December 2005

Web Services System Development: A Grounded Theory Study

Maddalena Sorrentino
State University of Milano

Francesco Virili
State University of Cassino

18th Bled eConference

eIntegration in Action

Bled, Slovenia, June 6 - 8, 2005

Web Services System Development: a Grounded Theory Study

Maddalena Sorrentino

State University of Milano, Italy
maddalena.sorrentino@unimi.it

Francesco Virili

State University of Cassino, Italy
francesco.virili@eco.unicas.it

Abstract

This study in progress presents a grounded theory analysis of a case study in the banking industry with a view to showing the role of "Web services" technology in information systems development practices. The case study relates to the implementation in the Central Europe Bank (a pseudonym) of a new software application based on Web services technology. In particular, the focus is on the following research question: what are the peculiarities of the Web services software development process? A tentative answer to the question is advanced here in the form of a preliminary formulation of a descriptive theory of the Web services ISD process. In particular, an effort is made to understand whether the Web services ISD process can be considered as a short-cycle development process (Baskerville and Pries-Heje, 2004). The process under observation might be categorised in a preliminary way as a "hybrid" one lying between methodical and amethodical development processes. Indeed, the data collected and analysed so far confirm the complexity and richness of the situation taken into consideration, offering useful insights to complete and further extend both theoretical and empirical analysis.

1. Introduction

Web projects "at Internet time" are often carried out in ways different to those characterising traditional Information Systems development projects (Baskerville and Pries-Heje, 2001; 2002; Baskerville et al., 2003). These differences, however, are still far from being well understood (Baskerville et al., 2002). The first studies on the subject

produced different reactions. For example, two divergent views on Web development were reported in Glass (2003). One maintains that the "golden rules" of classical Information Systems Development (ISD hereafter) still hold for Web systems development: there is nothing really new in Web work, nothing that has not already been experienced in the past (Kautz and Nørbjerg 2003). The opposite position holds that building Web systems may constitute a new and quite different kind of process, still "in the making" and partially unknown (Baskerville and Pries-Heje, 2002).

This paper arises out of a study conducted on the implementation of a new software application in a Swiss bank. We analyse empirically an actual case of Web services software development and seek to answer the following research question: *what are the peculiarities of the Web services software development process?* We would like to offer an answer to the question in the form of a preliminary formulation of a descriptive theory of the Web services ISD process. In particular, we hope to establish an understanding of whether the Web services ISD process under examination can be considered as a short-cycle development process (Baskerville and Pries-Heje, 2004).

The paper is organised as follows: Section 2 describes the research methodology adopted. Section 3 is dedicated to the Grounded Theory analysis of the case under observation. This is discussed in Section 4, where it is compared and contrasted with other recent research findings. The paper concludes in Section 5 The paper concludes in Section 6 outlining implications for researchers and practitioners, together with occasions for further research.

2. Research Methodology

Given the exploratory nature of the research question, we opted for an in-depth case study approach (Yin, 1994). Case studies can achieve a holistic understanding of cultural systems of action (Feagin, Orum, & Sjoberg, 1990; Tellis, 1997). Moreover, they facilitate multi-perspectival analyses that consider not just the voices and perspectives of individual actors but also those of important groups of actors in addition to the interaction between the various actors (Tellis, 1997). A relationship having been established with the Central Europe Bank, data was collected by way of semi-structured interviews and document reviews. A qualitative research approach of this kind is especially appropriate for the study of complex, dynamic social phenomena that are 'both context and time dependent' (Orlikowski and Baroudi, 1991).

2.1 Site Selection

Given the limited spread of Web services technology in the banking sector at the time the research was first embarked upon, gaining access to a fully-fledged Web services development project proved quite difficult. At the time, a major Swiss bank (here referred to as the "Central Europe Bank") was engaging in an important initiative to redesign a part of its front-office application portfolio using Web Services technology (Virili and Sorrentino, 2003).

The new software application (Cashier Management Systems - CMS hereafter) was adopted to automate the bank's retail banking activities. Because its primary business was private banking (i.e. banking services for affluent individuals, including investment and portfolio management), the bank did not consider retail banking strategically important. Consequently, this latter activity was considered an ideal "test-bed" to introduce a new technology in a real setting, in that, even in case of failure, the bank's core-business

activities would have not been affected. Nevertheless, reliability, security and quality of service were still primary requisites of the new software solution.

The Central Europe Bank's information system is managed internally by the IT Division. The development of new retail banking applications was partially outsourced to an external software developer specialising in financial and retail banking systems. The project team included both the bank's and the provider's specialists.

2.2 Data Collection

The investigation was set up as an exploratory case-study. The interviews took the form of a series of open-ended questions based largely on the questionnaire developed by Baskerville and Pries-Heje (2002). Some adaptation was effected with a view to better focusing on the particular subject under consideration. The interview was structured in terms of the following 5 headings:

1. information on the interviewee and the organization;
2. software development methods and tools;
3. software applications;
4. teams and people;
5. problems and challenges.

Interviews and field observations were supplemented with internal documents - mainly technical literature - made available by the bank and by external sources. In the first round, carried out between September 2003 and February 2004, a total number of 11 interviews were conducted. On average, each interview lasted about two hours. Two people were interviewed twice. The team leader was present at all the interviews and actively intervened when necessary.

Five of the nine interviewees were bank employees while four worked for the software company. Here follows the list of the interviewees.

From the bank:

- the IT executive;
- 2 project managers/functional analysts;
- 2 IS architects and integration managers.

From the software company:

- 3 components of the project team;
- the team leader.

The two researchers conducted together the interviews at the bank site. One of the researchers took notes of the interviews using a laptop computer while the other took hand-written notes. Later, the notes taken by the two researchers were re-examined and compared. The final text of each interview was generated after comparison and integration of the two different sources.

2.3 Data Analysis

The data generated by the interviews was analysed and coded according to the Grounded Theory Methodology, following in particular the detailed directions given in Strauss and Corbin (1990). Grounded Theory analysis is essentially based on two activities: *open coding* (the identification and labelling of particular concepts in text passages) and *axial coding* (the definition of the relationships between concepts). This process was carried out with the help of a well-known software application for text analysis (Atlas.ti, rel. 4.1 and rel. 5.0).

In an initial phase, after transcription and comparison, all the interviews were printed out and carefully examined to make sense of the general content and meaning of the texts. Then, in order to generate some initial concepts - together with their specific properties and dimensions - and to discover the basic relationships between them, a detailed analysis was carried out. To this end, the two researchers analysed and discussed line by line and even word by word a first batch of interviews, taking notes and writing memos along the lines suggested by Strauss and Corbin (1990) (see, in particular, chapter 5, "Analysis Through Microscopic Examination of Data").

The main categories (i.e. families of concepts) generated were:

1) The Web services technology and its properties;
2) The software development process and its main phases/activities;
3) The context: bank organization and its environment, project team and application software peculiarities.

After about 13 months from the first interviews, open and axial coding had generated a list of 298 concepts linked by relationships in 13 different networks.

The final outcome of both data collection and data analysis was finally validated together with the software company team leader.

In the rest of the paper some outcomes from this empirical study are presented.

3. Grounded Theory Analysis

The outcome of the Grounded Theory analysis is essentially constituted by a list of concepts grouped into categories and subcategories, with multiple hierarchical levels. During axial coding, categories and concepts are linked by relationships into several networks of concepts. The core categories generated through the empirical analysis of the European Central Bank were:

1) The Web services technology and its properties;
2) The software development process and its main phases/activities;
3) The context: bank organization and its environment, project team and application software peculiarities.

In order to offer a thorough insight into Web services development as experienced in the ECB case, the role played by all three categories was investigated in detail. Indeed, besides the three networks of concepts related to the core categories, the researchers built several additional networks, along with their mutual relationships.

Given space limits, only two of the three core categories will be (partially) discussed in this paper, i.e. Web services technology and software development process. Moreover,

within these attention will be directed only to the most important concepts. The importance of concepts is calculated in terms of so-called "groundedness", i.e. the number of citations (the number of times a concept was mentioned during interviews) and in terms of so-called "density", i.e. the number of links to other concepts. While no detailed analysis of the category context is provided, a few considerations about contextual factors will be offered in section 4.

3.1 Properties of the "Web Services (WS) Technology"

Figure 1 represents what the participants in the Central Europe Bank CMS development process perceived as some of the key characteristics of the Web services technology adopted in the project.

The graphic components represent the "Web services technology" in terms of concepts, properties and relations. Concepts, represented by rectangles, are organized in a hierarchy signalled by means of the "is a" relationship. Rectangles may also identify properties. Though both represented here by rectangles, concepts and properties are different in nature and should not be confused. Properties are attributes belonging to a concept. For example, "functional decomposition" is a property of "Components". A property at a higher hierarchical level is inherited by all the child member concepts: for example "functional decomposition" belongs to the whole "Components" family (WS, DCOM and CORBA). Properties are linked to pertaining concepts by the "is property of" relationship.

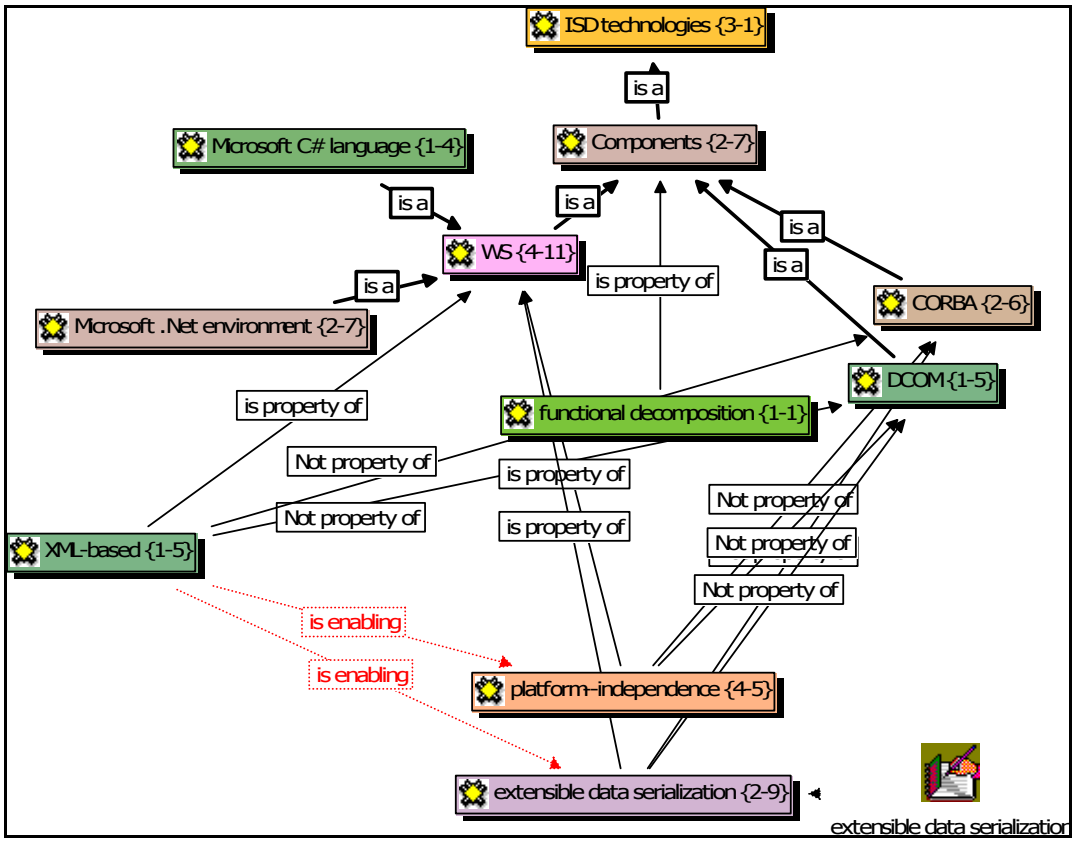

Figure 1: Web services technology: key characteristics in terms of concepts, properties and relations.

In Figure 1 Web services technology shows two peculiar properties: "platform independence" and "extensible data serialization"; the "is enabling" relationship indicates the enabling role of the property "XML-based" for the properties of "platform independence" and of "functional decomposition": The role of these features for WS-enabled ISD practices will become evident in the discussion of Figure 2 below.

3.2 Enabling Effects of "WS Technology": General View

In Figure 2 a first general view is given of the enabling effects that result from the fact that WS is XML-based. In particular, the upper part of the figure shows once again that the property "XML-based" enables "platform independence" and "extensible data serialization". In order to appreciate the role that these two factors play in the ISD process, the process as a whole may be divided conceptually - in accordance with mainstream Software Engineering literature (see, for example, Sommerville 1982, chapter 3, 6th ed. 2000) - into two main categories/macrophases: "system analysis and design" and "implementation, testing and operation".

As Figure 2 shows, "platform independence" is mainly associated with system implementation, testing and operation (see Figure 4 and accompanying discussion) while "extensible data serialisation" has an enabling effect in relation to "incremental implementation of the contract", "simplified versioning" and "reusability".

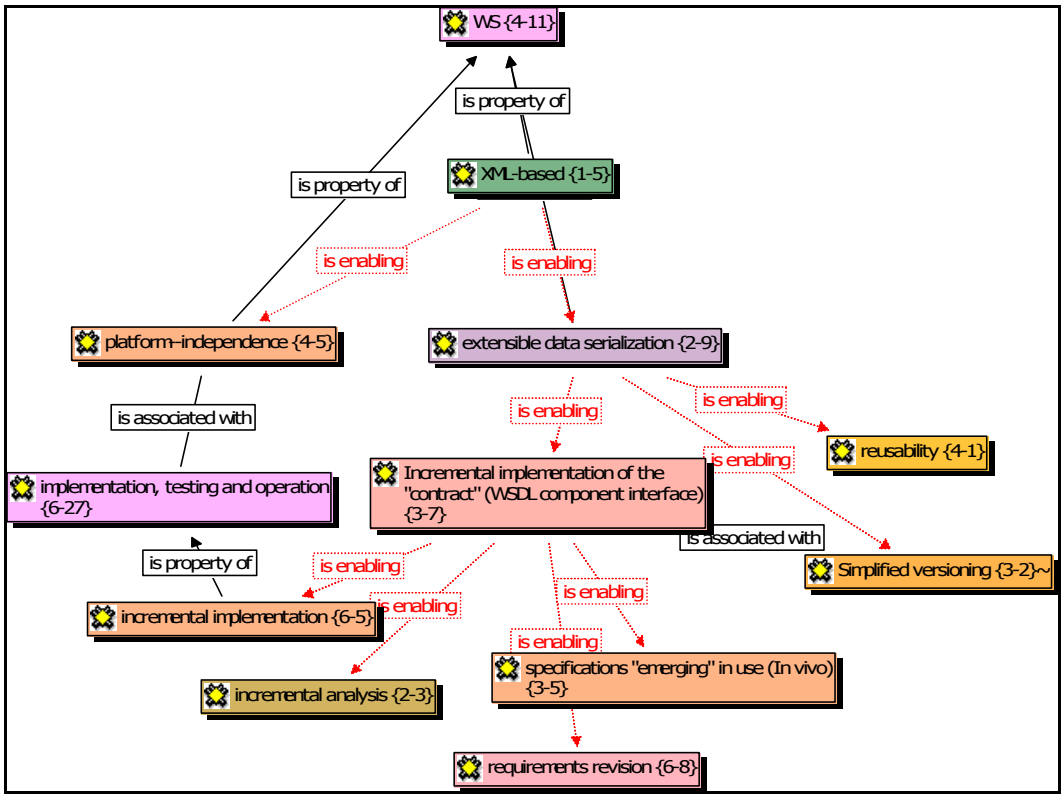

Figure 2: *The enabling role of Web services in the ISD process: how the key property "XML-based" affected other properties and activities in the CMS project.*

The key property of Figure 2, which is visible in the middle of the diagram, is designated "Incremental implementation of the contract (WSDL component interface)". What is a software contract? Software contracts play an important role in communicating to

potential consumers (e.g. people using a software component) the functional properties of the component itself (Crnkovic et al., 2002, p. 36). In particular, "a contract lists the global constraints that the component will maintain (the invariant). For each operation within the component, a contract also lists the constraints that must be met by the client (the pre-condition), and that the component promises to establish in return (the post-condition). The pre-condition, the invariant and the post-conditions constitute the specification of a component's behaviour" (Crnkovic et al., 2002, p. 37).

The possibility offered by WS technology to change a software contract is a fundamental enabler, as discussed in sections 3.3 and 3.4: System Analysis and Design is affected by incremental analysis, requirements revision, specifications "emerging" in use (Figure 3); system implementation is affected by "incremental implementation" (Figure 4).

3.3 How "WS Technology" Affected "System Analysis and Design"

How was the system analysis and design affected by WS technology? Figure 3 shows the enabling effects of the feature "Incremental implementation of the contract". The GT analysis revealed how, among the numerous properties of analysis and design referred to by the interviewees, the most prominent one (both in terms of groundedness: 6 and density: 8) was "requirements revision" (Figure 3, top left). The possibility of revising and changing software requirements originally formulated during system analysis was explicitly mentioned 6 times by 5 different interviewees (i.e. by everyone in the team except the system architect and one junior developer who had lower visibility on system analysis during the project).

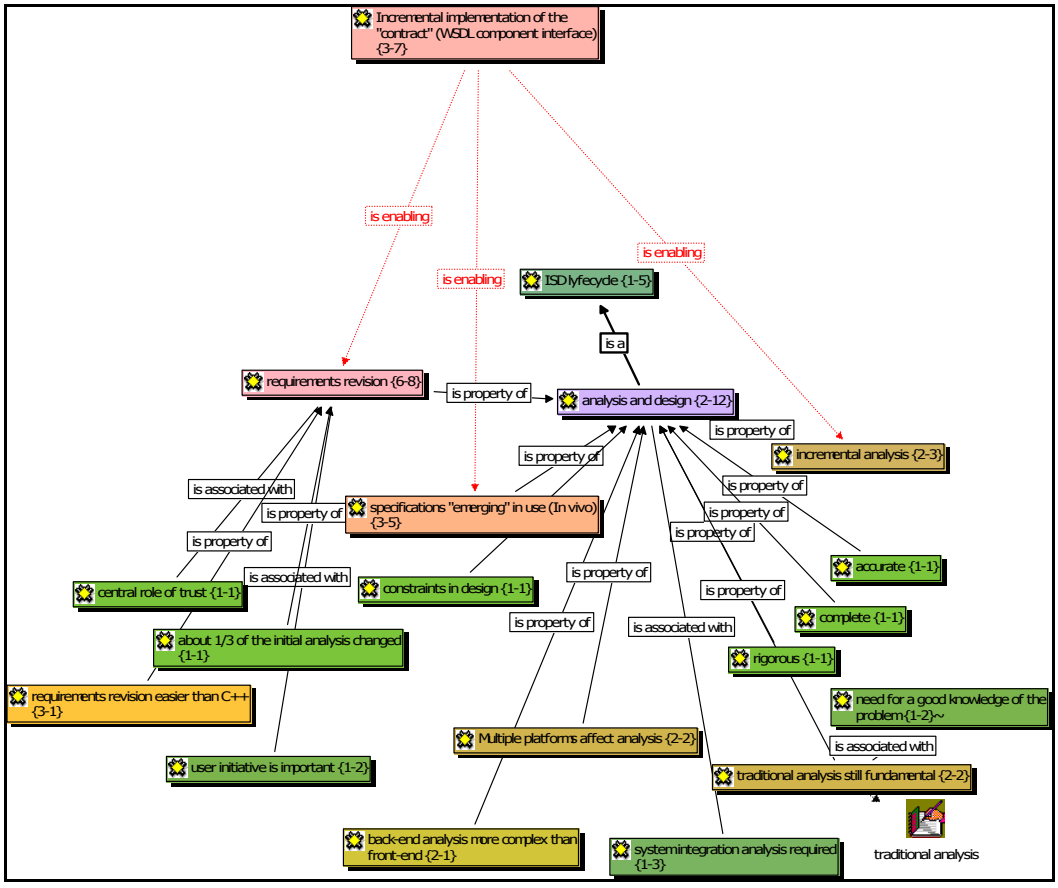

Figure 3: The enabling role of Web services in the ISD process: how it affected system analysis and design in the CMS project.

Recall from Figures 1 and 2 that this is due to extensible serialization, an exclusive property of Web services in the components family. Some of the interviewees had previous experiences with DCOM and CORBA and explicitly recognized the difference.

Not only ”requirements revision” (top left side of Figure 3), but also new “specifications emerging in use” (just below in Figure 3) are enabled by Web services, as explicitly stated 3 times; moreover, 2 interviewees declared that, by using Web services, system analysis, though still detailed and accurate, is somewhat less deep than before and is frequently updated during development iterations (incremental analysis): “The phases are the same, but depth and openness have changed” (one interviewee, functional analyst).

3.4 How “WS technology” Affected “System Implementation, Testing and Operation”

Figure 4 shows the ISD macrophase of “Implementation, testing and operation” and the enabling role of the WS technology in it.

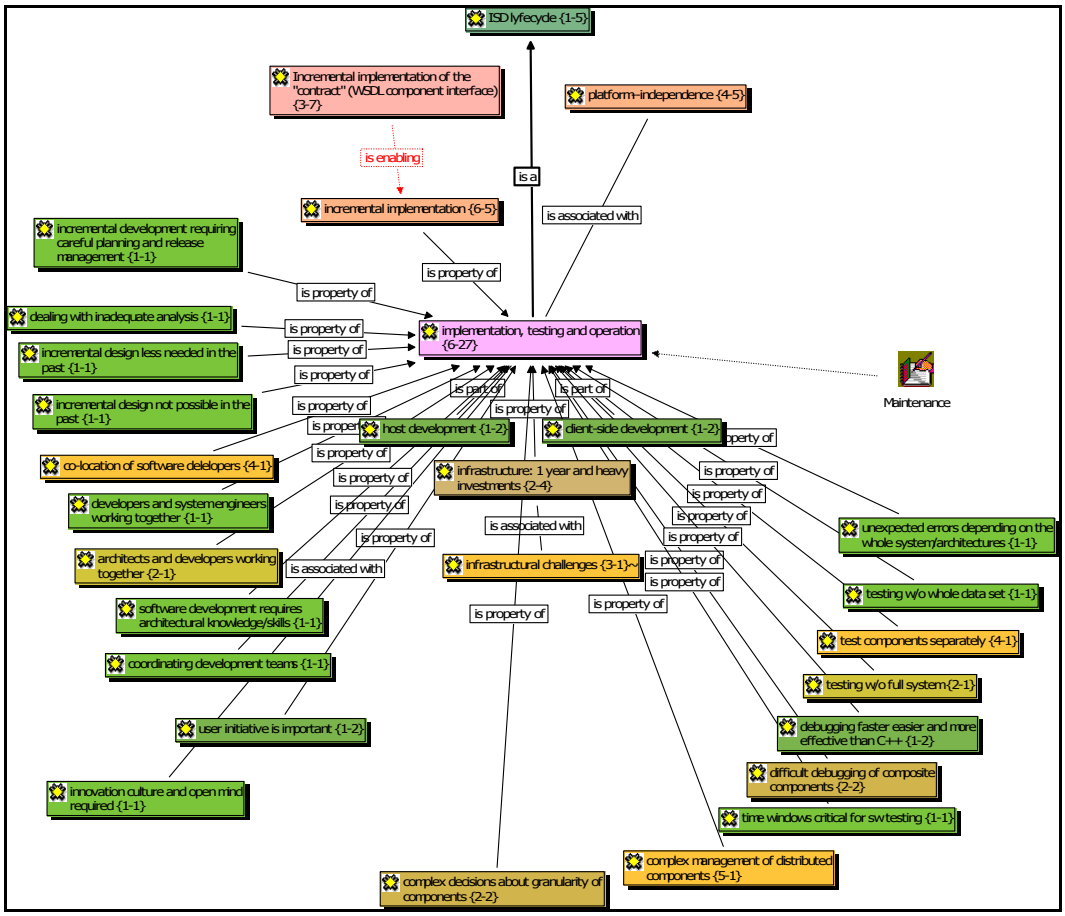

Figure 4: The enabling role of Web services in the ISD process: how it affected system implementation, testing and operation in the CMS project.

Once again, the WS feature “Incremental implementation of the contract” (see top left) plays an important enabling role, opening the way to incremental development. In the context represented by Figure 4, incremental development, which was mentioned by all the software developers and by one functional analyst, constituted the most prominent

concept (groundedness: 6, density: 5). In incremental development, software coding is based on small incremental additions, and it is achieved by working in collaboration with architects and system engineers, paying attention to user needs, cultivating and stimulating user initiatives all within a culture of innovation and open mindedness. Host development is significantly different from (and more complex than) client-side development. The latter is characterised by easier development of the single software components, which are quite independent from each other and can be developed and tested separately. On the down side, the overall management of distributed components, including their orchestration and tracing, poses new problems, including the need to handle a higher level of infrastructural complexity. According to one functional analyst, incremental development was not as viable in the past as it is today. Most of these aspects are associated with the property of "platform independence", represented in the top right of the figure.

To sum up, the Grounded Theory analysis of the CMS development project at the Central Europe Bank shows that Web services technology may act as a key technological enabler for more agile forms of IS development, even under the Central Europe Bank's stringent quality standards. In facts, the research suggests that Web services-based system development may offer significant opportunities in terms of incremental analysis, requirements revision, requirements emerging in use and incremental implementation. This could make Web services, as opposed to other component-based technologies like DCOM or CORBA, an ideal candidate for continuous redevelopment approaches (Truex, Baskerville and Klein, 1999). One question naturally arises here: are we dealing with a new class of software development practices? This topic will be addressed in the following section.

4. Discussion: Web Services System Development Practices

Does Web services development constitute a new class of software development practices? This issue may be explored by viewing our findings in the light of the so-called "short cycle system development" practices as illustrated by Baskerville and Pries-Heje (2004) in an extensive analysis of several web-based ISD projects in the US and Denmark. The relevance of such a comparison stems from the fact that, as the authors suggest, some of these emergent practices may be assuming a significant role in current IS development projects: "*We do not currently understand the essential characteristics of short cycle systems development that are enduring in actual IS practice. The field appears to be in a state of transition. It is a transition from a period where furious change engulfed a new sector of the systems development community (Internet speed software). It is a transition into a period where certain techniques of short cycle time development are likely to stabilize and endure*." (Baskerville and Pries-Heje, 2004, p. 239).

Figure 5 depicts the short cycle development process. It is notable how it is strongly conditioned by a market environment where time pressure is intense. Another important environmental determinant is lack of experience, something that is natural in the presence of technical novelty. A consequence of this is that software quality is less important than completion speed and delivery time. The development process is characterised by agility, prototyping, parallelisation and release orientation. In this scenario, architecture and infrastructure inevitably play a critical role.

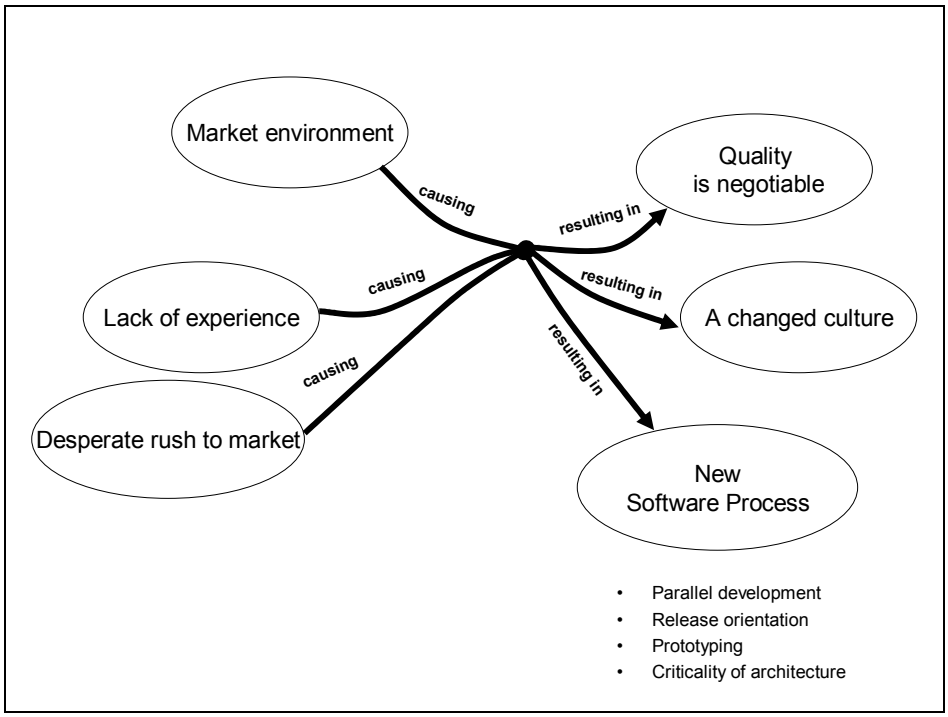

Figure 5: Short cycle system development process. Adapted from (Baskerville and Pries-Heje, 2004), Figure 2 and Table 5.

Figure 6 represents the Central Europe Bank Cashier Management System development process, clearly revealing the presence within it of the main conceptual categories of the short cycle system development process together with its essential causal structure. In addition, Figure 6 shows the enabling role that is played by Web services technology (see Section 3 above).

The inclusion of the "Web services" conceptual category is underlined by the use of a dotted circle and a dotted link. The causal link is here designated by the label "influencing" (as opposed to "causing"). It underlines how Web services played a central role as enabling technology, disclosing new opportunities and affecting in a significant way the spectrum of choice and action of the participants in the whole process, as previously shown in Section 3. The colour grey highlights those categories and concepts that, in view of the findings reported by (Baskerville and Pries-Heje, 2004) and shown above in Figure 5, constituted distinct features in the Central Europe Bank case.

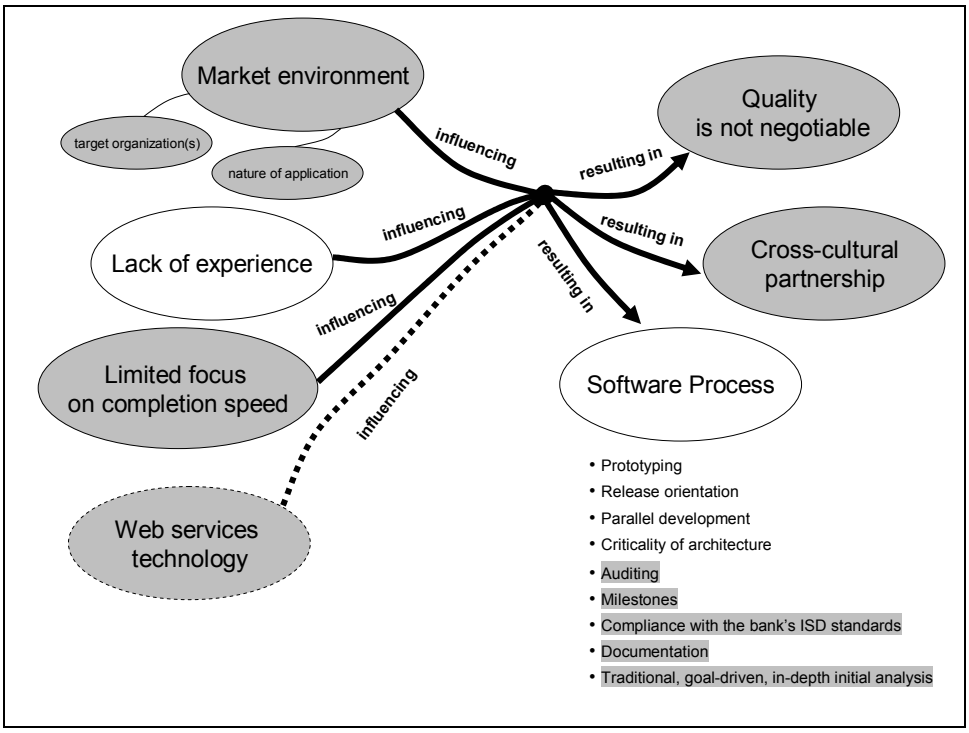

Figure 6: The Central Europe Bank Cashier Management System development process.

The contrasts between the Baskerville and Pries-Heje's model and the present case study were as follows:

Market environment: the characteristics of the target organisations and the nature of the applications were different. In the one case the market was a rather generic one and the applications were not mission critical. In the bank case, by contrast, what was at issue was a banking application characterised by marked reliability and security requirements and destined for a private banking context.

Changed culture: in the one case the software developers' organizational culture was characterized by informality, innovation and often undisciplined creativity. In the bank case there took place an encounter between two distinct and to some extent opposed cultures: that of the bank, formal, disciplined and rigorous and that of the external software developer, informal, racy and creative. Despite differences in backgrounds, skill sets and experience levels, though, both groups sought to proactively solve problems.

Completion speed: in the one case the emphasis was on the "rush to market", i.e. the need to rapidly introduce new products and services into the market. In the bank case, by contrast, the time factor did not have a high priority. Security and reliability of software applications were the most important factors.

Quality: in the one case quality was of secondary importance in respect of the "rush to market" factor. In the bank case, quality was not negotiable.

ISD practices: So far as the bank case is concerned, one can notice the presence of the four essential practices observed by Baskerville and Pries-Heje in short cycle development and listed above in Figure 5 (i.e. Prototyping, Release orientation, Criticality of architecture and Parallel development). In addition, however, highlighted in grey, there appear five additional practices:

- auditing,

- milestones,
- compliance with the bank's ISD standards,
- documentation,
- traditional goal-driven, in-depth initial analysis.

These five practices are usually adopted in traditional ISD contexts.

Figure 7 shows that the Central Europe Bank software development process observed here does not fit precisely into either of the two categories, i.e. methodical and amethodical processes (Truex, Baskerville and Travis 2000).

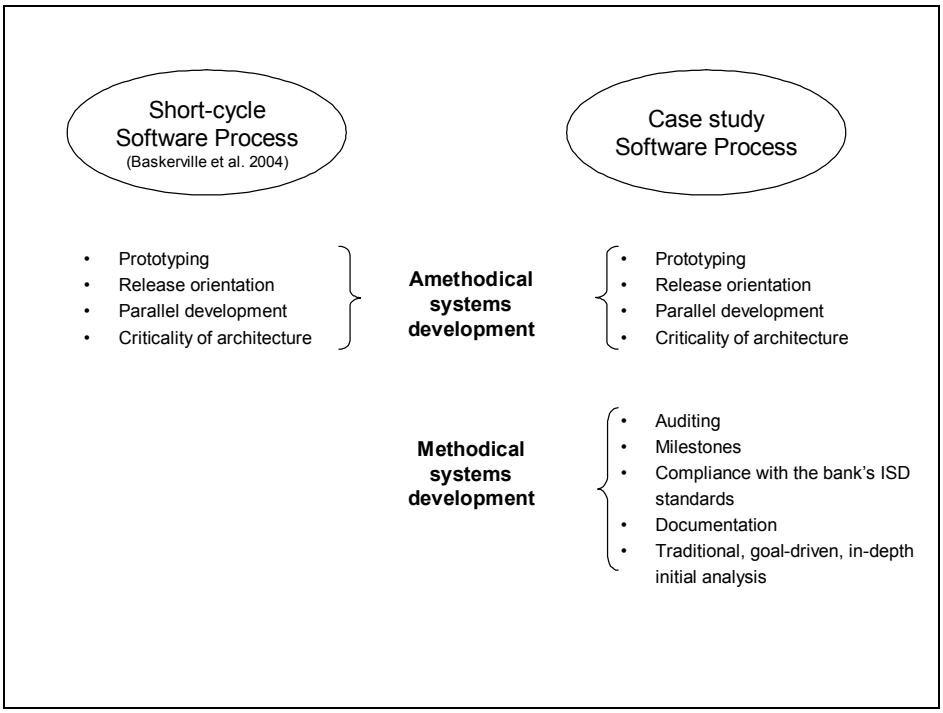

Figure 7: Comparison between the two studies.

In short, what appears to be involved is a process that could be categorised in a preliminary way as "hybrid", in that it does not fall precisely and completely into any predefined category.

5. Conclusions and Implications

The findings of this qualitative study provide an account of the characteristics of the Web services software development process within the context of the Central Europe Bank. As with any theory based on a single case, it remains to be determined to what extent the findings can be generalised to other settings.

The study suggests that Web services-based system development may constitute a new class of ISD practices. In particular, Web services-based system development offers the advantage of facilitating incremental analysis, requirements revision, requirements emerging in use and incremental implementation. In contrast to traditional short-cycle

development processes, the Central European Bank ISD process analysed here is partially methodical and partially amethodical. Rather than explain and interpret this from a narrow technological perspective a wider and more complex view was adopted, in line with an approach where a combination of factors (see Figure 4) is reputed significantly more important than choosing the "best" development platform. These factors include:

- market environment (i.e. target organisation and nature of software application, influencing completion speed and quality requirements);
- organizational learning associated with technical novelty;
- cultural factors emerging during an "outsourced cooperative process" (Sabherwal and Robey, 1993, p. 568), where the partnership between the bank and the supplier in the development and commercialisation of the software resulted in a cross fertilisation of two distinct organisational cultures.

5.1 Research Implications

It would seem that the case of the Central Europe Bank opens up some interesting research prospects. Although not arising directly out of the research evidence, several additional considerations can be offered to stimulate further research. These speculations are not limited to the analysis of software development processes but rather extend more generally to the action and decision-making processes that characterise complex organisations.

1) The Central Europe Bank project is characterised by uncertainty both in relation to means (and not just technological means) and aims (Thompson, 1967). This uncertainty explains the attempt on the part of the management of the bank to maintain a) a supervisory role in relation to the design processes through a control over the architectures and the company standards (even though it declined to realise the applications directly, due to a lack of competencies and an adequate capacity to carry out innovation); b) a supervisory role in the implementation processes (EAI organisation units); and c) a supervisory role over the processes of use, in particular, in relation to how the end users chose to use and control the services offered.
2) The process that led to the realisation of the CMS system at the Central Europe Bank throws light on a set of choices and objectives that changed over time in accordance with an increased awareness of the opportunities for and obstacles to change that sprung up. We believe that such opportunities and obstacles deserve to be analysed in detail, along with their business and functional implications. The technology used was only one of the factors that lay behind this continuous emergence of awareness.
3) The relational conception of power as "reciprocal dependence" (Crozier and Friedberg, 1977) explains, for example, the numerous processes of negotiation that took place at various level in the course of the design, implementation and utilisation phases of the system (e.g. between users and developers; between developers and the management of the bank etc.). In the future, the interpretation of such phenomena in the light of a process-oriented perspective will surely act as a great stimulus for researchers.

5.2 Implications for Practitioners

The implications of this study for the development effort in organizations are important for management and change agents involved in organizational contexts similar to Central Europe Bank.

1) The case has demonstrated that Web technology and, in particular, that based on "Web services" is not necessarily synonymous with amethodical and short-cycle development. Nonetheless, the experience at the Central Europe Bank shows that Web services can help to render more flexible and incremental "traditional" development projects in which the initial analysis of the requisites remains fundamental, the quality is non-negotiable and rigorous methods for the control and evaluation of results are deemed necessary. The greater level of flexibility, however, is not without costs: if they are not carefully controlled and managed, the increased architectural complexity and the rise in infrastructure management costs can outweigh the benefits.
2) Other problematic consequences of the approach include a high degree of difficulty in deciding upon the appropriate level of componentisation ("What is in and what is out", Levi and Arsanjani, 2002, p. 46). The difficulty of such decisions lies in achieving a correct balance between small fine-grained components that offer flexibility and larger coarse-grained components that are easier to manage and guarantee a certain level of performance.
3) Interactive development has proved of very high value, but at the same time system management poses some concerns in relation to problem identification, problem tracking, scalability and balancing of the IT infrastructure.

References

Baskerville, R., and Pries-Heje, J. (2001): Racing the E-Bomb: How the Internet Is Redefining Information Systems Development Methodology. In B. Fitzgerald and N. Russo & J. DeGross (Eds.), Realigning Research and Practice in Is Development: The Social and Organisational Perspective (pp. 49-68). New York: Kluwer.

Baskerville, R., and Pries-Heje, J. (2002): Information Systems development @ Internet speed: A new paradigm in the making!. Proceedings of ECIS 2002, Gdansk, Poland, pp. 282-291.

Baskerville, R., Levine, L., Balasubramanian, R., Pries-Heje, J., and Slaughter, S. (2003): Is Internet-Speed Software Development Different? IEEE Software, November/December.

Baskerville, R., Levine, L., Balasubramanian, R., Pries-Heje, J., and Slaughter, S. (2002): Balancing quality and agility in Internet speed software development. Proceedings of ICIS 2002, Barcelona, Spain, pp. 859-864.

Baskerville, R., and Pries-Heje, J., (2004): How Internet Software Companies Negotiate Quality, Information Systems Journal, No. 14, pp. 237-64.

Crnkovic, I., Hnich B., Jonsson, T., and Kiziltan, Z., (2002): Specification, Implementation, and Deployment of Components, Communications of The ACM, Vol. 45, No. 10, pp.35-40.

Crozier, M., and Friedberg H., (1977): "L'acteur et le système", Editions de Seuil, Paris.

Feagin, J., Orum, A., and Sjoberg, G. (eds.), (1991): "A case for case study", University of North Carolina Press, Chapel Hill (NC).

Glass, R. (2003): A Mugwump's-Eye View of Web Work: Choosing a point of entry into a contemporary software development debate. Communications of the ACM, 46(8), pp. 21-23.

Kautz, K. and Nørbjerg, J., (2003): Persistent Problems in Information Systems Development: The Case of the World Wide Web, in C. Ciborra et. al. (eds.): Proceedings of ECIS 2003, Naples, Italy, 19-21 June, 2003.

Levi, K. and Arsanjani, A., (2002): A Goal-driven Approach to Enterprise Component Identification and Specification, Communications of the ACM, Vol. 45, No. 10, pp. 45-52.

Orlikowski, W., and Baroudi, J., (1991): Studying information technology in organizations: research approaches and assumptions. Information Systems Research, Vol. 2, No.1, pp. 1-28.

Sabherwal, R., and Robey D., (1993): An empirical taxonomy of implementation processes based on sequences of events in information systems development, Organization Science, Vol. 4, No. 4, pp. 548-576.

Sommerville, I. (1982): Software Engineering, Addison Wesley, NY (6th edition: 2000).

Strauss, A. L., and Corbin, J., (1990): "Basics of Qualitative Research: Grounded Theory Procedures and Techniques". Sage Publications, NY.

Tellis, W., (1997): Introduction to case study, The Qualitative Report [On-line serial], Vol. 3, No. 2, http://www.nova.edu/ssss/QR/QR3-2/tellis1.html.

Thompson, J. D., (1967): "Organizations in action", Mc Graw Hill, NY.

Truex, D., Baskerville, R., and Klein, H. K. (1999): Growing Systems in an Emergent Organization. Communications of the ACM, Vol. 42, No. 8, pp. 117-123.

Truex, D., Baskerville, R., and Travis, J. (2000). Amethodical Systems Development: The Deferred Meaning of Systems Development Methods. Accounting, Management and Information Technology, 10, pp. 53-79.

Vessey, I., and Glass, R., (1998): Strong vs. Weak Approaches to Systems Development. Communications of the ACM, Vol. 41, No. 4, pp. 99-102.

Virili, F., and Sorrentino, M. (2004): Making stable systems grow with web services: a case study; in Proceedings of the IFIP WG 8.2 Organizations and Society in Information Systems (OASIS) Workshop, Seattle (USA), December 13-14, 2003.

Ye, F., and Agarwal, R., (2003): Proceedings of the International Conference on Information Systems (ICIS), "Strategic information technology partnerships in outsourcing as a distinctive source of information technology value: a social capital perspective", Seattle 2003, pp. 304-15.

Yin, R., (1994): "Case study research. Design and Methods", Sage Publications, Thousand Oaks.