Mediating technical platforms to support the development of shared work practices

Susanne Bødker

Department of Computer Science

Aarhus University

DENMARK

This paper discusses two examples of how shared technical support platforms are adapted and developed in organizations. One is support for object oriented design and programming, the other, production of documents in an office. These two cases represent longitudinal studies of the use, the adaptation and the development of these platforms. This development involved active mediation of platform coordinators, who were users of the shared environment, at the same time as they possessed, and developed further, skills and procedures to undertake its continued development. The paper applies a developmental perspective in that its concern is how the technical platform and the work practices surrounding it, are developing over time, and how the technical and organizational conditions have been in support of or a hindrance to this development.

Tailoring and platform coordination

The development in use of technical platforms is in literature often called tailoring. This process has been studied widely for some time, as have the roles of the tailors, or, as they are called in this paper, platform coordinators. Whereas e.g. (Gantt & Nardi, 1992; MacLean et al, 1990) looked primarily at adaptation by individuals, Mackay (1990) pointed out that *tailoring is an organizational process*. How tailoring is an organizational process was illustrated further e.g. by Okamura et al. (1994) who pointed out how the human mediation of the introduction of a CSCW system was important, an observation also supported by (Trigg & Bødker, 1994), a study of the organizational conditions in case number two of this paper. The paper discussed the increased structuring and bureaucratisation of the tailoring process, and the development of formally recognized roles of the tailors. It further pointed out how tailoring, usually seen as enabling an ever-increasing variety of use patterns, can play an equally important role in organization-wide efforts to standardize. The activities of sharing and distribution, thus, are natural parts of the organizational process of adapting and appropriating technology.

Whereas e.g. MacLean et al. (1990) looked at the qualifications of tailors as something that, though it may develop, can be categorized into three ways of dealing with the platform, this paper proposes that it is necessary to study how the qualifications of tailors and users, and the platform in general, develop over time: *tailoring is an emergent phenomenon in an organization, and cannot be pre-planned*.

This kind of tailoring that we look at here is what Schmidt (Schmidt & Bannon, 1992) called articulation work: Articulation work is the secondary activities needed to divide, allocate, coordinate, schedule, mesh, interrelate, etc. the individual work activities. In other words are the *tailors a kind of gate-keepers that handle the articulation work related to the technical platform*, and they play an active role in mediating the relationship between the users and the platform, which is why this paper prefers the term platform coordination to tailoring. And further, it is a new kind of articulation work introduced *qua* the new platform

The paper discusses further the issues of organizational and technical mechanisms for platform coordination, the emergent competences of platform coordination, and the articulation work related to the use of the platform.

Research approach

Based on the assumption that platform coordination is under continuous development and not just a set of procedures that are adapted once and for all, this paper uses longitudinal studies to understand the specifics of how the platform, the competencies etc. develop over time.

The primary case was conducted in two sequences of qualitative interviews some three years apart: The first investigation was carried out together with 15 graduate students in spring 1993. We conducted 50-60 hours of qualitative interviews with system developers, managers, and in-house users, added to presentations and demonstrations.

In fall 1995 I returned for hour-long interviews with a system developer, a project manager, and the platform coordinator in charge of the shared classes and libraries. All three were trained computer scientists who did not work with the company in 1993.

The secondary case was part of an action-oriented research effort, lasting for two years while an organization introduced PCs running WordPerfect and Windows (see e.g. Trigg & Bødker, 1994). We followed the use of this technology from its early introduction, conducting informal interviews and observations of everyday work of a small group of early users. For six months I spent a half day every third week at the office. I helped with technical problems as they arose in exchange for time spent talking to the users and platform coordinators. After six months of less frequent interaction, we conducted interviews with the two platform coordinators, a programmer and a user.

Case One: TR-Partner

TR-partner is a spin-off company from two major suppliers and consultant companies of IT in the public sector. TR-partner has existed since 1991, and its focus is support for and planning of public transportation.

TR-partner was staffed partly with people from the "mother companies". They came from a variety of backgrounds and had experiences from traditional approaches, mainframes and relational databases. Before the spin-off, the companies had developed two transport planning systems, and several of the employees of the new company had a background in transportation planning.

The company had never before applied any systematic or structured method, and they saw object-orientation (OO) as a step towards better control of the process and of the quality of the products. Thus, the organization based itself on object oriented technology and a pilot project (TR-bus) was launched using Coad/Yourdon's approach.

In 1993, three groups of designers worked on the project. The traffic planners were used to working with the existing system, and they saw the project as an expansion of this. Of the technicians, some were part of the project from the very start, and they were part of the decision to introduce OO. They had some understanding of traffic planning from previous projects, and they took part in analysis and design as well as in programming. Their main concern was to become systematically object oriented. Other technicians were new-commers, who were hired because of their understanding of OO. They were mainly programming components that had been defined earlier through paper screen images and textual descriptions of functionality.

The project produced an OOA model that played a dominating role in connecting parts as well as activities. It was exposed on a wall in one of the main offices of the group, where people would meet to discuss various problems related to the implementation of TR-bus.

The project group had several problems regarding the actual use of OO, and at a point the project went back to some of their old ways. The overall design of screen images was put in the hands of one experienced designer/traffic planner. The remaining designers started programming. As a consequence the design of screen images did not consider the potential of OO such as inheritance and reuse.

The overall picture in 1993 was that the OOA model gave a lot of attention to the problem domain, whereas the traditional technical concerns were taking over as regards the design/implementation.

In 1995, the projects were still object-oriented, based on Windows, C++ and Oracle. In some cases the design documents were used for customer approval, in others, the users see the OO model and design documentation only to a very limited extent.

People who had been hired since 1993 typically had a background training in these methods and tools.

TR-partner had established a core library of 50-60 classes that were applied by all projects. This library was maintained by one person, the platform coordinator.

It was the responsibility of the project manager to handle overall analysis, design, and resource estimates for the various components of a particular project. In many projects, the OO analysis was very sketchy, and mainly an outline based on the previous knowledge of the project manager. The main emphasis, however, was on the design model which was oriented toward a solution, specifying which components the product consisted of, how they interacted, and what components from the project platform were to be applied in building the various parts (what the platform coordinator referred to as the parts list). The model was a formalization that served its main purpose well: to support making good programs. Furthermore, the model was the starting point for design of the user interface and database, and the specification of these components was seen as more important in the communication with programmers than the OOA model.

Though the model was a help in delimiting components to be handed out to programmers, the decomposition was an iterative process, where the sharing and the development of ideas were important.

The platform coordinator offered his service to the projects through active participation in particular in the design of programs. He knew the platform well enough to be able to produce, for each project, a "parts list" of objects and classes that the project would need from the shared library. Typically, one or two people from each project followed the development of the shared library closely and constituted the main contact between a project and the platform coordinator.

Earlier experiences showed how inappropriate it was to let each project expand on the general classes. Instead, based on the parts list, each project got its version of the shared objects and classes to work on and specialize to its needs.

The platform coordinator decided what to put into the shared classes and which classes to make shared. He was able to make these decisions through his close contacts with the projects, and at times upon direct requests. The main criteria for the design of shared classes and objects were that they had to be:

- easy to use,
- without too many preconditions,
- not too complex.

If these criteria could not be fulfilled, the idea was abandoned, or left to be thought through later. News groups and literature were often used in finding inspiration for various solutions.

The role of the platform coordinator together with his assistants on the particular project resembles what e.g. Jakobson et al.(1992) and Bürkle et al. (1995) talk about as an architecture group. Only in this particular case the role is entirely taken on by one individually.

Information about new "stuff" in the shared classes was spread at meetings, mainly emphasizing the ideas behind the design. The experience of the coordinator was that through such meetings one can avoid myths among programmers about how the shared classes may be used, what bugs they contain, etc.

Characterizing his own role, the coordinator said that by knowing the material i.e. the programming environment/the shared classes very well, he was able to give advice to project managers and to be in open dialogue with the programmers.

	1993	**1995**
The organization		
basis technology	Windows, C++, Oracle	+ core library of 50-60 classes
customer relations	Formal req. not part of OO	Increasingly integrated
qualifications	No previous structured methods Domain knowledge 3 types of people	New people with OO background
The project		
mediators	OO analysis model on central wall	Design but only sketchy analysis. Abstracting and grouping from problem domain.
cooperation	Programming by individuals in each their style	Project manager does overall design. Design model used for delegation of e.g. user interface and database design and

		programming
Platform coordination		
coordinator role/qualifications	none	Active cooperation Platform coordinator follows the latest from literature
adding to platform	Each project expands on general classes	Platform coordinator decides based on simplicity and relevance criteria
informing about platform	Ad hoc	Meetings. 1-2 people from each project work closely with platform coordinator
using platform	Ad hoc, by rumor	Parts list

In 1995 the main focus was on solutions rather than models of the application domain. However, with the platform and platform coordinator, the concepts of the solution had been aligned more with OO thinking than in 1993. The common platform, thus, while originally being aimed at sharing and reusing code, served as means of implementing new principles and working methods in the organization.

Case two: The AT study

The AT is a governmental institution that inspects and advises companies on health and safety matters. The use of PCs, WordPerfect, and Windows started in 1992 with a small group at the Aarhus branch consisting of eight inspectors and a secretary. One year later, the use of the technology had spread to all inspectors. Accompanying the technological change was a crucial change in work practice: secretaries stopped writing for the inspectors. Towards the end of the project, inspectors produced their own texts and performed most of their own information retrieval work. They had access to e-mail and to some central databases from their PCs, but they ran almost no other computer applications.

Adapting technology to the local needs of the AT branch office involved customizing WordPerfect with button panels, macros, standard forms, and paradigma (collections of legally valid standard phrases).

Overall, some 50 inspectors and 10 secretaries worked at the Aarhus branch, and two persons had official roles as platform coordinators, along with their work as labor inspectors. Moreover the organization hired a programmer to run the technical platform.

The platform coordinators' tailoring efforts were mostly driven by their concern for the quality of inspection work and documentation. Their tailoring, integrating and otherwise adapting the technology to the work in the organization made the platform coordinators the organization's official mediators and articulators between design and use, i.e. the ones who made sure that the new designs were indeed usable and crystallized ideas from the use in the organization into new designs. Together with a programmer, they formed the heart of a new, emergent, community of practice of tailoring at the AT, which is analyzed further in (Trigg & Bødker 1994). And, as further discussed in Trigg & Bødker (ibid.), the evolving shared platform was important for the development of new working methods in the organization.

Platform coordination	
coordinator role/qualifications	Users w. additional technical insight. Lacks design understanding/support
adding to platform	Systematic
informing about platform	Meetings
using platform	Automatic for all

Comparing the two cases

Both cases contain an important integration of design/development and use. Compared to traditional systems development the developers of the shared platforms are, in both cases, part of the communities of use, and mechanisms are set up to support the interaction between development and use of the shared platforms. These mechanisms include e.g the formalized collaboration with project managers at TR-partner, the technology committee at AT, and in both meetings where information is provided to users about the changed platform. It is characteristic to both cases that the development in use emerges through the active work of the platform coordinators, and that, at the same time, the technical platform is an important basis for its own further development.

The differences between the two cases are, however, equally noticeable. First of all the mechanisms for sharing and distribution that emerged to support the development process are somewhat different, and in particular rather different strategies for the mediation of the tailors have emerged. Furthermore, the technical qualifications of the two groups of users are rather different, which may partly account for the differences in sharing, co-operation and distribution around the shared platform.

The mechanism of sharing and distribution

In TR-partner it has taken a long time to develop the sharing and reusing of components. Several approaches were tried out: copying from others, extension of existing classes to accommodate for new exceptions. and in Finally the shared platform was established from which components are pulled out to be applied by the various projects. Building and sharing this OO platform is very much dependent on the active mediation of the project leader and of the platform coordinator. The project coordinator explains that the shared use of the platform goes much more smoothly when he is involved, than in projects where he is not that much involved. In this case, in other words, the complexity of the work that goes into presenting the platform and coordinating the use of it is reduced mainly through increased integration with the actual use/work.

In the working method of TR-partner, delegation and distribution of work take place as well through the active mediation of the project managers and of the platform coordinator, and for both OO provides useful encapsulation and abstraction mechanisms that supports this delegation.

At the AT, significant effort has been put into finding appropriate ways of distributing so-called "standards" among the workers. These include paradigms (collections of legal phrases), new or modified buttons and button panels, new WordPerfect forms and schema for generating them, and new or modified macros.

The sharing/distribution of standards happens in three ways:

1. Paradigms are discussed in meetings of a technology committee.

2. WordPerfect schemas and forms are designed by individual workers or by platform coordinators.

3. Macros and button panels are developed by the platform coordinators and distributed throughout the branch.

At first, standards (forms, macros, etc.) developed and spread opportunistically. Someone heard about tailoring done by a colleague, copied their modifications, and perhaps performed further customizations of their own. Later the process became more systematic; ideas were conveyed to the platform coordinators, who used them as the basis for new standards. Distributing standards at the AT was semiautomatic; individual PC's were configured to download the new facilities when booted each morning. When potentially disruptive changes were downloaded, the workers were explicitly notified and told how the new functionality was intended to be used. Except for certain modifications related to the technical infrastructure (e.g. the network), people at the AT were free to use or ignore the standards they received. In any case, they normally did not make standards of their own independent of this process.

The platform coordinators were pleased with the new process, arguing that it gave equal access to standards throughout the branch, improved the quality of the standards, and eased their own work by ensuring consistency across the branch.

It is interesting to note how in AT sharing and distribution became formalized and somewhat automatic whereas in TR-partner the active mediation of the platform coordinator continued to be crucial. In AT, ensuring consistency across a fairly large number of users was important, whereas at TR-partner, the number of users, in terms of projects with particular needs, was much smaller. Furthermore, as the TR-partner platform coordinator pointed out, flexibility was important. It was essential to be able to support the varying needs of the individual projects and, thus, an increased variety of use patterns, in contrast to the AT situation where consistency, and producing platform components that were immediately usable by people without very much insight into the technical platform was essential, if for nothing else then for the workload of the platform coordinators. However, through active mediation by the TR-partner platform coordinator the variety of use patterns co-existed with sharing and reuse as natural parts of the organizational process of adapting and appropriating technology.

Tailoring competence

Where the AT experience showed how platform coordinators were placed between development and use, the TR-partner platform coordinator was more integrated and everybody should in principle be able to master what the platform coordinator did. In neither case the development was based on abstract models of use, or on formal conceptions of the technical artifact. However, as we pointed out for the AT case (Trigg and Bødker, 1994), the lack of ability or training to step back and think in a more overall and abstract way was a problem for the platform coordination in that it failed to do more radical construction changes to the platform. At TR-partner the initial technical skills were somewhat mixed, and there was initially a tendency of "falling out" of object orientation when things got critical. Despite this, all programmers possessed much more technical and abstraction skills than in AT, which made it interesting as the organization seemed to be on the same route as the AT until the present platform coordination work was initiated: The various programmers made their various here- and-now solutions to particular problems without concern for connections to other parts or for stability and accessibility as regards reuse. Thus, the problem seems less tied in with technical abstraction skills *per se*, and more related to a concern for the particular problems of creating parts of a technical solution that may be reused by others. Certainly one needs to be interested in how this was done: How the reusable components were made stable, accessible, got documented, etc. With their platform coordinator TR-partner got a person who had an interest in such matters, and a background that made it possible to look for advanced solutions to various problems. Thus, he took an interest in design patterns, and other advanced OO concepts. For the same reasons, the AT platform coordinator was asking for a technically competent "sparring partner" for discussions to potentially solve such problems (Trigg & Bødker (1994)). This was not needed in TR-partner because of the technical competencies of the platform coordinator.

The development of the shared platform, and the way the platform coordinators came to handle tailoring in the specific, are an interaction process with users, with fellow platform coordinators, and with the material worked on: In neither case could anybody have preplanned (designed/implemented) the actual shared platform, nor the procedures surrounding its use and further development. This has consequences in how we may think of tailoring or platform coordination methods, or of including this kind of continuous development into systems development in that we must see the platform not as static but under continuous development.

The work to make a shared platform work

Organizing the shared (re-)use was in both cases a stepwise process starting from very specific solutions to very specific problems, moving towards solving more general problems, in more general situations. However, both cases also showed that the shared use was only possible trough the continuous efforts of the platform coordinators. In AT the systematization of the process was important. In TR-partner the active role of the platform coordinator remained important. In both cases, the articulation work, or to use the terms of Bowers (1994), the work to make a shared platform work, changed over time, but did not go away. The active human mediation in both cases continued to be important, it was at no point left to the shared platform, and in neither case the platforms could be seen as reducing articulation work, as an overhead that would go away if people were able to cooperate around the shared platform. On the contrary, most of this work only existed because the platform and the ideas of sharing were introduced.

Conclusions

In the two cases, incorporating a new, shared platform into the everyday work of the organization was not just a matter of adapting, or adapting to, the shared platform. The adaptation process was one of cooperative development, making active use of integrating development and use, both at the level where the developers were also users and worked as users, and through active participation of the remaining users in the development of the platform.

A number of mechanisms for sharing, or distribution of the platform components that were emerging in the two cases partly dependent on the skills of the platform developers. In particular the paper has shown two rather different strategies for platform coordination: one where active participation in projects of use is essential, and one where structures and delegation became increasingly important. Participation and formalization are two alternative strategies for dealing with the increasing complexity of platform coordination. In both cases the efficiency and quality of sharing went hand in hand with additional work of the platform coordinators, additional articulation work. This means that either these kinds of platforms are not computer supported cooperative work (CSCW), as this is defined by Schmidt & Bannon (1992), or we have to abandon the reduction of (the complexity of) articulation work as a quality criteria for CSCW systems. This paper has illustrated that the latter is to be preferred to the former, since this kind of articulation work is an important vehicle for increasing the quality of the product resulting from the shared platform use.

Acknowledgements

To the platform coordinators of the two organizations - Jørn Andersen, Tommy Gilkov, Peter Hübel - and the rest the people of their organizations. The AT study was done in co-operation with Randy Trigg, who has also provided useful comments along with Olav Bertelsen. Marianne Iversen and Helle Holm-Nielsen helped improve the language of the paper.

References

Bowers, J. (1994). The work to make a network work: studying CSCW in action, in Futura, R. & Neuwirth, C: *Proceedings of CSCW 94*, ACM press, pp. 287-298.

Bürkle, U., Gryczan, G. & Züllighoven, H. (1995). Object-oriented system development in a banking project: Methodology, experience and conclusions., *Human Computer Interaction* 10 (293-363).

Button, G. & Sharrock, W. (1994). Occasioned practices in the work of software engineers, in Jirotka, M. & Goguen, J. *Requirement engineering. Social and technical issues*, London: Academic Press, pp. 217-240.

Gantt, M., & Nardi, B. A. (1992). Gardeners and gurus: Patterns of co-operation among CAD users. In *Proceedings CHI '92* (pp. 107-117). Monterey, CA: May 3-7.

Jakobson, I., Christersson, M., Jonsson, P., & Övergaard, G. (1992). *Object-oriented software engineering - a use-case driven approach*. Reading, MA: Addison-Wesley.

Mackay, W. E. (1990). Patterns of sharing customizable software. In Proceedings of *ACM CSCW'90 Conference on Computer-Supported Cooperative Work* (pp. 209-221). Portland, Oregon: ACM Press.

MacLean, A., Carter, K., Lovstrand, L., & Moran, T. (1990). User-Tailorable

Systems: Pressing the Issues with Buttons. In *Proceedings of ACM CHI'90 Conference on Human Factors in Computing Systems* (pp. 175-182).

Okamura, K., Fujimoto, M., Orlikowski, W. J., & Yates, J. (1994). Helping CSCW applications succeed: The role of mediators in the context of use. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work* , ACM press, pp 55-66

Schmidt, K. & Bannon, L. (1992). Taking CSCW Seriously. Supporting articulation work, *CSCW journal* vol. 1 nos. 1-2 (pp. 7-40).

Trigg, R. & Bødker, S (1994). From Implementation to design: Tailoring and the emergence of systematization in CSCW, in Futura, R. & Neuwirth, C: *Proceedings of CSCW 94*, ACM press, pp. 45-54.