Learning Participatory Design by Participatory Design

Morten Moth Iversen, Jan Ernfred Madsen & Jakob Svaneborg Vesterstrøm
Institute of Computer Science
Aarhus University
Ny Munkegade 116
DK-8000 Aarhus N.
Denmark
{moth,ernfred,jve}@daimi.au.dk

ABSTRACT
During the previous year we have experienced a noteworthy way of learning the techniques of Participatory Design (PD). We will demonstrate how the making of a study group with a supervisor can bring PD-learning from theory to practice during the University-period. The focus has been on learning the PD-techniques by engaging us in an actual developing process, where our prior PD-knowledge was utilized. We will show how this was done, remembering that neither the user nor developer had any practical experience with the use of PD.

Keywords
Participatory Design, participatory analysis, prototyping, study group, commitment, supervisor and hands on experience.

INTRODUCTION
In the process of designing software the system developer has to transcend the domain of the application – but mere transcendence is not enough, choices must have basis in tradition to reach qualitative and durable decisions. As Mogensen [8] puts it: 'One can focus on tradition or transcendence, but the question is always that of tradition *and* transcendence' - both must be part of practice since tradition in a sense represents accumulated knowledge and as such must be regarded as the only sensible point of departure. The Participatory Design tradition is widely recognized as a sound way of engaging a dialog when exploring and inventing new concepts. Approaches like future workshop [5] and Cooperative Analysis [9 + 10] are used to enhance the creativity in these sessions.

Since the traditional practice of PD is 'a collaborative approach to design, not a rigid set of design methods' [4], obviously one cannot use the techniques as a simple checklist in the development process. The very nature of PD indicates that more effort has to be put into it. Reported PD projects have typically been conducted by people well founded in the PD tradition, expanding the horizon of the concepts used within the field.

In *PDC 2000 Proceedings of the Participatory Design Conference*. T. Cherkasky, J. Greenbaum, P. Mambrey, J. K. Pors (Eds.) New York, NY, USA, 28 November - 1 December 2000. CPSR, P.O. Box 717, Palo Alto, CA 94302 cpsr@cpsr.org ISBN 0–9667818–1–3

For the entire PD society this research has invaluable impact and well established PD users can adapt ideas from the theories proposed straight away. However, users with limited practical experience are not as well situated. In the development process 'mediating artefacts and activities must be flexibly adapted to the demands of the situation' [1] This fact requires a display of initiative and guidance towards the end user, from the developer. These personal qualities cannot be acquired by solely theoretic studies; one has to learn by experience. Furthermore as Morten Kyng recognizes [2] the 'crucial elements of what makes particular artifacts effective in specific situations of use are not explicitly known to us [the developer] and cannot be deduced through prior, detached analysis'. This stresses the necessity of hands on experience within the development process and obviously, when engaging in PD this way, one must have practiced the discipline beforehand. In other words hands on experience is as fundamental in the process of learning the PD principles as it is in PD itself.

Attending Computer Science at the University of Aarhus, Denmark we have read a lot of papers and books about PD. As explained there is however the one culprit that only by experience one learns the entire aspect of PD. So the question goes:

"How do we gain practical experience with PD during the University-period ?"

The answers to the question is manifold, but we are under the impression that we have a well-suited answer: Using PD techniques on a *real* project in a considered setting, leaving room for creativity while under guiding supervision – *Learning PD by PD*

Our first experience with PD was acquired through a system development course called System Development (SD) held by The Institute of Computer Science, Aarhus University. The intent with the course was an enhancement of PD skills

of the participants, by developing a computer system, using the various PD-techniques; mainly focusing on the ideas of Object Oriented Analysis and Design (OOA&D) presented by Mathiassen et. al [6]. We read about the techniques, and engaged ourselves in an actual project, but due to the time limit only two workshops were held - so in this context we learned a lot of PD-theory but only little PD-practice. That factor, and bearing in mind that hardly any University-courses are practically orientated, gave us the inspiration to create our own course with a supervisor. We had the following requirements for the course:

- The system were now to be completed
- Workshops were to be held approximately every 2 week.
- The development strategies were to be discussed with our supervisor approximately every 1 week.

The idea was to gain more experience with PD and system development with the guidance of our supervisor. The intend was not to directly invent any new PD-techniques, but rather to discover the already described techniques to their full potential – and in effect achieve better understanding by use of our acquired PD-knowledge in practice.

In 'Bringing Design to Software' Mitchell Kapor [12] touches the subject on how to create a meaningful educational environment for 'software designers'. The present report proceed reflecting upon the course we created describing our ideas, as well as identifying critical points that should be considered when passing on the PD tradition to computer science students. In our course we have mainly been part of two activities: The activity of learning practical PD and the activity of implementing a concrete software system. In this report the main focus will be describing and discussing the former in contrast to the majority of articles on PD working within the PD tradition. But even though the focus is on learning PD, it is important to note that the system development process is the empirical foundation of our understanding of the field. So when reading this report it is important to have in mind that we describe and discuss how we tried to form a setting from which we could learn the richness of PD, not the actual development of the system. Sarah Kuhn suggests how the use of PD-techniques can move software design towards 'Human Centered-Design' [13]. This approach focuses on the social and political aspect of system development, whereas we mainly focused on enhancing the quality of software design. In both approaches the following objective is present, though: ease work practice and increase control with work for the end user, by appliance of the PD-techniques.

PROJECT SYNOPSIS

The program we started to develop during the SD course was an administration system to a tennis tournament, held by Aarhus Lawn Tennis Society (ALTS) - a Tennis Club in our hometown. The intention with the program was to make the getting through of a tournament effective, including registration of tennis-participants, printing and the whole management during the tournament. In this first phase in our project, we didn't manage to finish the system, but we came to understand most of the application domain.

In the second phase, after creating the study group, we intensified the work with the system, so that it now included web-facilities, database-integration and etc. The intensification also consisted of user-involvement that gave us invaluable knowledge, and it is primarily this second phase the report is based upon. During both phases we primarily worked with two users from ALTS: one, which is a committee member and the other, being the tournament manager. We set the deadline of the project to summer 2000 where the Tennis Club is holding the tournament.

THE WAY WE WORKED

Since our practical experience with PD was limited, we had to come up with a workprocess allowing us to utilize our theoretic knowledge to the full extent. We wanted to be able to experiment with the different approaches discussed in the SD course, yet avoiding the possible pitfall of following an experimental idea that we because of our scarcity of experience would fail to identify as a dead end.

Bearing in mind that we already had a fundamental understanding of the application domain. We agreed on a work-method conveyed in the following figure of information flow during the project:

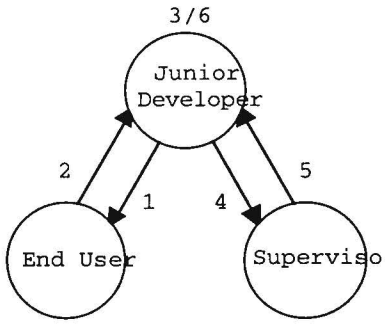

Figure 1: The information flow

The steps 1 to 6 would constitute a circle loop repeated during the semester, each cycle having a lifetime of approximately a fortnight.

1. After a developer meeting agreeing on program functionality and how to demonstrate ideas, the suggestions would be presented to the end user.
2. Suggestions would be discussed at the user meeting, giving user feedback for later analysis - the cornerstone of PD.
3. The information gained in step two was discussed internally in the group resulting in decisions on implementations – and usually new ideas - but at the same time making a point of discovering possible weaknesses/ uncertainties in respect to these choices.

4. Current progress and implementation status was presented to the supervisor, typically as an introduction to further discussion on the points identified in step 3 as being worthy of extra exploration.
5. The meeting with the supervisor often resulted in new enlightenment's and the discovery of entirely new aspects of the matter at hand.
6. New insights would be discussed and plans for the future development were established.

So in effect we were using a traditional iterative approach between analysis, design, and implementation – adding to it an extra dimension of consultation with the supervisor. Between every workshop/prototype-session the normal internal developer discussion was extended with such a follow-up. In this forum partly the discussion of user feedback was continued and partly the course of events considering the study group as such was looked at.

If one takes a step back and look at this setup, it is not obviously a sound constellation seen from the Junior-developer's point of view. On one hand we should meet the demand from the end user who wants a program up running today rather than tomorrow, on the other hand however, we were to satisfy the supervisors expectations in the line of following an in depth theoretic approach. The latter being dictated by the guidelines laid down in mutual agreement from the beginning of the course. So, in a sense, we were between the devil and the deep blue sea having almost contradicting requirements on either side of us. Further more we had to find room for our selves in this context - without us as the acting part the fundamental idea of the project would be lost.

Actually this contradiction were one of the main reasons, that it were possible for us to learn the richness of PD-techniques. Since the end user has one primary goal - the final product - one tends to, and possibly should to a wide extent, accept the user approval of a certain phase as the end of this particular development phase. This of course being a sensible decision out of resource considerations present in every project, it is not necessarily a wise decision in a different perspective. As there is most often not a single unequivocal solution/representation to the problems considered in the application domain there is always the possibility that a better approach exists. There is no doubt that we, with our limited experience, would have been very tempted to move on to other things every time an accept was granted. The sessions with the supervisor kept us from this pitfall. Here we would, on our initiative, analyze ideas with potential need for extra attention relying on the supervisor's vast experience as a source of inspiration and advice. The result of the meetings often fostered new ideas. Of course these ideas could often be credited to an individual in the group, but more generally they were direct offspring of a lively discussion between the four of us – the net result being a genuine feeling of enlightenment since every one of us had part in the new discoveries. During these meetings the common frame of reference was of course the theoretic knowledge obtained in the SD course. In this way we were constantly able to test our understanding of the problem in hand as well as the theory applied, as we were practicing and discovering the different practical aspects of PD. At the same time we were able to make use of our knowledge establishing theories and initiatives on the intern meeting beforehand giving an excellent opportunity expressing creativity in every phase of the project – a process very satisfying in itself.

THE USER/DEVELOPER INVOLVEMENT

One of the major goals in system development is to get the user to "own" the artefact, as formulated by Mogensen and Trigg [10]. When the user becomes holder of the system in question they hopefully commit themselves to the artefact and feedback quality increases. Before the user can get to the state where it is possible to identify the artefact with the referent system, the developer needs to have a thorough understanding of the application domain in order to create a fruitful artefact/prototype. To this end we used several different approaches, and thereby experienced the practical side of Cooperative analysis [9 + 10]. As a result we gained the experience that the more the user is involved the more motivated we get, and how that again intensifies the user involvement. Of course a lot of other factors needs to be present to consummate this involvement, but the fact that a healthy cooperation is present is without comparison.

When a system is being developed during a relatively long period of time more workshops, prototypes and functionality can be accomplished. During the SD course only two workshops were held, but with the creation of the study group we were able to intensify the user-involvement, which is one of the most important essences of PD. The following figure illustrates how several workshops (indicated in the figure with a dotted line), combined with time, might produce the desired commitment/involvement from both user and developer:

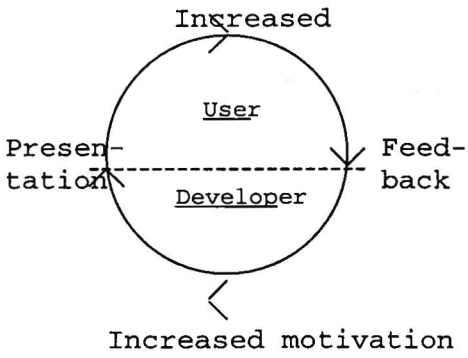

Figure 2: The involvement loop.

The figure illustrates the cooperation between the user and

developer, and how an element intensifies another. Initially the process starts at the very first meeting where the user clarifies what the application domain is. Hopefully the developer then gets the motivation to start producing the program, which is being presented to the user after some developing time. During the presentation the user preferably commits to the artefact and gives constructive feedback to the developer. At this point the involvement loop starts again, but now the program has evolved to a higher state, where more elaboration concerning the referent system and design has been provided. The dotted line indicates the getting through of a workshop where the program were discussed and feedback arose, and as mentioned in the previous section that happened approximately every fortnight.

LEARNING PARTICIPATORY DESIGN

By actually utilizing our theoretic knowledge to the full extent with our users we learned some of the practical sides of PD. Doing so made it possible for us to get our own experiences with PD and thereby looking at the project in question from our own perspective. Being able to do so is at the utter most importance, since theory is without much worth when there is no practice. The converse is also true since you can't practice PD without knowing the basic theory, which were taught to us during the SD course. So in order to be a successful PD-user both sides need to be mastered. When you then have the ability to see a complex problem from both a practical and theoretic viewpoint, there is greater chance of success in the problem solving. The study group, to a great extent, gave us that capability, since we achieved the crucial *hands on experience* with the use of PD-techniques.

The source of motivation is manifold – starting the study group in the first place was gratifying in itself, since this display of initiative meant that it was easy to find support and goodwill both in connection to the University as well as ALTS. As discussed in the section about the involvement-loop, we were able to maintain the commitment because of the increased interaction with the end-users at ALTS, this was only possible due to the amount of time at our disposal. In the process of the course we have been involved in the founding disciplines of participatory design including workshops, prototyping, and iteration between analysis, design and implementation. Meanwhile the recurring sessions with the supervisor assured that we were not pushed to far off track by being ready with ideas and advice when problems arose. In consequence we took a giant step towards being active PD-users by actually using PD actively. We hope by illustrating the forces and the possible dangers identified in our project, that we are able to inspire to similar initiatives and give ideas on, within the field of education, how to create the settings for *learning PD by PD.*

ACKNOWLEDGEMENTS

We acknowledge Preben Holst Mogensen for being a serious and motivated supervisor during the entire project. Also we thank the representatives from ALTS, the tennis club to whom the program were developed, for being a great partner. The workshops with ALTS and the meetings with Preben were constructive and none the less filled with patience and always held in a relaxed and humorous way.

REFERENCE

1. Blomberg, Jeanette, Suchman, Lucy & Trigg, Randsall. "Reflections on a Work-Oriented Design Project." *Human-Computer Interaction*, 11, 3. Page 52. A previous version appears in Proceedings of the *Participatory design Conference*, (Chapel Hill, north Carolina, 27-28 October, 1994) Palo Alto, CA: Computer. Professionals for Social Responsibillity, 1994, 99-109
2. Communications Vol. 38 #9
3. Floyd C. (1989). A systematic look at prototyping in approaches to prototyping.
4. Good, M. "Participatory design of a portable torque fedback device." In *Proceedings of ACM Conference on human factors in computing systems*, CHI '92 (May 3-7). Monterey, CA: ACM Press, 1992, 439-446 [' 96 p 52[13]]:[13]
5. Kensing, F. (1987). Generation of Visions in Systems Development. In P. Docherty, K. Fuchs-Kittowski, P. Kolm and L. Mathiassen (eds.) System Design for Human Development and Productivity: Participation and Beyond. North Holland
6. Mathiassen et. Al 1997 Lars Mathiassen, Andreas Munk-Madsen, Peter Axel Nielsen and Jan Stage. *Objekt Orienteret Analyse og Design*
7. McConnell S. 1996 Steve McConnell. Rapid Development: *Taming Wild Software Schedules* [p. 249]]
8. Mogensen P (1994) Challenging Practice – *an approach to Cooperative Analysis*. – p. 95
9. Mogensen P (1994) Challenging Practice – *an approach to Cooperative Analysis*.
10. Mogensen P. And R. Trigg (1992). Using artifacts as triggers for participatory analysis in PDC' 92: Proceedings of the Participatory Design Conference. November 1992.
11. Morgan G. 1997 Gareth Morgan. *Images of Organization*. –p. 249
12. Winograd T. (1998). *Bringing Design to software.* Chap 1 p. 6
13. Winograd T. (1998). *Bringing Design to software.* Chap 14