Chapter 21

Future Perspectives in End-User Development

MARKUS KLANN[1], FABIO PATERNÒ[2] and VOLKER WULF[3]
[1]*Fraunhofer FIT, Schloß Birlinghoven, 53754 Sankt Augustin, Germany, markus.klann@fit.fraunhofer.de*
[2]*ISTI—CNR, Via G. Moruzzi 1, 56124 Pisa, Italy, fabio.paterno@isti.cnr.it*
[3]*University of Siegen, Hölderlinstr. 3, 57068 Siegen and Fraunhofer FIT, Schloß Birlinghoven, 53754 Sankt Augustin, Germany, volker.wulf@uni-siegen.de*

Abstract. The research field of end-user development has evolved, during recent years, to a certain degree of internal structure, problem awareness and consistency. Both academia and industry have begun to consider it an important field for research and development. In order to let EUD research contribute to the Information Societies, research and development must continue in a consolidated and well-balanced way. This chapter provides an overview of major challenges, motivates why these challenges should be addressed with considerable effort to bring about an Information Society with empowered end-users, and finally discusses how these challenges should be translated into a concrete research and development agenda for the short- and mid-term future.

Key words. tailorability, end user programming, flexibility, usability

1. Introduction

This concluding chapter presents the most important aspects for future EUD research and development, and tries to identify the principle lines along which EUD should or will most likely unfold. Being a relatively young field, EUD is yet rather diversified in terms of terminology, approaches and subject areas considered. Recently, a number of activities started within academia and industry to gain a better understanding of this field, consolidate the terminology and identify the most urging research questions.

In the center of EUD are the users who change IT-systems to better meet their requirements. As such, EUD defines a specific perspective on the practical application level of IT-systems, rather than a specific set of technological or methodological questions concerning such systems. EUD has its roots in various disciplines and fields of research, including HCI, cognitive science, requirements engineering, software engineering, artificial intelligence, CSCW, user communities, information systems, and the psychology of programming. EUD can be considered a focus in the application domain, bringing together the various more specific or technical research done in these fields into one approach of high practical relevance. Increased networking between key players in research and industry is thus a prerequisite for developing interdisciplinary solutions and marketable products.

The environments IT-systems are operating in are increasingly characterized by change and diversity. As an example, networked mobile devices and computerized

Henry Lieberman et al. (eds.), End User Development, 475–486.
© 2006 *Springer.*

artifacts will enable computing anywhere and anytime in rapidly changing and diverse contexts of use. Also, IT-systems are used by heterogeneous groups of people, having diversified requirements that depend on the users' level of expertise, current task and other factors. Systems should adapt to these changing contexts and requirements. It is the goal of EUD to empower users to carry out and control these adaptations themselves.

Flexibility at this level, necessitating highly adaptable systems as well as users willing and capable of these adaptations, would allow for what may be EUDs central goal: a co-evolution of users and IT-systems through mutual adaptation to share a common context.

In the following we will look at a number of aspects that are important for EUDs future development. In particular we will discuss the needs of users and the software industry, areas of application, important technical requirements, appropriate methods, and design criteria. Finally, we will present a roadmap for EUDs development until 2020, pointing at some of the probable milestones and discussing how the unfolding Information Society relates to this process.

2. How to Carry on With EUD

A major goal of research on EUD is to provide techniques to make IT-systems cope with changes. Quite generally, this means making them easier to develop, including setting up the initial design before use, as well as modifying them during use. In order to adapt IT-systems to their needs, individuals have to invest time and attention that they would normally focus on the task at hand, and 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 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. Possibly, incentive systems could be used to encourage people in carrying out EUD activities. Another issue to be resolved is that EUD beyond a certain level of complexity will require people to acquire voluntarily additional skills beforehand. Finally, doing EUD in collaboration with other people will involve new communication and work processes, as well as privacy issues, for which research will have to provide solutions.

What seems to be clear is that good environments for end-user development (EUD) will differ from tools conventionally used in software engineering because of the different needs of end-users and organizations running EUD-systems. Within organizations, for example, there is particular need for support of collaborative EUD activities. Nonetheless, it is of course a promising approach to investigate what methods and tools from professional software engineering can be adapted to the needs of end-user developers.

Before starting with specific research topics, let's take a look at three more general requirements EUD research should comply to.

1. Research should be driven by sound theoretical assumptions about user needs. These assumptions can be identified and refined by a variety of methods: situated (ethnographical) analysis, prototype-led development of future scenarios, task analysis, cognitive modelling, and both successful and failing case studies.
2. There is a strong consensus for the need of a sound empirical base. These may be conducted to determine people's EUD behavior (e.g. their motivation), to investigate the long-term changes of IT-systems and the contexts of use, to validate methodology, tools and representational formats, and to study the impact of EUD on conventional software engineering processes.
3. EUD research must find good solutions for a number of trade-offs created by empowering end-users to carry out substantial adaptations of IT-systems 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.

In the following, we shall present a number of areas that are important for current and future EUD research.

2.1. APPLICATION DOMAINS FOR EUD

A survey questionnaire filled out by several parties from both academia and industry indicated that office, home, and research are considered the most promising application domains for EUD (Costabile and Piccinno, 2003). Other application domains, not listed in the questionnaire, were also pointed out: education (indicated by most people), decision analysis, and the medical domain. We will take a brief look at some of these application domains.

In their homes people are likely to interact with more and more electronic devices that will become interconnected and much more flexible in the near future. This will create a mass-market where people will want to adapt systems to their specific context and requirements and where they will value personalized, adaptive, and anticipatory system behavior. Such contextually embedded or "social devices" are obviously a hot spot for EUD research. Particularly interesting is the question of how to deal with adapting shared resources through collaborative EUD techniques such as negotiation and conflict resolution.

Another interesting application domain is industrial design in manufacturing enterprises, usually supported by CAD systems, with evident potential for improving financial and quality aspects of their development process. Designers as end-users, who have deep knowledge of their specific environment and who are not professional developers, must be supplied with visual development tools to adapt their design systems to their needs.

In the scientific domain there is a lot of interest in EUD. For example in biology, experience acquired at the Pasteur Institute in Paris during several years indicates that in the field of biology applications there are many local developments in order to deal

with daily tasks, such as managing data, analyzing results, or testing scientific ideas (cf. Letondal, in this volume). Moreover, it is worth mentioning that many biologists have no or very limited programming skills, and yet feel the need of modifying the application they use to better fit their needs.

Enterprise Resource Planning (ERP) is an important sector in the software industry. Again, leading companies in the market have recently realized the importance of end-user concepts that allow various types of users of large ERP systems to modify the software in order to obtain systems more suitable for their actual needs (cf. Beringer, 2004). Over the past years, we have seen a significant change in the expectation of business applications. Traditional ERP applications gravitated very much around one single functional area and the dominant user scenarios were data entry, reporting, and ERP workflow. This simplified user model is not sufficient for modern business solutions like Customer Relationship Management, Human Capital Management, Knowledge Management, and Supplier Relationship Management. In these systems, the user is an active knowledge worker who needs communication tools, analytics, content management, and ad-hoc collaborative workflow and the capability of tailoring the system to his own needs. At the same time, the total cost of ownership (TCO) of ERP software becomes the main competitive argument. TCO can only be reduced by dramatically simplifying the customization process and by enabling business experts and end-users to modify the software themselves without the need of hiring IT consultants or IT-administrators (cf. Beringer, 2004; Wulf and Jarke, 2004). Already today, Enterprise Portals offer the personalization or creation of custom-made web pages and reports. Last but not least, companies such as SAP see a shift into a service-based architecture of business applications that may result in a new application development paradigm in which traditional coding is replaced by orchestration of existing enterprise services. Service composition including generation of user-interfaces may become an activity of business experts using simplified development environments with pre-packaged semantics. Considering these changes in the user model of ERP software, such companies see an increasing relevance of EUD-functionality in their products.

Another application domain is the one related to systems supporting data intensive businesses like telecommunication, e-government or banking. Computer applications become integrated in infrastructures connecting different work practices within and across organizational borders. The flexibility of such infrastructures is of strategic importance when developing new services. Often the need to redevelop part of the computer support to accommodate business or organizational development prohibits the entire development. Thus, tailorable systems and domain specific EUD provide a competitive advantage.

2.2. ARCHITECTURES AND GENERAL EUD FUNCTIONALITY

The recurrent theme of EUD is that end-users should be empowered to make substantial adaptations to IT-systems easily. The answer to the apparent contradiction is that

there should be means of adaptation that are comparable in complexity to the problem at hand. This means that end-users will generally not program in a conventional programming language but will use higher-level means of adaptation that can do the job but are otherwise as simple as possible. The thing to observe here is that ultimately the modified system must be executed regardless of the means by which adaptations were carried out. Hence, allowing for adaptations from an ideally gentle slope of adaptation complexity to consistently and safely change a system's run-time behavior requires an appropriately specialized system architecture. For EUD to become a widespread success such architectures must become generally available in the form of frameworks to substantially facilitate the development of EUD-enabled systems.

There are a number of additional requirements that EUD architectures must provide for others than the adaptations as such. One is that EUD-systems must remain maintainable and interoperable in the face of run-time changes. Another is that EUD-systems should allow for reflexivity and inspection to make users understand the current system status and enable them to assess the consequences of their adaptation activities. Finally, knowledge on the relation between adaptation operations and system properties should be used as much as possible to analyze adaptation operations and restrict those that are insecure or otherwise undesirable.

One promising approach is to add a model-layer to the architecture of IT-systems allowing for relatively easy modifications of the underlying system. A similar approach is not to build-in 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.

Component-based EUD systems are another promising approach, allowing for a gentle slope of complexity by way of successive decomposition of components in case the overall component structure has been properly designed (cf. Won et al., in this volume). One challenge here is to find "patterns of decomposition" that facilitate finding appropriate component structures when designing new applications (cf. Stevens et al., in this volume). Another challenge is to combine general component interfaces, which may not be very intuitive to end-users, with domain-specific components, which users know how to handle within their domain of expertise.

The architectural challenge for EUD-enabled systems becomes particularly apparent in the vision of ubiquitous computing. Here, an array of distributed and interconnected devices is supposed to create and provide in an ad-hoc way a consistent, personalized, and context-sensitive service to its users. The context of use can be considered as the combination of the user (with his background, interests, tasks, ...), the surrounding environment, and the devices at hand. While adaptivity and self-configuration can certainly carry a long way, user-driven adaptability remains crucial so that users can fine-tune the system to their work practices, business goals, etc. These adaptation activities will also enhance the users' competence and support their understanding of the system. An example in this direction is the TERESA environment (Mori et al., 2003) that provides support for the design and development of nomadic applications, which can be accessed through different types of interaction platforms.

2.3. USER INTERFACES

As EUD wants to empower end-users to perform substantial modifications to IT-systems, while at the same time 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 systems and to specify and test their own EUD operations. In order to enable end-users to go from the running system to a changeable representation and back again, EUD-environments must support both reverse and forward engineering processes. Also, 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 user 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), augmented reality (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, like software professionals and end-users during initial system design as well as between groups of end-users during cooperative EUD activities.

2.4. COOPERATIVE ACTIVITIES AND ORGANIZATIONAL SETTINGS

Cooperation is an essential part of EUD. Future research will have to investigate effective means for communities of end-users to communicate about their adaptation problems, negotiate solutions, and share both their EUD expertise and reusable EUD artifacts. Cooperation on EUD activities is largely a social phenomenon and research will have to understand how an appropriate EUD culture can be fostered by incentive mechanisms, trust building, and community awareness.

As with any cooperation the organizational context must be taken into account when developing and deploying EUD systems. They must be properly embedded into their organizational environment to be interoperable with existing IT-systems, and thus 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.

One of the difficulties associated with EUD-software used within organizations is that administration and support of the software has to deal with a system that is continuously

changing through its users' adaptations. For such changing EUD-systems new ways of professional IT-services must be developed that go beyond the "If you change it, we won't support it!" mind-set while still being manageable for the service providers. One first step is to restrict the number of potentially hazardous end-user adaptations by defining and enforcing certain desired system properties, such as consistency.

2.5. THE ROLE OF ADAPTIVITY

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, interfaces should rather hide EUD-functionality and may just offer a specific access point, for instance via the context menu. Moreover, systems may proactively assist their users by adapting themselves automatically if sufficient information is available, or at least generate 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, etc.) and on how this context-awareness can be turned into adaptive system behavior (cf. Dey and Sohn, 2003). One promising approach is to investigate how an EUD-system might build-up a history of its own use and of the EUD operations it has been subjected to and to generate suggestions for future EUD operations in similar situations.

2.6. QUALITY ASSURANCE

Giving end-users the means to substantially alter IT-systems goes with the risk of having them produce erroneous adaptations. While it is possible to reduce this risk by properly designing the means for EUD operations, errors cannot be ruled out altogether. But it is possible to assist the end-users in detecting and correcting errors, by continuously monitoring and checking different system properties like coherence, consistency, and correctness, alerting the user in case an error has been detected and possibly making suggestions on how to correct it (cf. Burnett et al., in this volume; Won, 2003). To reduce the damage caused by errors, EUD-systems should provide some sort of "simulation environment" where users can test their modifications without risk (cf. Wulf, 2000). Moreover, systems should provide an undo-mechanism, so that users can easily and confidently reverse their operations. Finally, a more social mechanism of making the reliability of already existing EUD artifacts assessable to the users is to annotate the artifacts with information about their creator(s) and about their history of use (e.g. uses, malfunctions, and ratings) (cf. Costabile et al., 2002; Wulf, 1999). Research on these topics is needed to provide what might be called "quality assurance" for EUD.

2.7. INDUSTRIAL PERSPECTIVES

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. As explained above, this can be done 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. An example of a promising area for EUD applications of industrial interest is that of services for mobile devices. In the near future many people will access interactive software services through their mobile phones or PDAs. It will become important that such services will be modifiable and adaptable. Users should be enabled to carry out certain of these adaptations even by means of their mobile devices overcoming the limits of the limited input and output interfaces.

One specific field of industrial interest is to use results from EUD to foster the understanding of existing IT-systems and support the integration of new applications by generating comprehensible representations at an appropriate level of complexity.

Generally speaking, the industry will have to find out how the promising potential of EUD translates into concrete market opportunities for profitable products and services. 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 such EUD systems. Competition between various component vendors may cause interoperability issues, when, for example, one vendor will add proprietary extensions to his components to defend or extend his 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 are crucial.

3. An EUD-Roadmap to an Information Society With Empowered End-Users

In order to outline a roadmap for research in EUD, as illustrated in Figure 21.1, we suggest here to focus on three intertwined lines of research: software architectures, interfaces, and support for collaboration.

Starting with the current state of EUD, we discuss what research activities could reasonably be carried out until about 2007, what status would then be reached, and how research could continue until 2012. As for the predictions for 2012 and 2020, they are obviously rather general in nature and do not yet include concrete recommendations for research activities. Rather, they deal with the impact of EUD on the Information Society, state possible applications and societal aspects and make guesses on what EUD goals might be achieved at the corresponding time.

With regard to software architectures a number of different approaches exist in research (e.g. agent-based, component-based, and rule-based). However, empirical knowledge on suitability in real-world settings is still insufficient. A major challenge is to refine existing approaches and to develop new architectures permitting both systems and users to evolve. The combination with conventional architectures and the adaptation

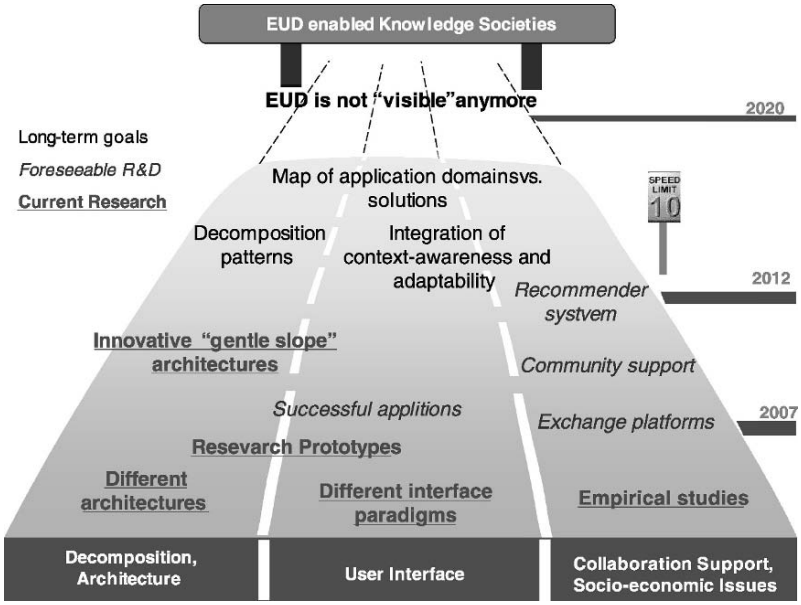

Figure 21.1. The future of end-user development.

of the respective development processes have to be investigated. Moreover, one has to look into the suitability of different EUD-architectures to support run-time changes with a gentle slope of complexity. Case studies need to show the suitability of these frameworks for different application domains.

With regard to interfaces, research has been carried out on various interface techniques, for example, Augmented Reality and Tangible User Interfaces. One of the main goals of current research in user interfaces is to obtain natural interaction, where users can interact with their applications in a way similar to how they communicate with other humans. This paradigm can be successfully applied to EUD. Natural development (Berti et al., 2004) implies that people should be able to work through familiar and immediately understandable representations that allow them to easily express relevant concepts, and thereby create or modify applications. On the other hand, since a software artifact needs to be precisely specified in order to be implemented, there will still be the need for environments supporting transformations from intuitive and familiar representations into precise—but more difficult to develop—-escriptions. Examples of informal input for more structured representations are sketches on board (Landay and Myers, 2001). For example, non-programmer users feel comfortable with sketch-based systems that allow them to concentrate on concepts by exploiting natural interactions, instead of being distracted by cumbersome low-level details required by rigid symbolisms. Such systems are generally able to recognize graphical elements and convert them into formats that can be edited and analyzed by other software tools.

New UI-techniques, for example, combining adaptability and adaptive context-sensitive system behavior (Klann et al., 2003), need to be developed. Moreover, interfaces need to be developed that make EUD-functionality available with a gentle slope of complexity.

Collaborative aspects have been taken up in research as a key element of EUD (e.g. gardening-metaphor). However, empirical knowledge on collaborative EUD is still insufficient and implementations of collaborative EUD-functionality are only in their beginnings. Therefore, concepts for collaborative EUD have to be developed. Conventions and standards for describing EUD artifacts have to be worked out to make them exchangeable, for example, with regard to quality, recommendations, and purpose. Based on such conventions, EUD-artifacts can be described and placed into repositories for sharing. Software agents should be able to recommend suitable artifacts available from such repositories.

Looking toward 2012, we believe that architectural frameworks, decomposition techniques, patterns, interfaces, and tools for collaboration support can exist in a consolidated and integrated manner. End-users will be supported by tools for exploring, testing, and assessing while carrying out EUD activities. Cases of best practices will have been documented to explain EUD as an activity that is embedded in social networks. Concepts to acquire EUD-skills will be integrated into educational curricula. EUD will become an important aspect of applications in most domains: education, scientific research (e.g. bioinformatics), business (CAD, ERP, GIS), and domestic domains.

Toward 2020, substantial adaptability has become a property of all newly developed software systems. Adaptable software-systems have penetrated into all domains, for example, business, leisure, home, culture. Most people have skills in EUD. EUD has become an important activity for most jobs and for the majority of people. A high level of adaptivity in all devices is a big part of what is called "Ambient Intelligence." EUD has gained central importance in the application of information technology and the use of EUD techniques has become a common practice for users of all ages and professional background. EUD has become an integral aspect of their cultural practices and appropriation of IT.

4. 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 physiological abilities or disabilities. People will have access to adapt IT-systems to their individual requirements, and if all actors will be involved the design of IT-systems will find a higher common acceptance. Providing end-users with effective development environments is therefore one strategic goal for the European Information Society.

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

As Figure 21.1 shows, the road to achieving an EUD enabled Information Society is still long, even if prototypes, mainly research ones, have already appeared. Particularly challenging areas that will be addressed in the near future are novel interface techniques for EUD, integration of context-awareness and adaptability, effective sharing of EUD artifacts through repositories with recommendation support, and decomposition guidelines for flexible component-based systems. Quite generally there is further need for theoretical research on the foundations of EUD as well as applied research to gain experience in and develop methods for EUDs various application domains.

The ultimate goal is to provide end-users with non-intrusive, "invisible" support for their developments and thus empower them to use their domain specific know-how to shape the IT tools to better support them in their daily practices. As such, EUD will become an integrative part of our cultural practices and should be considered part of a comprehensive computer literacy.

Acknowledgments

Our understanding of the research field of EUD and the problems associated with its industrial uptake, as expressed in this chapter, was substantially deepened by the valuable discussions we had within the context of the EUD-Net project. At several events organized by EUD-Net a number of participants from outside the project made further important contributions to this discussion. We thank everybody from academia and industry who shared their approaches and experiences with us.

References

Arondi, S., Baroni, P. et al. (2002). Supporting co-evolution of users and systems by the recognition of Interaction Patterns. *AVI 2002*, Trento, Italy.

Beringer, J. (2004). Reducing expertise tension, in communications of the ACM. **47**(9), 39–40.

Berti, S., Paternò, F. and Santoro C. (2004). Natural development of ubiquitous interfaces. *Communications of the ACM* (September), 63–64, ACM Press.

Boehm, B.W., Abts, C. et al. (2000). *Software Cost Estimation with COCOMO II*. Upper Saddle River, NJ: Prentice Hall PTR.

Burnett, M., Rothermel, G. and Cook, C. (in this volume). An Integrated Software Engineering Approach for End-User Programmers.

Cockburn, A. (2002). Agile Software Development, Addison Wesley.

Costabile, M.F., Fogli, D. et al. (2002). Computer environments for improving end-user accessibility. *ERCIM Workshop "User Interfaces For All"*, Paris.

Costabile, M.F., Fogli, D. et al. (2003). Building environments for end-user development and tailoring. *IEEE Symposia on Human Centric Computing Languages and Environmnets*, Aukland.

Costabile, M.F. and Piccinno A. (2003). *Analysis of EUD Survey Questionnaire*, D4.2, EUD-Net.

Dey, A.K. and Sohn, T. (2003). Supporting end user programming of context-aware applications. *Workshop on End-User Development at CHI 2003*, Fort Lauderdale, Florida.

Henderson, A. and Henderson, K.M. (1991). There's No Place Like Home. *Continuing Design in Use. Design at Work*, Lawrence Erlbaum Assoc., 219–240.

Klann, M., Eisenhauer, M. et al. (2003). Shared initiative: Cross-fertilisation between system adaptivity and adaptability. *UAHCII 2003*, Crete.

Landay, J. and Myers, B. (2001). Sketching interfaces: Toward more human interface design. *IEEE Computer* **34**(3), 56–64.

Lehman, M. (1980). Programs, Life Cycles, and Laws of Software Evolution. *IEEE 68*.

Letondal, C. (2001). Programmation et interaction, Phd-thesis, Orsay, Université de Paris XI.

Letondal, C. (in this volume). Participatory Programming: Developping Programmable Bioinformatics Tools for End-Users in context.

Liebermann, H. (2001). *Your Wish is My Command: Programming by Example*. Morgan Kaufmann, San Francisco.

Majhew, D.J. (1992). *Principles and Guideline in Software User Interface Design*, Prentice Hall.

Mehandjiev, N. and Bottaci, L. (1995). End user development with visual programming and object orientation. *1st International Workshop on End User Development at CaiSE'95*, Juvaskyla, Finland.

Mehandjiev, N. and Bottaci, L. (1996). User-enhanceability for organizational information systems through visual programming. *Advanced Information Systems Engineering: 8th International Conference*, CAiSE'96, Springer-Verlag.

Mørch, A.I. and Mehandjiev, N.D. (2000). Tailoring as collaboration: The mediating role of multiple representations and application units. *Computer Supported Cooperative Work* **9**(1), 75–100.

Mori, G., Paternò, F. and Santoro, C. (2003). Tool support for designing nomadic applications. In: *Proceedings ACM IUI'03*, Miami, pp. 141–148, ACM Press.

Oppermann, R. and Simm, H. (1994). Adaptability: User-initiated individualization. *Adaptive User Support—Ergonomic Design of Manually and Automatically Adaptable Software*. R. Oppermann. Hillsdale, New Jersey, Lawrence Erlbaum Ass.

Orlikowski, W.J. and Hofman, J.D. (Pipek 1997). An improvisational model for change man-agement: The case of groupware technologies. *Sloan Management Review*, pp. 11–21.

Paternò, F. (1999). *Model-Based Design and Evaluation of Interactive Applications*, Springer Verlag.

Repenning, A., Ioannidou, A. et al. (2000). AgentSheets: End-user programmable simulations. *Journal of Artificial Societies and Social Simulation* **3**(3).

Stevens, G., Quaißer, G. and Klann M. (in this volume). Breaking it Up—An Industrial Case Study of Component-based Tailorable Software Design.

Sutcliffe, A., Lee, D. et al. (2003). Contributions, costs and prospects for end-user development. *HCI International 2003*, Crete, Greece, Lawrence Erlbaum Associates.

Won, M. (2003). Supporting end-user development of component-based software by checking semantic integrity. *ASERC Workshop on Software Testing*, Banff, Canada.

Won, M., Stiemerling, O. and Wulf, V. (in this volume). Component-based Approaches to Tailorable Systems.

Wulf, V. (1999). "Let's see your Search-Tool!"—Collaborative Use of Tailored Artifacts in Groupware. In: *Proceedings of GROUP '99*, ACM-Press, New York, 1999, pp. 50–60.

Wulf, V. (2000). Exploration environments: Supporting users to learn groupware functions. *Interacting with Computers* **13**(2), 265–299.

Wulf, V. and Jarke, M. (2004). The economics of end user development. *Communications of the ACM* **47**(9), 41–42.