

FOUR INTEGRATION PATTERNS: IS DEVELOPMENT AS STEPWISE ADAPTATION OF TECHNOLOGY AND ORGANISATION

Bendik Bygstad, Norwegian School of IT, Schweigaards gt. 14, 0185 Oslo, Norway,
bendik.bygstad@nith.no

Peter Axel Nielsen, Aalborg University Fredrik Bajers Vej 7, DK-9220 Aalborg, Denmark,
pan@cs.auc.dk

Bjørn Erik Munkvold, Agder University College, 4604 Kristiansand, Norway,
bjorn.e.munkvold@hia.no

Abstract

Integration and adaptation between an information system and the related business organisation is an important challenge for IS development project managers. Both IS research and software engineering research have contributed important lessons on integration, but at the practical level of IS development projects the processes of integration and mutual adaptation is less well understood.

In this paper we apply a socio-technical perspective to analyze the integration of stakeholders and technical components, and we identify four generic integration patterns; Big Bang, Stakeholder Integration, Technical Integration and Socio-Technical Integration. Three case studies of iterative software development projects are used to illustrate the patterns. These illustrations highlight the management challenges of iterative integration. While early integration increases the likelihood of implementation success, it also increases the complexity of the projects.

For practitioners the four integration patterns represent an analytical tool for planning iterative systems development projects. For IS research the four integration patterns are a contribution to a vocabulary for describing socio-technical integration.

Keywords: Software development process, integration patterns, mutual adaptation

1 INTRODUCTION

This study analyzes three cases of iterative information systems (IS) development. Our main concern is to understand the management challenges faced by the business-oriented IS project manager. How should the information system integrate with and adapt to the business organisation in order to make the solution work?

The alignment between the organisation and the information system is a long-time concern for both IS research and the practice community. Successful IS development and implementation depends on a socio-technical approach (Coakes et al., 2000), both technical and political alignment (Leonard-Barton, 1988), as well as comprehensive business and user participation (Kappelman and McLean, 1994). There has also been a shift from the somewhat simple concept of users to the broader notion of social actors (Lamb and Kling, 2003).

These findings have been recognized in software engineering (SE) research as well, though in software engineering they are conceptualized more as problems with getting the requirements right. In 1988, Boehm introduced the spiral model for software development to address the risk of changing business requirements (Boehm, 1988). A decade later the mainstream SE frameworks, like the Rational Unified Process (RUP), were structured in iterative and incremental development steps, emphasizing the step-wise integration of technology and different types of stakeholders (Jacobson et al., 1999).

However, at the more operative project level the processes of integration and mutual adaptation are less well understood (Majchrzak et al., 2000). There are many questions that the IS project manager must address. How, and when, should the new software components in the development project be integrated with the existing technical infrastructure? How, and when, should the different stakeholders be enrolled into the project? To what extent do modern software engineering methodologies support the stepwise integration and adaptation between technology and organisation? What are the potential management trade-offs in such projects? In this paper we address these questions and seek answers in a set of four generic patterns of integration, which we develop based on previous theoretical research and an empirical study of three iterative software development projects.

The next section defines the key terms of integration and adaptation, in a socio-technical context. Then, in section 3, we briefly present our research approach and in section 4 we describe four integration patterns in detail. In section 5 the four patterns are illustrated through three longitudinal case studies. In section 6 we discuss practical and theoretical implications, and limitations. We conclude briefly in section 7 with our main findings and point to further research.

2 INTEGRATION AND ADAPTATION: A SOCIO-TECHNICAL PERSPECTIVE

In a socio-technical development perspective the main objective is to optimize the interplay between organisation and technology by taking a more holistic view on both: Technology is not only an artefact, but also embedded in organisational and human behaviour. Vice versa, an organisation consists not only of people, but is enabled and shaped by its use of technology. Thus, we regard our object of study as an emerging socio-technical network, consisting of both stakeholders and technology (Kling and Scacchi, 1982; Coakes et al., 2000).

In the practical context of IS development we regard this as the integration of two socio-technical networks; the business organisation and the development project, both consisting of stakeholders and technology. At a point in time the two networks, ideally, integrate fully and become one. The integration between the information system and the business organisation is seen through a process lens (Newman and Robey, 1992); we analyze integration and adaptation over time. The organisation is

often large and complex, coexisting and depending on information infrastructures (Ciborra 2000). In the beginning (at time 1), the software development project is usually quite small, and then grows gradually as the project proceeds. The information system is then set into production (at time 2), and the project is usually (but not always) terminated. For successful development projects, somewhere along the paths from time 1 to time 2 the different stakeholders and technology are enrolled and a socio-technical network is adapted and integrated. To a certain extent it is possible to plan and design parts of the integration, and this is usually done in a requirements specification and systems design documents. However, because of the emergent and social nature of the socio-technical network, important parts cannot be specified at the beginning of the project. In this research we analyse the integration process in three steps:

- *Enrollment of stakeholders and/or technology.* Formally, this is an act of agreement that specifies the role of the project and the new system in the organisation or the information infrastructure. In practice it is often necessary to convince stakeholders that it is in their interest to participate. Stakeholders may include future users, managers, other projects, IT managers, data centre technicians, vendors, consultants and also customers (Jacobson, 1999).
- *Adaptation of technology and/or organisational processes.* Fitting the software to the organisation is done through different techniques such as requirements, user input and prototyping (Jacobson, 1999). Conversely, the organisation may have to change to achieve the benefits expected from the technology; business processes may be redesigned and relationships to other organisations may be affected (Leonard-Barton, 1988). At the level of information infrastructure, a new system must comply with technical architectures and interfaces, but the infrastructure may also have to change because of the new system (Hanseth and Monteiro, 1996).
- *Stabilisation.* After adaptation the different elements may be validated and set into production. A successful implementation is here defined as the stabilisation of the socio-technical network; i.e. that the solution is technically stable and key stakeholders agree on the value of the network (Hanseth and Monteiro, 1996).

The three steps of integration are often difficult to achieve, for several reasons. First, both the development process and the organisation change during the time of the project, thus making the requirements a moving target (Boehm, 1988). Second, the new software system must integrate with the existing technical environment in the business organisation, which often is a very complex task. And third, implementation success is often defined differently by different stakeholders. For the software engineers, a successful implementation is usually achieved when requirements are fulfilled and the system is technically stable, while for the users a successful implementation is to make a new or modified organisational process fully operational through the use of the new information system (Alter, 2000).

3 RESEARCH APPROACH

To study integration in modern iterative IS projects we conducted three longitudinal case studies, following the approach of Longitudinal Process Research (Pettigrew, 1985). The cases were:

- Development of an e-business solution at an international airline
- Development of an audit support system for a large government auditing agency
- Implementation of a learning management system (LMS) at a university.

The cases were chosen on two criteria. They all addressed the challenge of integration, and they also followed modern, iterative software development process models. Data collection was done over 14 to 16 months at each site, using a wide range of sources; semi-structured, periodic interviews (a total of 40) with various stakeholders, participatory observation and an extensive amount of project documents. The empirical data were analyzed in several steps. First, a time line with important events and development process iterations was documented. Second, development process iterations, context,

stakeholders, and artefacts were modelled graphically. Third, a case description was written, and validated by the interviewees.

After this initial data analysis we applied a socio-technical framework as our interpretive lens. Stakeholders, technical components, organisational processes, information infrastructure and political factors had all played important roles. We then tried to understand the dynamics of the cases by identifying the mechanisms that integrate the socio-technical networks. Further, drawing on concepts from software engineering research (Jacobson et al., 1999) and earlier research on patterns of process integration (Lloyd et al., 1999), four process patterns emerged as a generic and comprehensive set of integration patterns. These patterns are explained in detail in sections 4 and 5.

Using the principle of the hermeneutic circle in interpretive IS research (Klein and Myers, 1999), the cases were reinterpreted within the patterns approach. The patterns were then finally discussed with the interviewees and validated.

4 FOUR INTEGRATION PATTERNS

The term pattern is used in the broad sense as a general solution to a common problem in a context (Ambler, 1998). We have described the IS development project and the organisation as two socio-technical networks, consisting of both stakeholders and technology. During an IS development project the integration between the information system and the organisation may happen early and stepwise, or it may be performed at the end of the development project, when the system is put into production and becomes fully operational in the business organisation. Thus, we have two types of elements (stakeholders and technology), and two types of integration modes (step-wise and at the end of the development project). Combining these two dimensions we identify four different integration patterns:

- “Big Bang” - the integration of both stakeholders and technology is performed at the end of the development project.
- “Stakeholder Integration” - the integration of stakeholders is done step-wise, while the integration of technology is done at the end of the project.
- “Technical Integration” - the integration of technology is done step-wise, while the integration of stakeholders is done at the end of the project.
- “Socio-Technical Integration” - integration of both stakeholders and technology is done step-wise.

The four patterns originate partly from the three cases and partly from the RUP; a mainstream software development methodology (Jacobson et al., 1999). In RUP the issue of integration is prominent and it addresses both technical integration and the integration of external stakeholders into the project. In this section we analyze the advantage and disadvantages of each pattern. We also comment on relevant knowledge from IS and software engineering research. In particular we assess the support for the pattern in RUP.

4.1 The Big Bang Pattern

In this pattern (figure 1) the network between the IS development project and the business organisation is established at the time when the development project is completed, and the software product is delivered. Integration between the processes is described in a requirements specification. The development project then produces the required software and documentation. At the production date the system is organisationally implemented, i.e. the components are installed and set into production, and the users are enrolled and taught how to use the system.

The main advantage of this pattern is a high degree of internal project control as well as software system integrity. The focus of the project is on the software system, executing control by traditional project management techniques. Supporting mechanisms in RUP are requirements, phases and workflows. Disadvantages are well documented: Late integration runs the risk of both user resistance and of technical integration problems (Royce, 1998; Jacobson et al., 1999). Both the organisation and

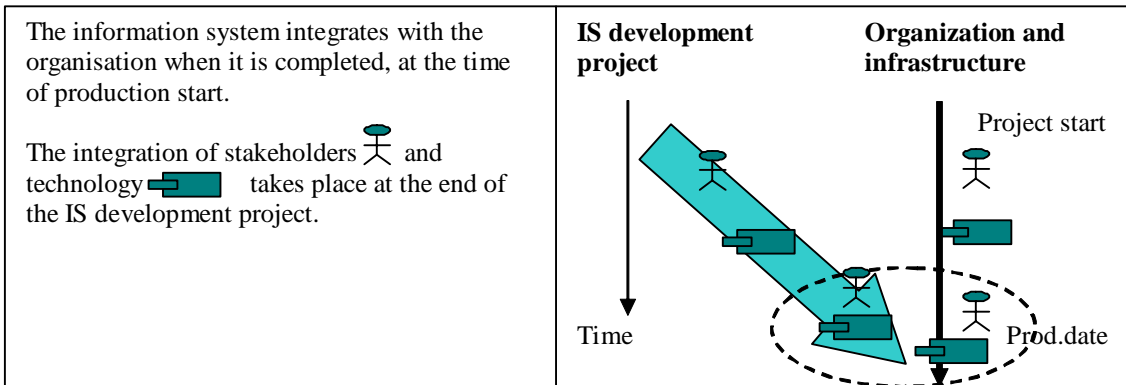


Figure 1. The Big Bang Pattern

infrastructure are subject to change, making it very difficult to predict the integration in detail at the start of the development project. Both user and technical requirements may be incomplete or inadequate at the start of production. Integrating a large number of new technical components into legacy systems has been made feasible the past years through interface standards and middleware, but puts heavy demands on testing and configuration management and tools. Even small and seemingly trivial technical mismatches may lead to serious production problems. Thus, stabilizing the socio-technical network is difficult within this pattern, because there is very little room for adaptation and learning during the project.

4.2 The Stakeholder Integration Pattern

In this pattern (figure 2) the stakeholders from the business organisation are enrolled step-wise, through workshops, user groups and other organisational mechanisms. Early user participation has long been established as an important success factor (Mumford, 1995; Lamb and Kling, 2003). Further, it secures more reliable requirements and early user validation of the product (Jacobson et al., 1999). The supporting mechanisms in RUP for this integration pattern are iterative (step-wise) and stakeholder based development (including, in principle, all actors that are influenced by the project).

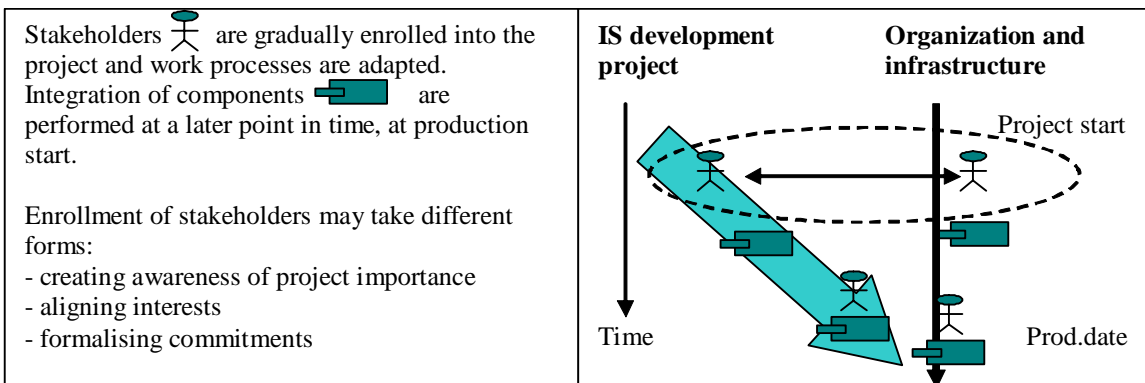


Figure 2. The Stakeholder Integration Pattern

Disadvantages include the risk of unstable commitment from the business stakeholders if this is only declared, but not supported in practice. They may lose interest in the project, and return to their own organisational processes. They can also become so involved that they *change side* and become part of the development project and move away from the business organisation. The late technical integration may also lead to some of the stability problems described in the Big Bang Pattern, since even small technical mismatches may lead to serious production problems.

4.3 The Technical Integration Pattern

In this pattern (figure 3) the focus is directed at step-wise technical integration. A network of components and systems is established early in the project by step-wise enrolment. The main advantage of this pattern is that early and step-wise technical integration secures technical stability at the time production starts. It may also validate the technical quality of the software system.

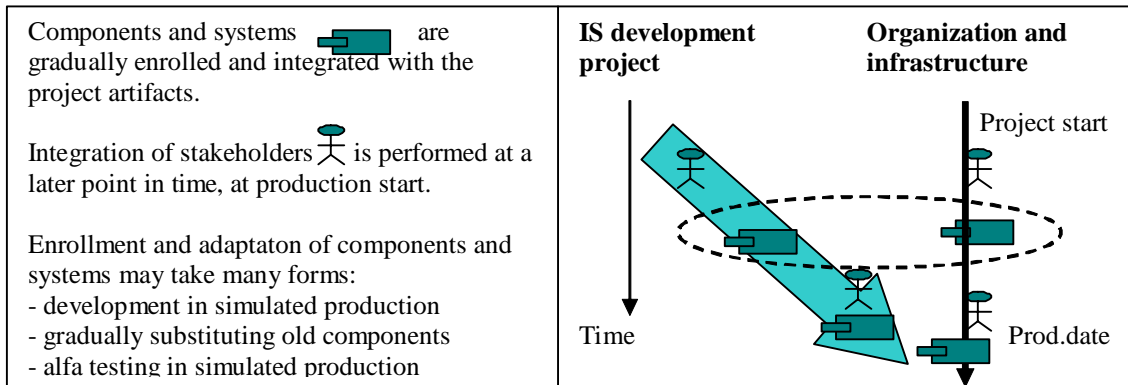


Figure 3. The Technical Integration Pattern

Stabilization is enhanced by the technical nature of the network, but without stakeholder integration there is the risk of friction at the production start date. Late stakeholder integration may lead to poor requirements and user resistance, as described in the Big Bang pattern. Software engineering research has documented that it is easier and cheaper to correct faults as early as possible in software development (Royce, 1998). Supporting mechanisms in RUP are component-based and incremental development (each new component extends the earlier version), which allows for integration and testing in each iteration (Jacobson et al., 1999).

Disadvantages include the aspect that early technical integration increases the complexity of the development process. Testing environments must include more of the production technology, which sometimes is not possible. Also, the logistics of the project becomes more complex, with both sides waiting for other components or other's support.

4.4 The Socio-Technical Integration Pattern

In this pattern (figure 4) there is a dynamic coupling between the IS development project and the business organisation. Innovations in one process may change the other process, and technology is used for early routinization of work practices in the business organisation. This pattern has the

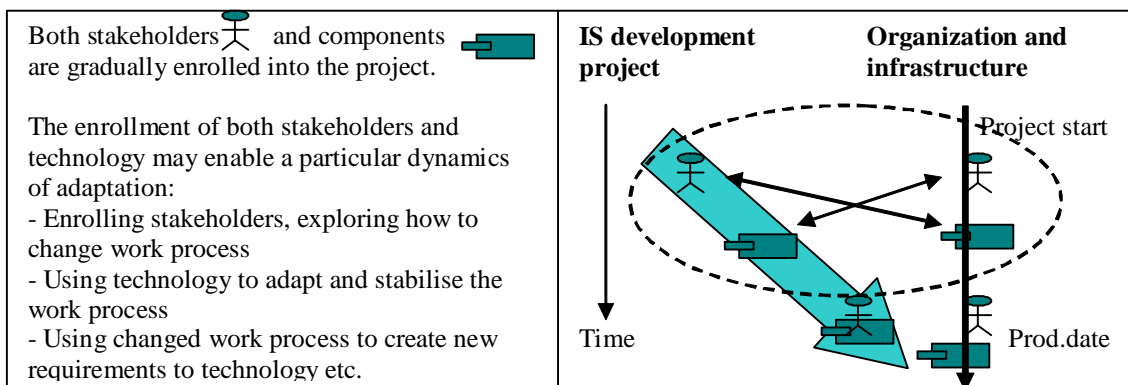


Figure 4. The Socio-Technical Integration Pattern

potential to support organisational innovation through a step-wise and dynamic interaction between stakeholders and technology. Stakeholders are enrolled through arguments of concrete and short-term benefits of the technology, and encouraged to experiment with the software components in the organisational processes. If this is successful the solution will attract new users, who may use the technology in new ways or request more or modified functionality. This may broaden the applicability of the technology, attracting more users, and so on. We may describe this process as self-feeding, where behaviour is stabilized through technology, and the installed base of users and technology grows (Hanseth, 2002). This perspective is also supported by IS implementation research contributions, regarding innovation as the mutual adaptation of organisation and technology (Leonard-Barton, 1988). However, while this pattern greatly reduces the risk of integration problems at the production start date, the disadvantages include huge challenges for project control. It increases the complexity of the development process, making it less predictable. Although RUP to a certain extent supports both stakeholder and technical integration, it contains no explicitly described mechanisms supporting this kind of dynamic.

5 CASE ILLUSTRATIONS OF THE PATTERNS

The four integration patterns are ideal types, and one should not expect to find them in pure forms. Rather, we are interested to explore the forces and challenges within them. In the following we use the three cases to illustrate the four patterns, as shown in table 2. For each case we describe the enrolment and adaptation of stakeholders and technology, and the stabilisation of the socio-technical network. In addition we discuss the managerial challenges faced by the project managers in each case.

Integration pattern	Case organisation	Development project	Managerial challenges
1. Big Bang	University	Learning Management System for supporting learning environment	<i>Stabilizing a top-down solution</i>
2. Stakeholder Integration	Airline	E-business: Marketing and selling tickets on the Internet	<i>Step-wise enrolment of users in an unstable organisation</i>
3. Technical Integration	Airline	E-business: Marketing and selling tickets on the Internet	<i>Step-wise enrolment of components in a large, changing technical infrastructure</i>
4. Socio-Technical Integration	Public auditing agency	Audit process support system for financial auditing	<i>Establishing an organisational mechanism for mutual adaptation</i>

Table 2: Four cases of integration patterns

5.1 An Example of the Big Bang Pattern: A Learning Management System at a University

The University of Oslo (UiO) is Norway's largest and oldest institution of higher education, currently having 32.000 students and 4.600 employees. As with most universities, both external (relevance) and internal (quality) pressures call for a more coherent organisational and technical support for learning. One important response is the establishment of pedagogically motivated integration of ICT in teaching, learning and student collaboration. In Oslo this took the form of implementing a Learning Management System (LMS), of which the main services were course-, content- and student management.

The project was organized and conducted by the central ICT department (USIT), who chose a top-down approach in its work to establish enterprise-wide common infrastructure solutions and services for learning. After a failed pilot project based on generic software (Lotus Notes) and local knowledge (within medicine), USIT decided to go ahead on with a top-down development strategy and a standard LMS package. Building on Classfrontier (a LMS package) and integrating with administrative systems,

the LMS was developed over a period of two years. The main challenge was enrolling scientific staff. Resistance was strong; the faculties and departments were generally hostile both to the use of an LMS and to central IS staff.

In 2001 the solution was set into production in a full-scale implementation: Installing software, educating and supporting users. The LMS was not integrated with existing department based systems, but a new infrastructure was established, tightly integrated with administrative information systems. Resistance was strong in the beginning. One year after the LMS was set into production only a small number of students were actively using the system. The organisational implementation was greatly helped by external pressures. In 2003, the national so-called 'Quality Reform' was implemented at the university. The LMS supported key aspects of the reform, such as net-based learning in a group-supporting environment, and became important for the university management in implementing the reform. During 2003 the LMS stabilized, and became an integral part of the learning process at UiO.

Managerial challenges: Supported by top management, the development project team chose to develop an ambitious and complex solution, and integrate with the university's learning processes at a late stage. This Big Bang strategy had the advantages of concentrating the efforts of the project group on developing a strategically aligned and well-designed solution. On the other hand, the implementation risk of this strategy is clearly illustrated. The organisational implementation was seriously delayed, and without the help of external pressures it is likely to have failed altogether.

5.2 An Example of the Stakeholder Integration Pattern: A RUP Project at SAS

SAS is an international airline carrier based in Scandinavia. In 2001 the SAS group had 25.000 employees, and turnover was 51 billion SEK. SAS's first generation of web marketing had been in use mainly in the Scandinavian market, and based on relatively simple technology. In 2000, acknowledging the commercial potential of web-based booking, SAS decided to establish an electronic marketing channel and e-business. In spring 2001 a project was set up with the following objectives:

- To establish a web-based marketing channel in all important SAS markets
- To provide the international marketing editors with an easy content management interface (CMI) tool to publish campaigns. The solution had to be integrated with the booking systems.

Following earlier practice, a parallel organisational change project was established in the marketing department with an SAS project manager and a user group consisting mainly of Scandinavian marketing editors. The main target group for the system was the international editors, needing a standardized and simple interface. Unfortunately, at this point they had not yet been appointed, and thus could not participate. Instead, the Scandinavian editors were invited to participate.

After two disappointing workshops where the project group failed to translate the editors' work practice into the system use cases, the project manager from the marketing division became a part of the development project, and very successfully prototyped the main part of the software product. In a way he changed sides, and later on became the project manager of the whole project. Late in the project the international marketing editors were appointed, and were sent to Stockholm to learn the new system. This led to a large number of change requests, and a rush to get the system finished on time, abandoning the iterative RUP structure. Thus, stabilisation was achieved by improvisation at a late stage of the project.

Managerial challenges: Step-wise integration of stakeholders certainly demands open communications with users and fora for discussing requirements. But as the case shows, this is not enough. Real enrolment depends on a successful translation of needs, mutually adapted to the work practices. When this was not achieved in the first iterations - for reasons not controlled by the project - the project responded through a certain "encapsulation", i.e. concentrating on the internal tasks of the development project, postponing the enrolment of the external actors. The price for this postponement was paid later in the project, when stakeholders had to be integrated by improvisation in the last iteration.

5.3 An Example of the Technical Integration Pattern

The SAS case is also an interesting example of the technical integration pattern. In accordance with RUP, it was planned with a step-wise integration of software components and systems. The technical environment was a complex one. The CMI was developed as an integrated part of the other e-business platform-related projects, and it also had to be integrated with the existing information infrastructure; the legacy booking systems and the international Amadeus booking system. All the projects were component-based, and the publishing solution was specified in UML and programmed in JavaScripts and Active Server Pages, using Vignette's application program interfaces. Through five iterations a large number of internal and external software components and systems were linked into the project.

Although the project was planned with step-wise technical integration, external component integration was concentrated on the last two, and in particular, the last iteration. This is not according to RUP, and it was not intended. The reasons for the postponement were outside of the control of the project manager. First, the external components were delayed, blocking the integration of the product. Second, the production environments were too large and complex to be installed, and had to be partly simulated. Adding to this, the technical infrastructure was changing during the project, making configuration management difficult. The technical solution was stabilized in the last iteration. This was partly achieved by improvisation, and not iterative planning. Cooperating with the experienced SAS data centre, the project group succeeded in setting the system into stable production in May 2002.

Managerial challenges: Step-wise technical integration is a beneficial way to reduce risk, but it comes at a price. It increases the complexity of the development process. Examples of this in the SAS project were a risky dependence on external software components, and a complex and changing technical infrastructure. This situation created much friction in the development process, and thus also considerable pressure on the project manager. In parallel with the Stakeholder Integration pattern, in such situations the development project may choose to respond with project encapsulation; concentrating on internal development issues. This will increase internal project control, but also the risk of implementation failure.

5.4 An Example of the Socio-Technical Integration Pattern: Development of an Audit Support System

The Office of the Auditor General (OAG) is instituted by the Norwegian Constitution. Its main tasks are to audit the central government, ministries and their agencies' accounts. The OAG is based in Oslo, where most of the 450 employees are located. Auditors perform three types of audit: financial audit, corporate control, and performance audit. The system developed in this case study supports the financial auditing process.

The PROSIT Project (Process-Oriented IT Audit Support)

After a feasibility study had assessed and rejected five commercial audit support systems, it was decided in 1999 to develop a tailor-made system. The main objective for the project was to standardize the auditing process through a modern information system. The project was planned and organized within the Microsoft Solutions Framework, MSF (Microsoft 2004). Like RUP, MSF is an iterative and incremental process framework focusing on step-wise integration.

The project organisation was designed to align the organisation and the development, with a strong management and quality focus. Large resources were allocated to secure a step-wise integration. Through five iterations a number of stakeholders and software were gradually enrolled into the project. First, a strategy of broad user involvement and long workshops in the first two iterations failed to stabilize a version. Then, in the third iteration, an organisational mechanism for mutual adaptation between the development process and the organisational processes was introduced. This was the Audit Methodology Group, which had been given two mandates; to provide requirements input to the PROSIT project, and to change the auditing process. Seeing that the two mandates reinforced each other, they were able to use the learning from the PROSIT project to restructure the auditing process,

and also structure the PROSIT software system according to the new auditing process. The organisational implementation of the resulting information system was done step-wise. After redesigning the auditing process, all departments were asked to plan their own PROSIT implementation. This was successfully done during the fourth and fifth iteration.

Managerial challenges: The PROSIT project illustrates that the Socio-Technical Integration pattern has the great advantage of coupling stakeholders and software components dynamically. It allows for organisational experimentation. An important lesson learned is that large user workshops were unsuccessful, while a small, mediating group with a clear mandate including both the development project and the auditing process was very successful. The complexity of the integration strategy of the project was considerable. It was well supported by MSF, and by the organisational construct of the Methodology Group.

6 IMPLICATIONS

6.1 Implications for Practice

For practitioners, this paper has two important findings. First, four generic integration patterns were identified. The patterns and their illustrations show that there are both advantages and disadvantages with each pattern. They also show that support for integration in software engineering frameworks is varying; in RUP (and MSF) there is support for both stakeholder integration and technical integration, but not for socio-technical integration. Second, as the four examples show, integration comes neither free nor easily. In the Big Bang pattern the internal development project risk is low, but organisational implementation risk is high. In the Stakeholder Integration and Technical Integration patterns, the internal development project risk increases, while there is a lower implementation risk. The Socio-Technical Integration pattern lowers the organisational implementation risk, but increases greatly the complexity of the development project.

The four integration patterns together represent an analytical tool for planning iterative software development projects. Choosing a pattern will help the practitioner to analyze the forces within it, in order to handle some of the risks of integration. However, care should be taken to evaluate the application area of the patterns. Selecting the right integration pattern requires sensitivity to context. While the Socio-Technical Integration pattern may appear most attractive, it might be dangerous to use it in a setting of strong time pressure. As the SAS case shows, the Stakeholder Integration and the Technical Integration patterns require careful managerial trade-offs, balancing project complexity against project encapsulation; to reduce complexity, the project manager may choose to concentrate on internal project tasks, postponing external integration issues. Sometimes the Big Bang pattern may prove beneficial in situations where stepwise integration is simply not feasible. But an organisation choosing this pattern should also be aware of the organisational implementation risk. In table 3 we summarize the selection criteria.

Integration pattern	Suitable context	Managerial trade-off
Big Bang	Step-wise integration not feasible	Prioritizing project control, with the risk of user resistance and integration problems at start-up
Stakeholder Integration	Simple technical solution in a complex organisational env.ment	Balancing stakeholder complexity against project encapsulation
Technical Integration	Technically complex solution in a stable organisational setting	Balancing technical complexity against project encapsulation
Socio-Technical Integration	Organisational and technical flexibility	Allowing process dynamics, but at the risk of losing project control

Table 3. Integration pattern selection criteria

6.2 Implications for Research

For IS research the patterns represent an extension to the body of knowledge that addresses the dynamics of socio-technical integration (Kling and Scacchi 1982; Coakes et al., 2000; Hanseth, 2002). This research stream emphasizes that a new information system must *link into* an existing socio-technical web of mutually dependent elements that cooperate to create value. The four patterns describe the managerial trade-offs for achieving these links. As illustrated in the SAS case, this linking is non-trivial. Enrolment of stakeholders and technology from the surrounding networks is often outside the control of the project manager, and has to take place in windows of opportunity. Managing this very situated and context dependent challenge is grossly underrated in the project management literature. We also aim to bridge a gap between the information systems and software engineering communities. Acknowledging the complex non-linear and political nature of integration, the patterns incorporate existing knowledge in IS implementation research. But at the same time, the patterns address the practical challenges inherent in software engineering, relating directly to modern iterative process frameworks like RUP.

6.3 Limitations

From an SE perspective one may criticize the four patterns for expanding the object of study too much. While acknowledging that the integration of the information system and the business organisation is important, this cannot be a realistic objective at a project level. Few professions have such complex challenges to cope with as the IS project manager (Royce, 1998). Perhaps, the project manager should not be burdened with the integration challenge. Somebody who has the necessary authority and business knowledge should handle this at a higher level. Sauer and Willcocks (2004) support this when they suggest an organisational architect to be responsible for top-down integration.

While we agree that the integration challenge calls for complex interventions we do not think it can be solved at top management level. The emergent nature of integration makes it an ongoing challenge which the IS project manager must address, whether he feels competent or not.

From the perspective of information infrastructure (Ciborra, 2000), the four patterns may be criticized for being too optimistic on the question of control. According to Ciborra, IS projects tend to drift, i.e., to deviate from plans. Trying to control this often leads to side effects, which make control even harder. Thus, from this perspective we cannot choose an integration strategy at will. Instead, we should build on existing structures and practices. We agree that integration is hard to plan, and we have earlier in the paper argued that the patterns are context dependent. However, the integration patterns are not planning templates, ready to be used in IS development projects. They are primarily analytical constructs aimed at understanding the dynamics of integration, and thus helping the project manager to assess the risks of integration.

7 CONCLUSION

This paper started out by asking: how should the IS development project manage the integration between the information system and the business organisation? The theoretical approach was to view integration as the emergence of a socio-technical network.

The main finding is the suggested typology of the four generic integration patterns; Big Bang, Stakeholder Integration, Technical Integration, and Socio-Technical Integration. The patterns represent ideal types for how, and when, a development project should enrol and adapt stakeholders and technology, and stabilize the behaviour of the network. For IS research the four integration patterns contribute to a socio-technical vocabulary of integration. Since integration is an issue for both SE and IS research, the four patterns also seek to bridge the gap between the two disciplines.

Second, careful analysis of the patterns in the IS and SE research, and in the three cases, shows that integration always comes at a price. Step-wise integration lowers the implementation risk, but increases greatly the complexity of the project. Thus, for practitioners, the four patterns represent a tool to analyze some of the risks of integration, and to assess the managerial trade-offs.

The patterns are context dependent. They were elicited and studied in the context of iterative systems development where the success of the projects was associated with the integration with the organisational processes. Further research should study more closely how managers achieve successful integration. It could also test the validity of the patterns in other projects, especially where the context is not software development, but adapting and implementing business software packages.

8 REFERENCES

- Alter, S. (2000) Same Words, Different Meanings: Are Basic IS/IT Concepts Our Self-Imposed Tower of Babel? *Communications of the AIS*, 3(10), 2-87.
- Ambler, S. W. (1998) *Process Patterns*. Cambridge University Press.
- Boehm, B. W. (1988) A Spiral Model of Software Development and Enhancement, *IEEE Computer*, 21(5), 61-72.
- Ciborra, C. (2000) *From Control to Drift*. Oxford University Press.
- Coakes, E, Willis, D., and Lloyd-Jones, R. (2000) *The New SocioTech. Graffiti on the Long Wall*. London: Springer.
- Hanseth, O. (2002) From systems and tools to networks and infrastructures. Toward a theory of ICT solutions and its design methodology implications, http://heim.ifi.uio.no/~oleha/Publications/ib_ISR_3rd_resubm2.html
- Hanseth, O. and Monteiro, E. (1996) Inscribing Behaviour in Information Infrastructure Standards," *Accounting, Management and Information Systems*, 7(4), 183-211.
- Jacobson, I., Booch, G. and Rumbaugh, R. (1999) *The Unified Software Development Process*, Addison Wesley, Reading.
- Kappelman, L. A. and McLean, E. R. (1994) User Engagement in the Development, Implementation and Use of Information Technologies, in *Proceedings of HICSS 27*, 512-521.
- Klein, H. K. and Myers, M. D. (1999) A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems, *MIS Quarterly*, 23(1), 67-94.
- Kling, R. and Scacchi, W. (1982) The Web of Computing: Computer Technology as Social Organization, in *Advances in Computers*, 21,1-90.
- Lamb, R. and Kling, R. (2003) Reconceptualizing Users as Social Actors in Informations Systems Research, *MIS Quarterly*, 27(2), 197-235.
- Leonard-Barton, D. (1988) Implementation as Mutual Adaptation of Technology and Organization, *Research Policy*, 17(5), 251-267.
- Lloyd, A. D., Dewar, R., and Pooley, R. (1999) Legacy Information Systems and Business Process Change: A Patterns Perspective, *Communications of the AIS*, 2(1).
- Majchrzak, A., Rice, R., Malhotra, A., King, N., and Ba, S. (2000). Technology adaptation: the case of a computer-supported inter-organizational virtual team, *MIS Quarterly*, 24(4), 569-600.
- Microsoft (2004) Microsoft Solutions Framework. <http://www.microsoft.com/technet/itsolutions/techguide/msf/default.msp>. Accessed May 2004.
- Mumford, E. (1995) *Effective Systems Design and Requirements Analysis: The ETHICS Approach*, Macmillan, Basingstoke, England.
- Newman, M. and Robey, D. (1992) A social process model of user-analyst relationships, *MIS Quarterly* 16(2), 249-266.
- Pettigrew, A. M. (1985) Contextual Research and the Study of Organizational Change Processes, in Mumford, E. (ed.), *Research Methods in Information Systems*, North-Holland.
- Royce, W. (1998) *Software Project Management*, Addison-Wesley Longman, Reading, Mass.
- Sauer, C. and Willcocks, L. P. (2004) Strategic Alignment Revisited: Connecting Organizational Architecture and IT Infrastructure, in *Proceedings of HICSS 37*, Big Island, Hawaii.