Groupware Construction with the Oregon Software Development Process

Till Schümmer
Computer Science Department,
FernUniversität in Hagen, Germany
till.schuemmer@fernuni-hagen.de

Abstract. This position paper describes how appropriation of groupware systems takes place in the Oregon Software Development Process. It calls for a special focus on high-level groupware patterns as communicative and educational means for end-users and developers.

1 Introduction

The development of groupware applications is still a difficult task. One main reason is that both developers and end-users are not aware of possible solutions for supporting group interaction. A second aspect is that group interaction involves many users, which makes the definition of requirements difficult. From that perspective, end-user involvement is one of the most important but much too often neglected issues in groupware development.

Developers and end-users need to be supported in the requirements elicitation and the design of tools that help to satisfy these requirements.

I argue to approach this need from three different perspectives: (1) the abstract view on development that is not bound to the specific nature of the developed artifact, (2) the software development perspective that focuses on processes and tools for the development of software, and (3) the groupware development perspective, which brings together software development aspects with social

aspects and thus addresses the task of groupware development from a socio-technical view.

The basis for my abstract view on design is the holistic view, as it is currently propagated by many designers, especially by Christopher Alexander (Alexander 2003). In his view, a holistic approach to design has to be combined with an evolutionary process and focus on end-user education in order to empower the enduser to play an active role in the development process. Together with Heidegger (Heidegger 1927), Alexander puts special attention on the situatedness of design (Alexander, Silverstein, Angel, Ishikawa & Abrams 1980). Users should reflect on their activities whenever their flow of action is disrupted. This situated reflection (as it was also propagated by (Schön 1983)) provides the best access to the requirements and supports solutions that meet the requirements.

Situatedness requires that the end-user is heavily involved in the design process. To reach this level of control, they have to be educated regarding possible good practices – the patterns – for reshaping their environment. In *A Timeless Way of Building*, Alexander defined a pattern as a morphological law that explains how to design an artifact in order to solve a problem in a specific context (Alexander 1979). The goal of describing best practices with patterns is that end-users are empowered to shape their environment in a professional manner.

These core requirements relate to current software development processes. Especially evolutionary processes (Malotaux 2001), agile methods (Boehm & Turner 2004), and participatory design approaches (Muller & Kuhn 1993) propose an interaction between developers and end-users that gives the control over the built artifact back to the end-users.

Relevant groupware development processes that focus on end-user involvement include the extended eXtreme Programming process (Rittenbruch, McEwan, Ward, Mansfield & Bartenstein. 2002) that focuses on involving the user community in planning and development (instead of just a single customer representative in XP) or an iterative process based on the STEPS process model (Floyd 1993) that puts special attention tailoring during system use (Wulf & Rohde 1995). The latter demands that the end-user adapts the system in order to meet requirements that evolve from reflection in action. Other groupware processes, like SER (Fischer, Grudin, McCall, Ostwald, Redmiles, Reeves & Shipman 2001) put a special focus on sharing of design knowledge. But still they do not help to shape the knowledge in a way that it can be easily used by end-users.

In summary, the field still lacks a groupware development process that supports end-users in a way so that they can learn, plan, implement, modify, and share appropriations based on best practices.

The Oregon Software Development Process presented in this paper tries to fill this gap. It fosters end-user participation, evolutionary growth, and reuse and

exchange of design knowledge. Patterns play an important role in the whole process since they are the means for communicating design knowledge between users, developers, and between users and developers. Expert Involvement 3 8 2 6 11 5 7 9 User Involvement 4 1 Analysis of forces Planning Conflicting forces Design Pattern driven tailoring design - high level patterns - Pattern driven groupware design - low level patterns - Scenarios Implementation Patterns & mockups Pattern driven groupware tailoring Test & Usage Usage with diagnosis & reflection - health map - functional tests conceptional iteration development iteration tailoring iteration Initial forces Groupware development Discussion 10 12

While the complete process has been described before (Schümmer & Slagter 2004), this paper discusses how appropriation work of end-users is supported by the process. After giving a short introduction to the core practices of OSDP, I will focus on tailoring iterations that have the goal of appropriating the groupware system during use.

2 The Oregon Software Development Process

The Oregon Software Development Process (OSDP) intends to foster end-user participation, pattern-oriented transfer of design knowledge, piecemeal growth of the system under development in the form of short iterations, and frequent diagnosis or reflection that lead to an improved application.

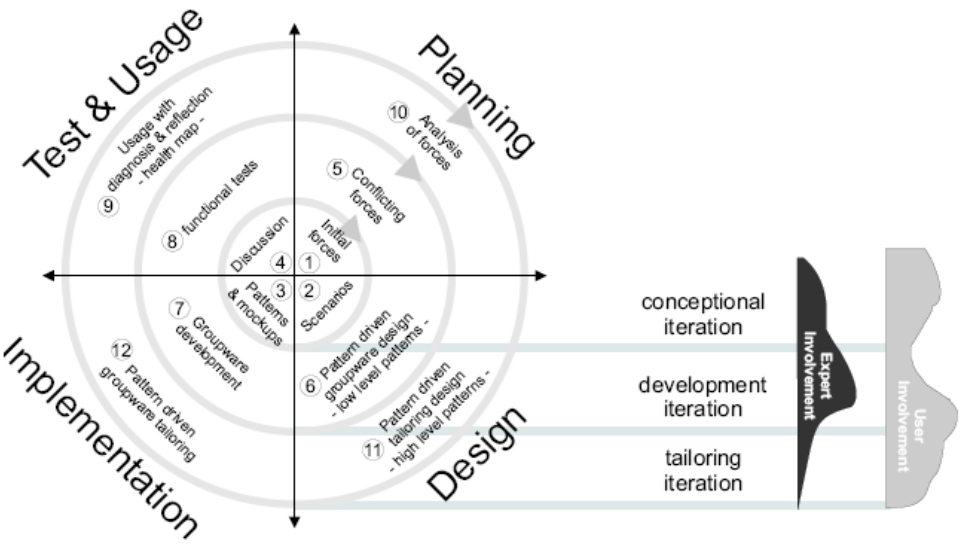

Figure 1: The Oregon Software Development Process.

OSDP structures the development of the application in three kinds of iterations: conceptual iterations, development iterations, and tailoring iterations. Figure 1 shows these iterations denoted by the three circles.

– In conceptual iterations users and developers collaborate in scenario interest groups (SIGs) in order to collect and refine scenarios of system use. They make use of groupware patterns to inform their mapping of social processes to a groupware setting.

– Development iterations focus on the implementation of the scenarios from conceptual iterations. Users and developers collaborate to create task descriptions and to relate these descriptions to groupware patterns. These descriptions are collected and estimated regarding their costs and benefits. The development team continuously focuses on the most important tasks in order to implement the most critical aspects first. Important aspects of the development iteration include that the list of tasks – the backlog – is made public in the user community and that the task wishes of different users or SIGs are ordered and merged.

– In tailoring iterations users reflect on their groupware use and fix conflicting forces by adapting the groupware system. Pattern scouts encourage the users to share their adaptations with other users. If the adaptation matured it is formulated as a pattern and added to the community's pattern language.

While the first two iterations mainly take place before the system is in use, the tailoring iteration describes how the system is appropriated during its use. The following section will thus have a closer look at this kind of iterations.

3 Tailoring Iterations Close Up

In the *tailoring iteration* end-users use the application for the desired purpose. While using the system, end-users with pattern-based design knowledge are encouraged to reflect on their activities whenever they encounter a *breakdown* (9 in fig. 1). A breakdown leads to an entry in the groupware's health map (in the simple case, a note that a specific group need could not be satisfied with the groupware system). In cases where the user does not detect this breakdown (i.e., if the user feels uncomfortable but thinks that this feeling cannot be changed), an evaluation user (as proposed by cf. (Rittenbruch et al. 2002)) can expose the breakdown, discuss it with the user, and initiate a reflection process together with the user.

Users then take a closer look at the detected shortcoming. First, they *analyze the forces* that are in conflict (10). High-level groupware patterns help in this process by describing frequently occurring issues, the various forces, and a proven solution in a way that is appropriate for tailoring end-users.

The solution provided by the high level patterns informs the successive *groupware tailoring design* (11) and the execution of the *tailoring* (12). Tailoring actions can take place at different levels. In content level tailoring, the users change the artifacts that are managed by the groupware system in order to solve the problem. At this level, the pattern assists the user in using a tool. At the functional level, users appropriate the functionality provided by the tool. They activate needed functions and deactivate functions that are in the way. At the component level, the users perform more extensive tailoring actions: they compose functional groupware components in order to create new configurations of applications that support the team in an unanticipated way.

To support tailoring at a group level, a *pattern scout* looks for solutions that work well. As the evaluation user, the pattern scout observes users in the system with the goal of finding recurring successful system use. The found best practice is then discussed with the users and documented in the pattern format. Such new best practices then find their way in the pattern catalogue. Note that these patterns are in most cases very domain specific (e.g. patterns for supporting customer relationship management in the context of a support system). These patterns will, however, be used most frequently in the user community since they are appropriated for the interaction that typically takes place in the community.

4 Experiences

The OSDP has been applied in the CURE project in which a collaborative learning environment for the Distance University of Hagen was developed and installed. The project showed that users and developers ware able to follow the proposed steps and that the patterns play an important role in the communication between all stakeholders. One could observe that users were able to play an active role in all phases of the process. They shaped their environment and developed new best practices (e.g., practices to support literature research). First of these practices were captured as patterns.

The opportunity to tailor motivated users to reflect on their activities. Users created new domain-specific patterns and provided implementations using the tailoring mechanisms of CURE. Some users referred to patterns to support their tailoring, others based their tailoring operations on intuition. This may be one reason why many users reported that tailoring was still difficult.

Propagating the collection of domain-specific patterns to the users is thus still a challenging task in order to educate the end-users and to support better tailoring actions. In cases where patterns were used the forces were also made explicit and discussed between the tailoring users. In other cases, the forces did not play an explicit role.

5 Conclusions

This paper presented parts of the Oregon Software Development Process that focus on the appropriation of groupware systems at runtime. The tailoring iterations of OSDP structure end-user activities in order to support reflection on system use, capturing of best practices, and sharing of expert-user's appropriations.

Most important tools in the OSDP are patterns that help to capture design knowledge in a way that is easy to understand for end-users as well as groupware developers. These patterns evolve during system use, steer the tailoring of the system, and capture evolving best practices of system use.

However, the OSDP is no silver bullet. Patterns with a high level of abstraction can often be implemented by tailoring the application, but low level patterns still require development and design expertise by the involved software developers.

6 References

Alexander, C. (1979), *The timeless way of building*, Oxford University Press, New York.

Alexander, C. (2003), *The phenomenom of life*, Vol. 1 of *The nature of order*, Center for Environmental Structure, Berkeley, California, USA.

Alexander, C., Silverstein, M., Angel, S., Ishikawa, S. & Abrams, D. (1980), *The Oregon Experiment*, Oxford University Press, New York.

Boehm, B. & Turner, R. (2004), *Balancing Agility and Discipline – A Guide for the Perplexed*, Addison Wesley, Boston, MA.

Fischer, G., Grudin, J., McCall, R., Ostwald, J., Redmiles, D., Reeves, B. & Shipman, F. (2001), Seeding, evolutionary growth and reseeding: The incremental development of collaborative design environments, *in* G. Olson, T. Malone & J. Smith, eds, 'Coordination Theory and Collaboration Technology', Lawrence Erlbaum Associates, pp. 447–472. *http://www.ics.uci.edu/ redmiles/publications/B002-FGMcC01.pdf

Floyd, C. (1993), 'Steps – a methodical approach to pd', *Commun. ACM* **36**(6), 83.

Heidegger, M. (1927), *Sein und Zeit*, 17 (1993) edn, Niemeyer, T¨ubingen.

Malotaux, N. (2001), Evolutionary development methods, *in* 'Proceedings of PROGRESS 2001', Technology Foundation (STW), Utrecht, the Netherlands. *http://www.stw.nl/progress2001/proc2001/malotaux.pdf

Muller, M. J. & Kuhn, S. (1993), 'Participatory design', *Communications of the ACM* **36**(6), 24–28.

Rittenbruch, M., McEwan, G., Ward, N., Mansfield, T. & Bartenstein., D. (2002), Extreme participation -moving extreme programming towards participatory design., *in* T. Binder, J. Gregory, &

I. Wagner, eds, 'Participation and Design: Inquiring Into the Poltics, Contexts and Practices of Collaborative Design Work – PDC 2002 Proceedings of the Participatory Design Conference', Malmo, Sweden.

Schön, D. A. (1983), *The Reflective Practictioner: How Professionals Think in Action*, Basic Books, New York.

Schümmer, T. & Slagter, R. (2004), The oregon software development process, *in* 'Proceedings of XP2004'.

Wulf, V. & Rohde, M. (1995), Towards an integrated organization and technology development, *in* 'DIS '95: Proceedings of the conference on Designing interactive systems', ACM Press, pp. 55–64.