet its users' requirements. In the first case, the system is called adaptable, whereas in the second, adaptive.

Adaptability in the sense of EUD calls for a system flexibility that allows for adaptations that extend well beyond simple parameterizations, while being substantially easier than (re)programming. More precisely, a system should offer a range of different adaptation levels with increasing complexity and power of expression. This is to ensure that users can do simple adaptations easily and that they only have to accept a proportional increase in complexity for more complicated ones. This property of avoiding big steps in complexity to keep a reasonable trade-off between ease-of-use and expressiveness is what is called the 'gentle slope' of complexity [1, 6]. As an example, a system might offer 3 levels: on the first, the user can set parameters and make selections; on the second the user might integrate existing com-

ponents into the system; on the third level the user might extend the system by programming new components.

But adapting systems to users during usage does not necessarily require dedicated EUD activities by the user. Adaptive systems monitor their users' behavior and other contextual properties, like the current task or situation and use different approaches, notably from Artificial Intelligence, to automatically adapt themselves. One important approach to increase system adaptivity is to increase this contextual awareness by taking more contextual properties into account and to set up user models to better assess how the users' requirements relate to different contexts.

However, the distinction between system adaptability and adaptivity is not so sharp in practice. Users may want to stay in control of how systems adapt themselves and might have to supply additional information or take certain decisions to support system adaptivity. Conversely, the system might try to preselect the pertinent EUD options for a given context or choose an appropriate level of EUD complexity for the current user and task at hand, thus enhancing adaptability through adaptivity.

Apart from the system, an individual person might also be assisted by other people in its EUD activities. Such collaborative EUD activities [10] within groups of end-users can be supported by repositories for sharing EUD artifacts, as well as recommendation and awareness mechanisms for EUD-artifacts and expertise. It is one goal of current research to understand how to foster the building up of communities of end-user developers in which knowledge and artifacts can effectively be shared.

As for presenting EUD functionality to the end-user it is generally acknowledged that the adaptation interface should be unobtrusive so as not to distract user attention from the primary task. At the same time, the cognitive load of switching from using to adapting should be as low as possible. There seems to be a consensus that the adaptation functionality should be made available as an extension to the existing user interface.

Finally, the described level of system adaptability requires highly flexible software architectures. Various approaches exist, ranging from simple parameters, rules and constraints to changeable descriptions of system behavior [8] and component-based architectures [10]. A key feature of the more advanced architectures is to allow for substantial changes during run-time, i.e., without having to stop and restart or even rebuild the system.

Practical Implications

Understandably, industry players interested in EUD are looking for practical applicability and fast deployment, while not being enthusiastic about major changes to their development processes. This must be taken care of by integrating EUD with existing development practices. Nonetheless, finding the right processes and organizational structure for

EUD development and making appropriate changes will still be necessary. To this end, results from EUD research must be validated in real-world projects within the industry and the acquired experience must effectively be disseminated in adequate communities within industry and research.

This concerns the costs of providing EUD systems and, for example, whether there is a market for selling software components that can be used and adapted in EUD systems. Competition between various component vendors may cause interoperability issues when they choose to add proprietary extensions to their components to defend or extend their market share. This has not been uncommon in the software industry and as it constitutes a serious threat to a widespread success of EUD, industrial standardization efforts will be very important.

RESEARCH ON EUD

There are a fairly large number of research fields pertinent to the subject of End User Development. A selection of the most important ones is presented below, while such fields as supportive technology (e.g. development tools) or quality assurance for EUD (e.g. simulation environments, undo mechanisms) had to be left out because of space constraints. While there is not yet a stable and well-established classification of the field of EUD, first proposals have been presented in [2] and [9].

Understanding people as EUDs

As noted above, people adapting IT-systems are at the center of EUD research. Individuals carrying out EUD operations have to invest time and attention that they would normally focus on the task at hand. While being responsible for their operations they run the risk of committing errors. Accordingly, research on EUD has to provide the means for end-users to understand the consequences of their EUD operations, carry them out as safely as possible, and to exercise an appropriate level of control. Also, end-users must be motivated to pay the (cognitive) cost of performing EUD operations. To this end, EUD research has to find ways of keeping these costs at a minimum, to make operations intuitive, to provide assistance and to make the benefits transparent and assessable. Another issue to be resolved is that EUD beyond a certain level of complexity will require people to acquire additional skills beforehand, which they will have to be willing to do. Finally, doing EUD in collaboration with other people will involve new communication and work processes, as well as privacy issues, requiring appropriate solutions.

Organizational environment

EUD-systems must be properly embedded into their organizational environment to be interoperable with existing IT-systems to fully exploit the benefit of widespread EUD activities within the organization and to motivate end-users to actually carry out such activities. Conversely, EUD will have an impact on organizational structure and processes, allowing faster and more precise adaptations of IT-systems

to support, for example, the setting up of project-specific team structures and collaborative processes. Research is needed to determine how organizations must change to exploit the full potential of EUD for becoming more flexible and powerful.

Interfaces

As EUD wants to empower end-users to perform substantial modifications to IT-systems, while not hampering them in their every-day work, extending user-interfaces with EUD-functionality is as important as it is difficult. Users must be able to understand and assess the existing system and to specify and test their own EUD operations. Therefore, representational formats must be devised that are especially suitable for end-users, keeping them from making errors typical of conventional programming languages. Research is necessary on creating and evaluating domain-specific and graphical (2D and 3D) formats. Interfaces should proactively assist the users to explore and understand the systems and to create and annotate new EUD artifacts. To this end, various interesting approaches exist, like 'interactive microworlds', zoomable multi-scale interfaces, tangible user-interfaces (TUIs), AR, etc. Another requirement is that EUD functionality has to be presented as unobtrusively as possible and only when needed, so as to deviate as little of the users' attention as possible from their primary task.

Generally speaking, interfaces and representational formats play an important role in mediating communication processes between different actors (e.g. software professionals and end-users) during initial system design as well as between groups of end-users during collaborative EUD activities.

Context-awareness

As noted above, interfaces should provide users only with such an amount of EUD-functionality that is appropriate to their current context. In particular, for normal use requiring no adaptations the interfaces should generally provide no EUD-functionality at all. Moreover, systems should proactively assist their users by adapting themselves automatically if sufficient information is available, or at least generating suggestions for partial solutions for the users to choose from. In order to do this, research is needed on how systems can build up a knowledge base by monitoring their environment (e.g. user, task, place, time) and on how this context-awareness can be turned into adaptive system behavior. One promising approach is to investigate how an EUD-system might build up a history of its own use and of EUD operations it has been subject to in order to generate suggestions for future EUD operations in similar situations.

Architectures

In order to have IT-systems that are changeable at run-time while remaining maintainable and interoperable with other systems, it is quite obviously crucial to have appropriate software architectures. Loose coupling between software components through well-defined general interfaces is a

promising approach. One challenge here is to combine general interfaces which may not be very intuitive for end-users with domain-specific components which users know how to handle within their domain of expertise. Another promising approach is to add a model-layer to the architecture of IT-systems allowing for a relatively easy modification of the underlying system. A similar approach is not to build the system behavior into the actual architecture and implementation, but to separate it into a sort of meta-description which the system interprets during run-time. Finally, in order to be able to make the current system-status understandable and to let end-users assess the consequences of their operations the architectures for EUD must allow for reflexivity and inspection.

Issues and Trade-offs

Enabling end-users to substantially alter IT-systems creates a number of obvious issues concerning correctness and consistency, security and privacy. One approach to handle these issues is to let the system monitor and maintain a set of desired system properties during EUD, like integrity and consistency by, for example, allowing only safe operations. But as H. Lieberman (Massachusetts Institute of Technology) points out [4], user errors and incompleteness of information cannot be ruled out altogether, whereas users may often be able to supply missing information or correct errors if properly notified. For this reason, handling the issues above may often best be done by a cooperation of both user and system. Another issue of EUD is how to make users aware of existing EUD functions and how to make these functions easily accessible.

Finally, EUD research must find good solutions for a number of trade-offs created by empowering end-users to carry out substantial adaptations at a complexity-level no higher than needed for the task at hand. These trade-offs exist between expressiveness, freedom, and being general-purpose on the one hand and usability, learnability, control, and being domain-specific on the other.

CONCLUSION

EUD can be seen as an important contribution to create a user-friendly information society where people will be able to easily access information specific to their current context and to their cognitive and physical abilities or disabilities. People will have access to adapt IT-systems to their individual requirements and the design of IT-systems can be made more socially acceptable by collaboratively involving all actors. Apart from empowering individuals to take part in design processes, EUD can also support communities by letting them share experience on how to adapt IT-systems. In particular, communities might share EUD artifacts by way of repositories for reusable and potentially domain-specific components. These repositories will help people in choosing and assembling components appropriate for their requirements by making available the explanations, recommendations and critique of their peers.

On the economic side, EUD has the potential to enhance productivity and create a competitive advantage by empowering employees to quickly and continuously adapt IT-systems to their specific business requirements.

In EUD research much needs to be done, notably to conduct empirical research, to develop a sound theoretical basis and last but not least to establish a consistent and stable terminology. Suggestions on concrete research and development activities for EUD are currently being developed within EUD-Net and will be made available as a research agenda for the field.

ACKNOWLEDGMENTS

This article owes very much to the input of the members of EUD-Net [3]. I'd like to thank all members of EUD-Net for valuable documents and discussions. Funding of EUD-Net is provided by the European Commission.

REFERENCES

1. Blackwell, A. *User priorities for EUD: Review and research agenda*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
2. Costabile, M. F. *End-User Development - Empowering people to flexibly employ advanced information and communication technology*. Report of the 1st EUD-Net workshop at Pisa, Italy, 23./24. Sep. 2002. Available at [3].
3. EUD-Net Network of Excellence.
<http://giove.cnuce.cnr.it/eud-net.htm>
4. Lieberman, H. *Your Wish is My Command: Programming by Example for End-User Development*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
5. Mørch, A. *End-user participation in evolutionary development*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
6. Myers, B. *Making Programming Easier by Making it More Natural*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
7. Paternò, F. *Introduction to the EUD-Net EU Network of excellence*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
8. Reppenning, A. *End User Development: Who needs it?* Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
9. Sutcliffe, A. *A Comparative Framework for End User Development*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].
10. Wulf, V., and Won, M. *Supporting End-User Tailoring: Component-Based Approaches*. Presentation at first EUD-Net workshop in Pisa, Italy, 23./24. Sep. 2002. Available at [3].