In O.W. Bertelsen, S. Bødker, and K. Kuutti (eds.), *NordiCHI 2002: Proceedings of the Second Nordic Conference on Human-Computer Interaction* (Aarhus, Denmark, October 19-23, 2002), pp. 199-202. ACM Press, New York.

Preprint version

Organisational Implementation: A Complex but Under-recognised Aspect of Information-System Design

Morten Hertzum
Centre for Human-Machine Interaction
Risø National Laboratory
DK-4000 Roskilde, Denmark
+45 4677 5145
morten.hertzum@risoe.dk

ABSTRACT

The development of information technology (IT) is often seen as consisting of analysis, design, technical implementation, and testing. While this may involve user input at several stages – to support the software engineers in devising and revising the system – this view on systems development marginalises organisational implementation. Organisational implementation comprises the activities that prepare organisations and users for a new system as well as the activities that prepare the system for the transition period during which it enters into operation and takes over from previous systems and artefacts. This study analyses three organisational-implementation activities undertaken during the complete redesign of a large information system: the information plan, the data conversion, and the release strategy. It is found that these activities must be carefully aligned with the technical implementation of the system and that they involve the development of additional system facilities. In sum, this study provides evidence of the complexity and importance of organisational implementation and, thereby, argues that it must be recognised as a first-rate constituent of the design process.

Keywords

Organisational implementation, Data conversion, Release strategy, Information-system design, Software engineering.

INTRODUCTION

An assessment of five recent IT projects in the Danish public sector finds that the customers as well as the developers have had unrealistic expectations about how easily the systems could be installed and put into operation (Teknologirådet 2001). At a general level this indicates that the involved actors have been overly focused on the technical aspects of the projects and have underestimated the organisational aspects. The frequency and consequences of this fallacy are well documented (e.g., Eason 1988, Gasser 1986, Greenbaum & Kyng 1991, Keen 1981, Kling & Iacono 1984). More specifically, the organisational implementation of information systems must be recognised as a first-rate constituent of the design process and incorporated in software-engineering models and procedures. From a socio-technical point of view (Mumford 1987) this is quite obvious, but as an illustrative example of the current state-of-affairs Sommerville (1996), in his widely read textbook on software engineering, devotes just two of the 742 pages to organisational implementation.

This study analyses, and calls attention to, three organisational-implementation activities undertaken during the complete redesign of a large information system[1]. These three activities concern the information plan, the data conversion, and the release strategy. Whilst these activities may not be exotic they are nevertheless complex and crucially important to the success of the project. The aim of this study is to provide evidence of this complexity and importance and, thereby, raise the general awareness of organisational-implementation issues. Irrespective of the quality of the technical implementation, information systems will likely encounter severe problems, or even fail, if the organisational implementation is not handled properly.

THE CSA PROJECT

The company where this study took place is a large software house, which has developed and marketed a range of systems for use in the Danish municipalities. The studied project concerns a system to support municipal authorities in the handling of cases concerning child support and alimony (CSA). The CSA project was initiated in 1999 and will, according to the project plan, last three years. During its first year the project was staffed with a project manager, eleven designers/developers, two service consultants, a methods & tools consultant, a usability specialist, and a secretary. The project manager and six of the

[1] It should be noted that redevelopment, as opposed to development from scratch, is a very common phenomenon. For example, Yourdon (1989) estimates that in some businesses 80% of all software expenditure is consumed by maintenance and evolution.

designers/developers worked full time on the CSA project, the remaining ten persons were assigned to the CSA project on a part-time basis. In its second year the CSA project was extended with additional staff. This multidisciplinary group – in the following termed the CSA engineers – is comprised of people with an average of more than ten years of professional experience.

The CSA engineers are to completely redevelop the existing CSA system, which several of them have been heavily involved in developing and maintaining. Whereas the existing CSA system contains substantial amounts of code that duplicate functionality from other systems made by the company, the new CSA system will distribute this functionality onto components that are to be developed by other project groups in the company (Hertzum 2000). The adoption of component-based design means that the CSA engineers collaborate closely with a number of people outside the CSA project to negotiate and follow up on the definition and development of the components. Naturally, the CSA engineers also interact with management, marketing, user representatives, and a number of other stakeholders internal as well as external to the company.

METHOD

The data collected for this study include that I have (1) participated in the two-day seminar arranged to kick off the project, (2) been present at the fortnightly project meetings, (3) conducted interviews with eleven of the core project participants, and (4) inspected various project documents. The meetings and interviews have been recorded on tape and transcribed. While the data collection was most intense during the first year of the CSA project it has, by now, been going on for 2½ years. This study is based on an analysis of the 42 project meetings that have been observed (79 hours), supplemented with data from the interviews.

The main purposes of the project meetings have been to provide a forum for sharing information about the status of the project, maintaining awareness of the entire project, coordinating activities, discussing problems and progress, making decisions, and reviewing major project documents. During the meetings, I have been seated at the meeting table with the other people present. From their point of view, I have been invisible in that I was not to be spoken to and have myself remained silent. During the breaks, I have talked informally with people.

The interviews provided an opportunity to talk with the CSA engineers about their individual experiences and concerns, and to dig deeper into issues and discussions that were merely hinted at during the meetings. The interviews, which lasted 1-1½ hours each, concerned the CSA engineers' roles and responsibilities in the project as well as their views on what was critical to successful completion of the project.

ORGANISATIONAL IMPLEMENTATION MATTERS

The work directed toward the organisational implementation of the new CSA system started when the project was initiated and will continue throughout the project.

Information Plan

The first activity undertaken to manage the organisational implementation was the creation of an information plan. The information plan was reviewed and approved at the first project deadline, along with the requirements specification, and provided a schedule for when users were to receive the variety of information necessary in preparing for and starting to use the new system. This involved, among other things, information about (1) when the new CSA system would be released, (2) the new facilities included, (3) the technological platform required, (4) how data would be transferred from the existing to the new system, and about (5) the scheduling of education and training.

The information plan defines a number of dates at which large numbers of people will be gathered to receive, for example, marketing information or education and training. For such events to be effective they must be carefully aligned with the progression of the development of the system. Thus, the information plan is dependent on the CSA engineers' ability to correctly estimate how long it will take them to develop the new CSA system. This dependence entails a conflict of power in that the organisational implementation is mainly the responsibility of the service consultants among the CSA engineers whereas the technical implementation is the responsibility of the designers/developers. The service consultants are, however, not in a position to set or insist on deadlines; they, rather, have to dynamically adapt their planning and activities to the evolving status of the technical implementation. Hence, the information plan must, on the one hand, be flexible and floating in order to accommodate delays and rearrangements of the technical implementation and, on the other hand, be fixed well ahead of time to enable timely announcement and planning of organisational-implementation activities. Under such circumstances it is truly demanding to manage user expectations and ensure that users are ready for the new system once it is released.

Data Conversion

CSA cases last for years (e.g., child support cases generally last until the child turns 18). It would therefore introduce an inordinately long transition period if the existing cases remained in the existing CSA system and the new CSA system was only used for cases initiated after its release. Further, the number of existing CSA cases precludes a manual procedure for terminating the cases in the existing system and re-entering them in the new CSA system. Finally, CSA cases are modelled differently in the new CSA system to enable new facilities and remove shortcomings in the existing system. Thus, for many cases it is no simple matter to perform the mapping from the existing system to the new system. Automatic data-conversion procedures will not be able to correctly convert

all CSA cases, so it is instead necessary to develop facilities for the automatic conversion of most cases and for supporting users in the conversion of the remaining cases.

As an example of the complexities of the data conversion there is, in the existing system, no representation of an entire CSA case, only two separate representations of its two subcases. One subcase concerns the person obliged to pay CSA and is the responsibility of the municipality in which this person lives. The other subcase concerns the person entitled to receive CSA and is handled by the municipality in which the payee lives. In the new system the two subcases will be two different views on one unified case. Thus the data conversion involves the correct pairing of the subcases of the existing system, and this turned out to be a quite complex task. Every Danish citizen has a unique CPR (Central Person Register) number, which is the standard means of identifying people in municipal systems, including the CSA system. However, some people, such as those who are Danish residents but not Danish citizens, have a surrogate CPR number rather than a genuine one. For some reason surrogate CPR numbers are assigned locally by the individual municipalities without any coordination of the numbers assigned by different municipalities. Thus, several persons may have the same surrogate CPR number. Further, if a person is obliged to pay CSA to more than one payee then the municipality may, in the existing system, decide to collapse these several subcases into one subcase. During the data conversion such collapsed subcases, which are not uncommon, have to be split into independent subcases before they are paired with the subcase for the payee (see Figure 1). Here the problem is not so much in identifying the right persons as in correctly dividing the payments among the payees.

The work on the data conversion started six months into the project and involves test conversions and feedback to the users about cases that cannot be automatically converted. Based on this feedback the users adjust and add to the registration of existing cases to enable automatic conversion of as many cases as possible. These adjustments and additions can be evaluated during the next test conversion. This way the data conversion becomes a controlled but lengthy process.

New systems are not introduced into a blank world. Rather, the task-artefact cycle (Carroll et al. 1991) reminds us that new artefacts replace old ones and redefine existing tasks and ways of working. When a new system replaces another digital system, rather than a physical or analogue artefact, data conversion or some other automated means of backward compatibility becomes important to the successful organisational implementation of the new system. Discarding this issue to save software-engineering resources will typically force users to retain the old system – and the skills needed to operate it – along with the new system to be able to access data generated before the introduction of the new system. In environments where users frequently need access to old, unconverted data this drawback will detract severely from the value of the new system.

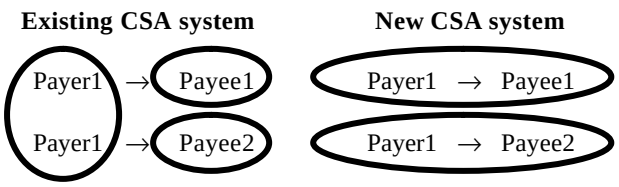

Figure 1. Example of re-pairing of subcases during the data conversion. Payer1 is obliged to pay CSA to both Payee1 and Payee2.

Release Strategy

The way in which the new CSA system, once completed, is going to be put into operation is a crucial phase in its organisational implementation. The CSA engineers have considered several release strategies:

- *"Big bang"* where the complete system is released to all customers at the same time. This strategy is technically simple because there is no period of transition from the existing to the new system and thus no need for extra system facilities dedicated to the handling of the transition period. However, "big bang" is obviously a risky strategy because users generally find it difficult to use a new system for the first time and because early errors will affect many users. Thus, the support centre will receive many calls but probably not be able to provide swift answers because the difficulties and errors will be new to the service consultants as well. Though temptingly simple from a technical point of view the CSA engineers found that it would be naïve to use "big bang" for the release of a system as complex as the CSA system.
- *Case-based* release, which consists of adding a tag to each case indicating whether it is to be handled with the existing or the new system. Initially only straightforward cases are handled with the new system but gradually more and more cases are moved to the new system. If successfully implemented this strategy is very flexible. There are, however, serious drawbacks. First, users will be changing back and forth between the existing and the new system when they switch from one case to another. This is confusing to the users who are striving to learn the new system and adapt their ways of working. Second, CSA cases are not static but may dynamically develop in ways that change a case from simple to complex. This necessitates prohibitively complex facilities for migrating cases from the new system back to the existing system, and caused the CSA engineers to look for another release strategy.
- *Region-wise* release where the system is initially released to only a subset of the users, such as a geographically confined region. This incremental strategy reduces the consequences of early problems and errors because fewer users will experience them. It is, however,

not possible to define a region for which all CSA cases are either internal (both payer and payee belong to the region) or external (neither payer nor payee belongs to the region). Thus, region-wise release requires the development of temporary but elaborate means of handling the CSA cases that are only partly included in the region. This involves both automatic facilities for handling some of the split cases and decisions to handle other cases manually during the transition period. Despite these complications the CSA engineers chose this release strategy. A further argument in favour of this strategy is that it lends itself to a practicable, region-wise schedule for education and training.

Numerous studies (e.g., Keen 1981, Kling & Iacono 1984) have shown that information systems affect the distribution of power in organisations and become subject to organisational politics. This means that in addition to learning the newly introduced system and establishing new ways of working the users are also in the midst of a process where personal positions and taken-for-granted arrangements are evaporating and new ones have yet to settle. These aspects of the organisational implementation are beyond the software engineers' control but software engineers should be aware that their criteria for what constitutes a smooth system release may be very different from what the users, given their perspective on the new system, are expecting or able to cope with.

CONCLUSION

Organisational implementation comprises the activities that prepare organisations and users for a new system as well as the activities that prepare the system for the transition period during which it enters into operation and takes over from previous systems and artefacts. This study shows that unless naïvely simple solutions – such as a "big bang" release strategy and no data conversion – are chosen organisational implementation involves extra complexity, extra work, and extra system facilities. Further, organisational implementation concerns issues that have considerable impact on the users' reception of the new system. Software engineers are likely to experience problems if they treat organisational implementation as a mere appendix to the development process. The more the system is integrated into the users' execution of their work, the more severe these problems are likely to be.

Seemingly, software engineers tend to underestimate the complexity of organisational implementation or under-recognise its importance. This study has argued that organisational implementation – bringing the system to its users – is as important an element of software engineering as user involvement – the more well-recognised activities of bringing the users into the design process.

ACKNOWLEDGEMENTS

This work has been supported by the Danish National Research Foundation through its funding of the Centre for Human-Machine Interaction. I wish to thank the members of the CSA project group who have put up with my presence in spite of their busy schedule.

REFERENCES

Carroll, J.M., Kellogg, W.A. & Rosson, M.B. 1991, 'The task-artifact cycle', in *Designing Interaction: Psychology at the Human-Computer Interface*, ed J.M. Carroll, Cambridge University Press, Cambridge, pp. 74-102.

Eason, K.D. 1988, *Information Technology and Organisational Change*, Taylor & Francis, London.

Gasser, L. 1986, 'The integration of computing and routine work', *ACM Transactions of Office Information Systems*, vol. 4, no. 3, pp. 205-225.

Greenbaum, J. & Kyng, M. (eds) 1991, *Design at Work: Cooperative Design of Computer Systems*, Erlbaum, Hillsdale, NJ.

Hertzum, M. 2000, 'Component-based design may degrade system usability: Consequences of software reuse', in *OZCHI 2000 Conference Proceedings*, eds C. Paris, N. Ozkan, S. Howard & S. Lu, CSIRO, North Ryde, Australia, pp. 88-94.

Keen, P.G.W. 1981, 'Information systems and organizational change', *Communications of the ACM*, vol. 24, no. 1, pp. 24-33.

Kling, R. & Iacono, S. 1984, 'The control of information systems developments after implementation', *Communications of the ACM*, vol. 27, no. 12, pp. 1218-1226.

Mumford, E. 1987, 'Sociotechnical systems design: Evolving theory and practice', in *Computers and Democracy: A Scandinavian Challenge*, eds G. Bjerknes, P. Ehn & M. Kyng, Avebury, Aldershot, pp. 59-76.

Sommerville, I. 1996, *Software Engineering*, 5th edition, Addison-Wesley, Reading, MA.

Teknologirådet 2001, *Erfaringer fra statslige IT-projekter – hvordan gør man det bedre?*, Report no. X, Teknologirådet, Copenhagen, Denmark.

Yourdon, E. 1989, 'RE-3: Re-engineering, restructuring and reverse engineering', *American Programmer*, vol. 2, no. 4, pp. 3-10.