

# IN SEARCH OF INFORMATION SYSTEMS DEVELOPMENT THEORY: A FRAMEWORK TO UNDERSTAND AGILE SOFTWARE DEVELOPMENT IN PRACTICE

Sabine Zumpe

University of Queensland, Brisbane, Queensland 4072, Australia

[s.zumpe@business.uq.edu.au](mailto:s.zumpe@business.uq.edu.au)

Karlheinz Kautz

Copenhagen Business School, Howitzvej 60, DK-2000 Frederiksberg, DK

[Karl.Kautz@cbs.dk](mailto:Karl.Kautz@cbs.dk)

**Abstract.** *The field of information systems development (ISD) is still not well understood and suffers from a lack of sustainable theories which are firmly based on research of ISD practice. This is also true for agile software development (ASD). In this paper we provide an integrated framework to support a theoretical understanding that allows both for a broad view on ISD practice in general and for a specific view on ASD. We demonstrate the framework in a case study of an ASD project in a large German public service organization and we discuss the identified development practices with regard to a theoretical foundation of ASD and ISD.*

## Introduction

The field of information systems development (ISD) is still not well understood and suffers from a lack of sustainable theories which are firmly based on research of ISD practice (Kautz 2004). This is also true for agile software development (ASD). The concept ASD serves as an umbrella for a number of pragmatic approaches, which have emerged out of a critique of traditional, document driven development approaches (Highsmith, 2002). In the agile manifesto the advocates of these approaches state their now well-known four values, namely (1) individuals and interactions over processes and tools; (2) working software over comprehensive documentation; (3) customer collaboration over contract negotiation, and (4) responding to change over following a plan (see [www.agilemanifesto.org](http://www.agilemanifesto.org)). They also provide 12 principles, which guide their work and provide a better understanding of the four values (see Table 1).

N.	Agile Principles
1.	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2.	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3.	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4.	Business people and developers must work together daily throughout the project.
5.	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6.	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7.	Working software is the primary measure of progress.
8.	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9.	Continuous attention to technical excellence and good design enhances agility.
10.	Simplicity - the art of maximizing the amount of work not done--is essential.
11.	The best architectures, requirements, and designs emerge from self-organizing teams.
12.	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Table 1. Principles behind the agile manifesto (see [www.agilemanifesto.org](http://www.agilemanifesto.org))

Some ASD proponents (Highsmith & Cockburn, 2001; Highsmith, 2002) put forward that these principles have a theoretical grounding in complex adaptive systems (CAS) theory. But as Vidgen & Wang (2006) and others (Kalermo & Rissanen 2002; Turk et al., 2002; Conboy & Fitzgerald, 2004) argue this claim is a post-rationalization, the theory is – if at all – used in a loose way to justify what is done in practice. Consequently the large amount of literature available on agile development in practice is of anecdotal and descriptive character. While these reports are useful as accounts of practice, they do not provide the theoretical underpinning for a deeper comprehension of ASD.

With the research presented in this paper we aim at contributing to fill in this gap. We provide an integrated framework to support a theoretical understanding that allows both for a broad view on ISD practice in general and for a specific view on ASD. It also assists in catching a wide range of empirical data from practice.

We demonstrate the framework in a case study of an ASD project in a large German public service organization and we discuss both the identified agile development practices from the standpoint of our framework and its contribution to a theoretical foundation of ASD and ISD.

The remainder of the paper is structured as follows: In the next section we present the theoretical background and the framework, which we developed for this study. Section 3 introduces the research approach which we applied. In section 4 we analyse our empirical data and discuss our findings. Finally, in section 5 we provide some conclusions.

## Theoretical Background and Framework

Wang & Vidgen (2007) argue that ISD that is driven by the paradigm of agility is substantially different compared to the traditional approaches. Agility is the ability “to sense and respond swiftly to technical changes and new business opportunities”

(Lyytinen & Rose, 2006). It emphasises the innate ability to deal with changes to the extent that an agile approach survives in this environment and emerges with success (Anderson, 2004). Vidgen & Wang (2006) present one of the few frameworks to theorize about ASD. Their framework draws on three overarching principles which constitute complex adaptive systems (CAS). For the purpose of their analysis they also offer six core concepts (see Table 2). The framework's focus on ASD is at the same time its strength and its weakness as it does not allow for a more expansive view on ISD practice. In addition, the six core concepts allow a certain level of detail, but are still quite coarse.

Core CAS Principles	Core CAS Concepts
Managing internal rates of change to match or exceed the relevant external change rates	Poise at the edge of chaos
	Time-pacing
	Coevolution
Optimizing self-organizing	Self-organization
	Interconnected autonomous agents
Synchronizing as concurrent exploration and exploitation	Poise at the edge of time

Table 2. The core CAS principles and concepts

A more general framework (see Table 3) that at the same time allows for a fine-grained analysis of ISD practice has been developed by Kautz and Madsen (Kautz, 2004; Madsen et al. 2006). The framework has also been successfully tested in a couple of case studies investigating Multimedia and Web information systems respectively (Madsen & Kautz, 2008).

Perspectives on ISD	Key Concepts of the Perspectives
Structuralist	Structural context
	Developers
	Information system
	Formalised method
Individualist	Repertoire and language
	Media
Interactive Process	Social Context: 1) Social Relations 2) Infrastructure 3) History
	Social Process: Politics and Culture
	Content of Change

Table 3. A three perspective framework to study ISD in practice

It explores the relationship between what influences and shapes ISD and in particular a unique and local method and how it consequently emerges. Based on a synthesis of prominent IS literature, the analytical framework consists of three perspectives, namely (1) the structuralist, (2) the individualist and (3) the interactive process perspective. Each perspective supplies a set of key concepts for conceptual understanding and empirical exploration of ISD and ISD method emergence in practice on a very detailed level. While the structuralist perspective consists of the information system under development, the formalised method to be used (if any), the structural characteristics of the involved development team and its members, as well as the project's structural context, the individualist perspective focuses on the involved individuals' repertoire, language used and their preferred media. The interactive process perspective centres around the social

relations, the social processes, and the content of change of the development project under investigation.

Combining the two frameworks and the 12 pragmatic principles of the agile manifesto exploits their respective strengths of generals and specifics. Their merger needs to be, however, carefully concerted due the relationships of the different framework elements. While the 12 principles relate to different aspects of the structuralist, individualist and interactive process perspectives, Vidgen & Wang's (2006) framework covers elements of the individualist and the interactive process perspective. In the following we present the resulting framework in more detail.

The **structuralist perspective** is concerned with the structural characteristics in an ISD project and how these characteristics impact the emerging method. Understanding the structural characteristics provides better insights into the settings of the project, the organisational design, and the participants involved (Madsen et al., 2006). The focus of this perspective is on static and hence, descriptive characteristics. The key concepts are: *context*, *developers*, *information system*, and *formalised method* (Madsen et al., 2006). Four ASD principles which describe compositional aspects of the approach can be related to the structural perspective. Principle 1 emphasises that the "highest priority is to satisfy the customer through early and continuous delivery of valuable software". The importance of working software is also noted in principle 3 "deliver working software frequently ..." and principle 7 where "working software is the primary measure of progress". In this sense, the software of the created information system provides a structure even in a highly flexible and agile development process. Although the software created is not static per-se it presents the results of ASD. Further, working software creates a link between the two concepts of formalised method and information systems as the former is used to develop the latter. Finally, the structure of agile projects should take simplicity (principal 10) into account, otherwise the structure might collapse and the managing of the change (rates) might fail.

The **individualist perspective** focuses on how individual project participants relate to each other stimulated by the emerging method. It is of particular interest to see how their behaviour shapes the method (Kautz, 2004). The developers' prior methodical and practical knowledge, but also their skills in the language and media deployed, influence the ISD approach (Madsen et al., 2006). The agile principles most relevant for this perspective are 5, 9, and 11 as they emphasise the single individuals by defining that "the best ... emerges from self-organizing teams" (principle 11). In ASD projects are built around motivated and trusted individuals working within an environment and with the support they need to pursue technical excellence and high quality to get the job done (principles 5 and 9). The capability to optimizing in self-organization is a core concept in ASD (Vidgen & Wang, 2006) where the organisation evolves bottom-up rather than top-down. All ISD participants are loosely interconnected as autonomous agents to ensure responsiveness to changes, but also to avoid to be overwhelmed by (unnecessary) information (Andersen, 1999). The individual perspective focuses only on the past experiences, knowledge and repertoire of both, business people and developers. It does not look at the social interactions in and between the two groups. The key concepts are: *repertoire*, *language*, and *media* (Madsen et al., 2006) which become visible in an agile development methodology such as eXtreme programming (xP) (Beck & Fowler, 2001; Beck & Andreas, 2004) in the capacity of conducting pair programming, leveraging of on-site customers and in the diversity of, for example, detail, correctness and completeness, of the story cards.

The **interactive process perspective** focuses on the dynamic aspects of ISD. ISD is impacted by the structural aspects of the project, the action of the individuals, and the

creation of the system as a series of changes (Kautz, 2004). Hence, this perspective builds on and adds to the previous two perspectives. The key concepts are: the *social context* consisting of the *social relations*, the *social infrastructure*, and the *history* of the project and those involved, the *social process* comprising its *politics* and its organisational (sub-)cultures, and finally the *content of change* covering both the characteristics of the planned and actual IS product and the process of change (Madsen et al., 2006). The agile principles emphasise interaction in ASD, for example by establishing “face-to-face conversation” (principle 6) and through the installation of stand-up meetings where “the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly” (principle 12). Collective ownership expresses (Lippert et al., 2002) the responsibility of each individual for the achieved results and encourages making changes to improve the quality of the information system and its software. The interactive process perspective considers the content of change and ASD amplifies the need for changes to the extent that changes are welcome “even late in development” (principle 2).

The management of the rates of change and the synchronization in an ASD project are only possible through interaction between the participants. The concept of time-pacing emphasises this through principal 4: “Business people and developers must work together daily throughout the project”. In an ASD project the participants work at the edge of chaos and interaction helps them to poise at the edge of chaos and find the balance between stable and unstable which characterises this edge. It is at the edge of chaos the participants of a development project tend to alter their behaviour as a response to their interactions among each others, but also with other parts of the system such as the used media and the environment. They do not just evolve as individuals, but co-evolve as members of a team. (Kauffman, 1993; Kaufmann, 1996).

The continuous poise at the edge chaos also shapes the balance of the internal rates of change and the external change rates (Vidgen & Wang, 2006). This relates again to principal 2 (“Welcome ... change for the customer's competitive advantage”). We also follow Vidgen & Wang (2006), who argue that change is triggered by the passage of time rather than by the occurrence of events. They cite Brown & Eisenhardt (1998), who contend that organizations create an internal rhythm that drives the momentum for change. This is related to the core concept of time pacing and its dynamic, rather than its structural aspects and to the agile principle 8 in which “agile processes promote sustainable development”.

The developed framework provides an analytic structure to investigate ASD as an instance of ISD in practice (see Figure 1). The benefit of using our framework is its capability to capture the ISD through different lenses. These lenses allow us to gain a comprehensive understanding of the agile development project and offer a powerful analytic instrument. Before demonstrating the benefits of our framework by applying it to a concrete case, we now introduce our research approach.

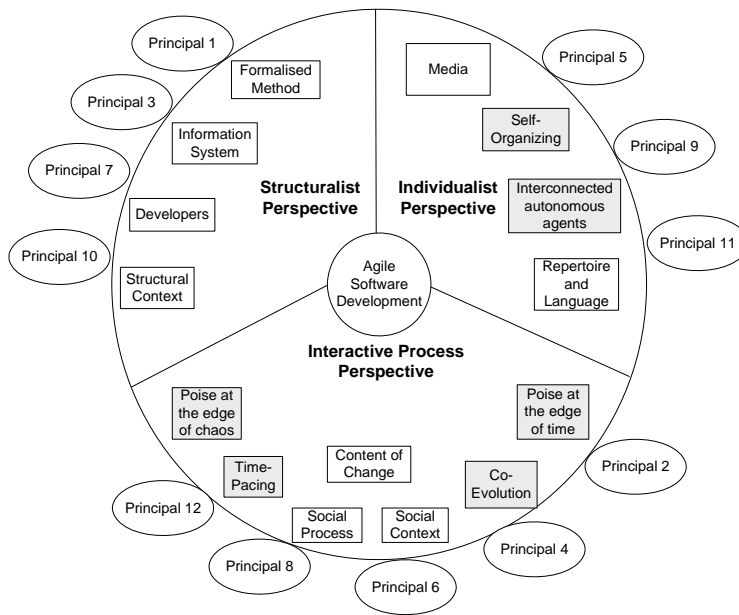


Figure 1. Extended Framework for investigating ASD Projects.

## Research Method

The research presented in this paper is interpretive and is based on an empirical case study of a project where a software development company applied an ASD approach to develop an information system for a large customer organization. The project was organised in 2 phases. When this study was performed phase one had been successfully closed and phase two had been going on for 4 months. The project ended 10 months later on time with all parts of the IS being operational.

Name	Role	Affiliation
Mr. A	Project Manager/analyst/developer	AgDev (2 interviews)
Mr. B	Project Manager	WaterWorks
Mr. C	Developer	AgDev
Mr. D	Developer	AgDev
Mr. E	Subproject manager/onsite user	WaterWorks
Mr. F	Subproject contact/analyst/developer	AgDev
Mr. G	Subproject contact/developer	AgDev
Mr. H	Chief IT co-ordinator	WaterWorks
Ms. K	Subproject contact/developer	AgDev
Mr. L	Subproject manager/onsite user	WaterWorks
Mr. M	Developer	AgDev

Table 4. The Interview partners

The empirical data for the case study was collected in semi-structured, open-ended interviews, which were conducted by a team of two researchers in a three days period. The research team performed 12 interviews with 11 individuals. Table 4 presents the roles and affiliation of the interviewees and shows that we talked to nearly have the

development team and a representative sample of key players and future users in the customer organisation. The interview protocols were tape-recorded and subsequently transcribed to be processed in a qualitative data analysis. We used the qualitative software tool NVIVO 7 for this purpose. For the coding of the documents we used a set of coding categories, which were derived from our extended framework (Miles & Hubermann, 1984). However, we also allowed for further codes during the coding process when interesting statements were identified. The subsequent analysis led to the results discussed in the next subsection.

## Analysis and Discussion

The project under investigation is concerned with the development of an operations management system (OMS) for the WaterWorks of a large German city by a small software company, AgDev. The system is developed with a web-based graphical user interface and a backend to interface the technical infrastructure as defined by an underlying ERP system. The project is organized in 4 subprojects to provide IT support ranging from customer management to the maintenance of the sewer system.

The OMS project was described by both the customer and the supplier as a success. In a presentation to the board the project champion at WaterWorks said: *“I am with the company now for 25 years but I never experienced a project that could generate output so fast.”* One of the WaterWorks subproject managers explicates: *“Our users have been very satisfied”* and the project manager confirms that the agile concept of working software as the measurement of project progress is much appreciated as *“it was amazing to have something to test so quickly.”* The project provided already after the initial exploration phase various benefits to the company, for example time savings and cost reductions. The emerging information system afforded, in the words of Mr. E, *“to identify synergies among the various departments”* and in particular in the duct department it enabled improved planning that resulted in the possibility to *“dispose cleaning vehicles and reduce related staff.”*

However, various comments were made about reaching the right balance between stability and instability to develop a viable ASD approach. AgDEV’s project manager for example comments on customer behaviour: *“Yesterday I said something and today I say something else. Requirements have to be clear at the beginning of an iteration and can not change right in the middle of it. We are agile but not on a daily or hourly change request rate.”* One of the WaterWorks subproject managers underlines that he sees himself as an *“agile person”* in a *“flexible and agile project”* which is embedded in an *“agile company”*. But he also emphasises that with regards to the project size and complexity *“planning is essential in such kind of projects”* and Mr. L adds *“you need this kind of documents – just because of the complexity of the project”* when arguing for the importance of project plans and requirement specifications also in ASD projects.

To gain a deeper understanding of ASD in practice we now apply our extended framework and analyse the investigated project in more detail in light of the three perspectives.

## The Structuralist Perspective

At the time of the project AgDev consisted of about 25 employees, 20 of them being developers, and based its development approach on the agile method xP (Beck & Fowler, 2001; Beck & Andreas, 2004). The AgDev project manager says: *“Sure we make everything that needs to be done in a xP-project, so, for example, pair programming, collective ownership, and testing.”* The formalized method includes planning techniques for releases and iterations called planning games, user stories and story cards to specify user requirements, onsite customers to support customer-developer communication, daily meetings (stand-up meetings) of the whole project team to support team communication, pair programming, re-factoring, collective ownership, continuous integration and testing to develop the software proper and tuning workshops to improve the development processes regularly.

AgDev has extended the method with some project management processes to cater for larger projects such as an elaborate overall project plan, formal reporting mechanisms and a formal contract based on a requirements specification produced by the customer. In the tender process AgDev had convinced the management of WaterWorks that their approach was viable and would deliver the OMS as requested.

The project was organised in 2 phases. In a first 12 months exploration phase prototypes catching requirements and possible solutions were developed. This led to the development of a comprehensive requirements document by the customer organisation and their decision to contract AgDev also for the development of the OMS proper.

In this main development phase a team of about 12 development staff with multiple roles such as project manager, analyst, customer contact, and developer worked onsite in a building owned by WaterWorks. The project team also consisted of a varying number of users with at least one representing one of the subprojects. These users were by and large, however, not the whole time onsite as well.

In addition to two project managers, each representing one of the companies, a sophisticated management structure was established. It consisted of one subproject manager, also acting as contact person, from AgDev and one subproject manager, also acting as onsite-customer, from WaterWorks for each individual subproject. According to Mr. B *“to have this counterpart was important to provide a first contact person.”*

The development team consists largely of highly educated and motivated, young staff and only the project managers have experience with ISD using an agile method, but none of them had ever participated in such a large project. The AgDev project manager was the most experienced team member: *“I have worked in projects with an agile approach for almost 5 years, but I never was involved in a project of this size.”*

Working software as addressed in the agile principles 1, 3, and 7 structures the development process. In the project software releases are provided every 3- 6 months with each release being organised in iterations of 3 – 6 weeks duration. Beyond this working software is also presented to the customer in shorter cycles as for example Mr. F describes: *“We really try to show something every week... there we make a presentation and demonstrate the software. It is a great way to get fast feedback.”*

Concepts	Characteristics in the ISD Project
Information system	Technically and functionally complex integrated operations management system (OMS) with web –based GUI user interface and ERP back end
Structural context	Sophisticated project organization with mutual human resources for each role in AgDev and WaterWorks: 2 overall project managers (1 AgDev, 1 WaterWorks), 4 subproject leaders AgDev, 4 subproject leaders WaterWorks; varying developer team size (6 – 12 people involved)
Formalised method	ASD approach Method with XP Programming: short releases and iterations of 3-6 months/3-6 weeks planning games, user stories, story cards, onsite users pair programming, collective ownership, stand-up meetings continuous integration, testing, re-factoring
Developers	University education in computer science and information systems, differing level of experience in the team, staff highly motivated to learn and to explore

Table 5. The Structuralist Perspective

## Individualist Perspective

The emergent method and the way how the ASD project evolved are driven by the AgDev team members' knowledge about agile development and user representatives' domain knowledge. Although some prior experience with agile methodologies existed, no team member had worked in an ASD project of this scale and size before. In this situation the experienced developers' repertoire of development techniques as provided by xP and their prior work with more traditional documents as media shaped the development process. Both, AgDev and WaterWorks relate to ASD "as a heavy learning process" for the involved individuals and Ms. L comments: "I enjoyed the learning. It was something new every day."

Quite a number of different documents exist, but they are all comparable short and concise. From a customer perspective they are related as follows: "Well, we have the overall realization concept as the basis for the contract and as a refinement hereof the requirements lists. These lists govern what should be the outcome of an iteration. ... And on the level below there are the story cards, these, so to speak, represent the detailed specifications and plans for the developer's process." The developers share this perception and confirm that the documents, both in length and t in number, are adequate, Mr. D says: "Absolutely sufficient" and is supported by Mr. C: "I flipped through the realization document in the beginning and never touched it afterwards ...."

The predominant media used in the development process are the story cards. They are based on the overall realization concept, which was produced by the customer as a basis for the contract and the requirements lists. These are largely produced by the AgDev staff, both their project leader and some of the subproject leaders, who also work as contact persons for their counterparts at WaterWorks and as developers. They develop these documents with input from the onsite customers. In the planning game users describe their requirements for a particular iteration and then the developers create the stories. "Story cards enable you to understand the context and if you need more details you just ask" summarises a developer." In general, iterations have 10 – 12 stories.

Despite the number of documents, the developers use little written material when implementing the story cards. Direct communication between the developers and

WaterWorks is preferred and perceived as sufficient. The level of communication and the degree of self-organisation varies across the four teams. Some developers interact more often with the users than others. Mr. L estimates “*that a story card explains only 60% of the requirement and the other 40% of it need further enquiries*”. The way feedback is gathered from WaterWorks reflects the self-organization of the ASD project members. Ms. K describes: “*Certain users insist on being contacted by phone while others prefer to be contacted per e-mail.*” Principal 11 emphasises that requirements best emerge from self-organizing teams.

Interaction and self-organization are also displayed in the way pair programming is organized. The developer teams emerge through spontaneous formations, as Ms. K puts it “*first I look who is free and then I go and work with him*”. When implementing the story cards, the developers act largely autonomous with regard to the design decisions they make. The AgDev staff members, however, maintain their relationships through daily stand-up meetings where all developers come together to report briefly about their activities. The different forms of interaction and (self-)organization support achieving technical excellence as described in principal 9. One WaterWorks subproject manager states “*... this way we have seen that we are on the right way, as we can use 95% of what has been developed this way, and just the last 5% we have to do something about again ...*”. This is confirmed by one developer by saying “*Yes, that functioned well, we made all 3 weeks a short presentation of the running software.*” and another extended: “*...we got very quick feedback when we showed what we had done.*” The short feedback cycles as part of the methodological repertoire provide the necessary structure to achieve the quality of the working software.

Finally, principal 5 emphasises trust in the developers to get the job done. Ms. K describes such a trust relationship within the development team: “*In a stand-up meeting a developer had a good idea and I assigned the task to him. The project manager allows such decisions made by ourselves.*” and on behalf of WaterWorks, Mr. H, their Chief IT co-ordinator confirms: “*We have a relationship of trust here. If we recognise, however, that it is exploited we need to react, but so far it has never been the case.*”

Concepts	Characteristics in the ISD Project
Repertoire and language	Repertoire and languages formed by experienced team members, on-site customers have domain knowledge
Media	Little use of comprehensive requirement documents, preference for direct communication, use of story cards to capture and realize system requirements
Self-organization	Establishment of short communication paths among the project members, self-organization in pair-programming
Interconnected autonomous agents	Implementation of daily stand-up meetings to achieve interconnectivity across the four development teams, but independence in the programming pairs

Table 6. *The Individualist Perspective.*

## Interactive Process Perspective

The OMS integrates complex functionality and was developed in an iterative and collaborative development process to replace systems, which had independently been developed in the different departments on platforms such as Access databases. The social

relationship between WaterWorks and AgDev evolved over time from non-existing to a partnership based on trust. It started in the tendering process, which WaterWorks had opened after several attempts of traditional ISD based on a standard ERP system had not led to the desired results, and with AgDev's tendering presentation. The project champion remembers: *"They presented a method, they explained it, and could convince us to get soon user feedback and a working solution."* and although Waterworks based on previous experience decided to separate the project into two phases with the option to leave it after phase 1, the project champion also states: *"... we decided not to be tough on change requests and back-up formalities, but rather to work constructively with them to make progress. And my good feelings have been confirmed."* The AgDev project leader confirms this and describes for the context of requirement changes: *"The customer is quite relaxed. In such situations they look where they can cut expenses planned for other requirements or we discuss if we can make the implementation simpler to meet the budget planned."*

The social infrastructure beyond AgDev's measures within the developer team - such as daily stand-up meetings and pair programming - can be portrayed as a close collaboration between AgDev and WaterWorks. There are onsite users available, as Mr. L said: *"My management put me here for 100%."* Further, to achieve a closer relationship between the teams of AgDev and WaterWorks the pairing of the team leaders was careful thought through. Mr. B said: *"... it was important for me to be confident that they matched with each other."* It was not by chance that the female AgDev sub-manager's counterpart is a female subproject manager.

The politics of the project are characterised by a relative flat hierarchy and autonomy in the development team. To support ASD a project champion was appointed, who strongly lobbies the project. But at WaterWorks two further power players exist, which influence the project: the employee committee and the internal IT department. The employee committee is perceived as a strong political player because they *"can stop the entire project"* as Mr. L put it. Involving them in software testing and the presentations of the working software calmed them down. Mr. E describes it as follows: *"We invited them early to our user workshops and they could observe and already see if they find critical issues"*. In addition, some employees of WaterWorks' IT department preferred a particular ERP system, which they had been involved to develop, and pressed the case for this system. Mr. L remembers *"they came to me and explained how their system could support my business processes."* and Mr. B adds in regards of the two players *"... we have lived with this potential conflict situation since the first day."* However these issues do not dominate and are dealt with by applying techniques, which show the benefits of the approach. Handling change is one such measure.

Change in the ASD project, especially change of requirements is an accepted fact of life. Many change requests are detected through the scheduled acceptance test sessions for an iteration with the customer representative onsite and are then dealt with in the next iteration. The changes emerge through the weekly and bi-weekly feedback sessions built into an iteration. The AgDev project manager explains: *"And then after a week the customer rep is back and wants to see what happened during the week and he gets the first feedback and this then continues ..."* They have the following consequence: *"... often we show the customer rep something once a week and then he's going 'well, I thought this would be different' ... thus there are always small changes ..."* as one developer puts it. These processes illustrate that the project welcomes change, has business people and developers work together and emphasises face-to-face conversations as recommended by the agile principals 2, 4 and 6.

The close collaboration also has an impact on the culture of the project. The developers despite their varying experience make up a quite homogeneous group, which in general is quite sympathetic to all customer related issues. Thus “*the well-known wars between the business people and the technicians*” as Mr . L puts it has no negative influence on the project. The frequent feedback loops have the effect that minor misunderstandings are caught and dealt with as changes early before they can grow into something larger. In each cycle the focus is on the current iteration and on the current user stories while taking the existing working software in account and design for future extensions as emphasised by the concept of poise at the edge of time and principal 8. An AgDev developer explains “*Until now it has not happened that everything was totally wrong; there are of course some refinements or a bug is found or something similar. There is always something.*” The feedback is taken seriously and immediately responded to with action: “*Through the feedback we got, we could react directly ...*” as it is described by one developer. This feedback is also used to reflect on their own behaviour as suggested in principal 12. Within all these structures, which support interaction and collaboration, the project balances continuously at the edge of chaos, the area, which is simultaneously stable and unstable. The example of reacting directly upon change requests illustrates this.

Concepts	Characteristics in the ISD Project
Social Context: Social Relations – Infrastructure – History	Relationship developed from non-existing to a trustful partnership project time, a thoroughly thought through project organization to support trust and cooperation
Social Process: Politics – Culture	Power plays with two forces: employees committee and ERP supporters Homogenous developer group and collaborative work atmosphere
Content of Change	OMS development as an iterative and collaborative development process
Time-Pacing	Continuous interaction with onsite customers
Coevolution	Continuous learning, interaction and collaboration
Poise at the edge of chaos	Structures support interaction and collaboration, yet the project balances continuously between the simultaneously stable and unstable, illustrated by the direct reaction on change requests
Poise at the edge of time	Working software focuses on the present taking the existing software and future design into account

Table 7. *The Interactive Process Perspective.*

At the edge of chaos coevolution takes place; stand-up meetings and pair programming are such situations where the developers learn from each other and as a consequence change their behaviour. Ms. K describes how her pair programming partners contribute to her further development: “*... anyway, he is better than me, and I learn something, and well with the other one I again learn something.*” Coevolution takes also place on a broader level. During one planning game one WaterWorks department needed more detailed requirement lists and story cards. The AgDev developers created these detailed requirement descriptions through a change in their usual behaviour and added this procedural change to their repertoire of acceptable agile actions.

## Conclusion

This study demonstrates the usefulness of our framework for understanding an ASD project in practice. By combining two existing frameworks with the 12 pragmatic principles of ASD we gained a comprehensive tool for our research.

Our analysis reveals how a successful ASD project with particular structural characteristics in an interplay with the involved individuals unfolded over time. It shows how the individuals as interconnected, but autonomous agents organize themselves and coevolve in the course of the process and how in this complex web of mutual influences the social context and process impact on how the information system in question is developed by poising at the edge of time and chaos.

As such our case analysis contributes to the theorizing on ISD and ASD in particular. More research applying the framework and providing more empirical evidence is however needed to further our understanding of the phenomenon.

## References

- Anderson, P. (1999). Complexity theory and Organization Science. *Organization Science: A Journal of the Institute of Management Sciences*, 10(3), 216-232.
- Anderson (2004). *Agile Management for Software Engineering*. Prentice Hall, Upper Saddle River, NJ.
- Beck, K., Andreas, C. (2004). *Extreme Programming Explained: Embrace Change*. 2<sup>nd</sup> Edition, Addison Wesley Professional, Mass., USA.
- Beck, K., Fowler, M. (2001). *Planning extreme programming*. 2nd Edition. Addison-Wesley, Boston (Mass.).
- Brown, S., Eisenhardt, K. (1998). *Competing on the Edge: Strategy as Structures Chaos*. Harvard Business School Press, Boston.
- Cockburn, A. (2002). *Agile software development*. Addison-Wesley, Boston (Mass.).
- Conboy, K., Fitzgerald, B. (2004). *Toward a Conceptual Framework of Agile Methods. Extreme Programming And Agile Methods - XP/ Agile Universe 2004, Proceedings*, Berlin, Springer-Verlag Berlin. Gordon, R. (1992). *Basic interviewing skills*. F.E. Peacock, Itasca (Ill).
- Highsmith, J. (2002). *Agile software development ecosystems*. Addison-Wesley, Boston (Mass.).
- Kalermo, J., and Rissanen, J.: *Agile Software Development in Theory and Practice*, M.Sc. Thesis on Information Systems Science. University of Jyväskylä, Finland (2002)
- Kauffman, S. (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford
- Kauffman, S. (1996). *At Home in the Universe: The Search for Laws of Self-Organization and Complexity*. Oxford University Press, New York.
- Kautz, K.(2004). *The Enactment of Methodology: The case of developing a multimedia Information system*. *Proceedings of the International Conference on Information Systems, ICIS 2004, December 12-15, 2004, Washington, DC, USA*, pp. 671-684.
- Lippert, M.; Roock, S., Wolf, H. (2002). *eXtreme programming in action : practical experiences from real world projects*. Wiley, New York (J.).
- Lyytinen, K., Rose, G. (2006). *Information system development agility as organizational learning*. *European Journal of Information Systems*, 15(2), 183-199.

- Madsen, S., Kautz, K. (2008). A Framework for identifying the Drivers of ISD Method Emergence. In Chiang, R. (ed.), Information Systems Analysis and Design -- Foundations, Methods, and Practices, Advances in Management Information Systems (AMIS) Monograph Series, M. E. Sharpe, Armonk, NY, USA (in print).
- Madsen, S.; Kautz, K., Vidgen, R. (2006). A framework for understanding how a unique and local IS development method emerges in practice. *European Journal of Information Systems*, 15(2), 225-238.
- Miles M., Huberman, M. (1984). *Qualitative data analysis : a sourcebook of new methods*. Sage Publications, Beverly Hills (CA).
- Turk, D., R. France, Rumpe, B. (2002). Limitations of Agile Software Processes. Third International Conference on eXtreme Programming and Agile Processes in Software Engineering. Alghero, Sardinia, Italy.
- Vidgen, R., Wang, X. (2006): Organizing for Agility: a complex adaptive systems perspective on agile software development process. Proceedings of the Fourteenth European Conference on Information Systems (Ljunberg J., Andersson, M. eds.), 1316-1327, Goteborg, Sweden.
- Wang, X., Vidgen, R. (2007): Order and Chaos in Software Development: A comparison of two software development teams in a major IT company. Proceedings of the Sixteenth European Conference on Information Systems (Winter, R. et al. eds.), 807-818, St Gallen, Switzerland.