20 THE ROLE OF EXTREME PROGRAMMING IN A PLAN-DRIVEN ORGANIZATION

Helene Dahlberg
Volvo Technology Corporation
Gothenburg, Sweden

Francisco Solano Ruiz
Elanders AB
Gothenburg, Sweden

Carl Magnus Olsson
University of Limerick
Limerick, Ireland

Abstract *The difficulties and even lack of commitment to follow plans within plan-based organizations is a well known phenomenon (see Ciborra et al. 2000; Suchman 1987). For software development companies, this problem has become an increasing dilemma, as typically plan-driven software development assessment standards like the capability maturity model (CMM) or ISO/IEC 15504 have not always been easy to conform processes against. Particularly, in environments where requirements are rapidly changing, more agile approaches such as Scrum and extreme programming (XP) have caught on. In this work, we are reporting from a case study of an organization looking to not move completely from their plan-based processes (as they are but a part of a larger organization operating in a plan-based way), but rather adapt their overarching processes in a way that allows them to use XP to support their everyday work precluded by their current processes. To this end, we present four perspectives that organizations may take when they desire or consider becoming more agile in their development. We use the Nerur et al. (2005) key issues for moving from plan-based to agile software development to compare and analyze our findings. In doing this, we highlight a set of likely criteria necessary to successfully create a combination of the plan-driven and agile approaches.*

Please use the following format when citing this chapter:
Dahlberg, Helene, Solano Ruiz, Francisco, Magnus Olsson, Carl, 2006, in International Federation for Information Processing (IFIP), Volume 206, The Transfer and Diffusion of Information Technology for Organizational Resilience, eds. B. Donnellan, Larsen T., Levine L., DeGross J. (Boston: Springer), pp. 291-312.

Keywords Agile software development, extreme programming (XP), plan-driven software development, Scrum

1 INTRODUCTION

Retaining control within an organization may be an elusive objective when it comes to software development. Ad hoc development tends to yield unpredictable quality, costs, and time to delivery. In an effort to address this, software development companies have been looking to define processes through which software quality, costs, and deliveries could be projected at an early stage, monitored throughout design and development, and insured prior to release to the customer. Through continuously improving such processes, organizations hope to find ways of working that guarantee software quality. Agile and plan-driven are two approaches to describe processes for software development. They are referred to as different approaches (understood as perspectives) when mentioned in general terms. Using the concept *processes* refers to an organizational point of view, where a series of actions to create software are described, while the term *method* defines a detailed level. Some of the most well-known agile approaches are XP and Scrum. Examples of plan-driven approaches are the Waterfall Model and CMM-based methods. Today, although plan-driven software development remains the most common approach, organizations are often drifting into ad hoc adaptations of these plan-driven processes, using quick-and-dirty solutions and behind-the-scenes improvisations that are far from what the official process prescribes. Frequently, management is not unaware of this, but simply elects to officially not be aware of such development habits. On the one hand, you have customers and managers demanding quick deliveries of excellent quality, using the latest and most appropriate technology, software development languages, and techniques. On the other hand, you are expected to strictly follow processes that demand a great deal of documentation, meetings, consideration, and decision makers to come together and make informed calls for what to do.

Research within Information Systems has recognized this tendency to drift (see Ciborra et al. 2000; Suchman 1987). While Suchman (1987) focuses on how plans, although not strictly followed, may still serve as a protective umbrella under which whatever action needed may be taken without anyone the wiser, Ciborra et al. (2000) describe how the natural tendency of plan-driven organizations is to drift more and more away from their initial plans, often resulting in a perceived need to plan even better the next time, with the same failing result. Through embracing and cultivation of the bricolage of actual work habits, Ciborra et al. argue that organizations stand a better chance of coping with change. It is not hard to see the similarities between these two examples with the elusive nature of plan-driven software development, as one of the fundamental problems it has is in coping with environments that change rapidly (Boehm 2002).

Agile approaches typically focus on simplicity and speed (Abrahamsson et al. 2003; Beck 1999a). The change from a plan-driven to an agile approach often involves refining the processes to identify the most appropriate ones for the organization. Even when the organization uses an agile approach, the process refinement is an ongoing work (Williams and Cockburn 2003). Although still required in agile approaches, the

need for continuous software process improvement is less demanding than in plan-driven approaches. While agile approaches have captured widespread interest, they are still not established as a mainstream development approach, and a reason for this may lie in the cultural heritage organizations have from plan-driven development processes. Another reason is that many organizations feel that being a certified CMM or ISO user increases their customers' confidence in them, while adopting agile methods could jeopardize relations.

For the purpose of this paper, we report on how Volvo Technology Corporation (VTEC), operating in a plan-driven organization with roots in assembly line production, is approaching agile software development methods *within* plan-driven setting. This represents a promising direction for how organizations may gain the credibility often perceived as needed from operating within a plan-driven and control-oriented organization, while still being able to use more flexible work patterns that better respond to changes in the development process. VTEC is an innovation company that develops new technologies and concepts for products and processes in the transportation and vehicle industry. Upon an initiative from VTEC to challenge the plan-driven model for development that they were prescribing but not seriously subscribing to in their everyday work, the study was initiated to help make an assessment of the current work habits and leave informed recommendations for if and how an agile model for development could be incorporated to support the work inside their plan-driven process framework. In particular, this interest was born from a desire (including customer requests) to become assessed according to ISO/IEC 15504 while also adapting an agile approach, in which they could gain the day-to-day support and use that they currently perceived as missing.

The study has been conducted using a seven-phase approach (section 3), where on-site observation over 3 months, informal and formal interviews, repertory grid sessions (Kelly 1955; Tan and Hunter 2002), workshops, and literature review (section 2 and further in section 4), have served as input to our analysis. Our findings are categorized and discussed in section 4. Representing the main contribution of this paper, the categorization and discussion is an expanded overview of the four key areas when considering a move from plan-driven to agile software development processes, identified by Nerur et al. (2005). Finally, section 5 summarizes the study and reiterates the contribution of this paper.

2 RELATED RESEARCH

Below, agile and plan-driven software methods are presented and compared. These are later used in section 4 to discuss our empirical findings. Plan-driven methods may be characterized as those in which work begins with the elicitation and documentation of a "complete" set of requirements, followed by architectural and high level-design development and inspection. Agile methods, on the other hand, often argue for incremental requirement specification, a minimal amount of documentation, and—importantly—reliance on individuals and interaction (including customer collaboration) rather than processes and tools.

2.1 Plan-Driven Approaches

Plan-driven approaches generally look to reduce risk by investing in life-cycle architectures and making long-term plans. Although taking steps in the process to reduce the problem, they accept that rapid change may make plans obsolete or costly to change on a frequent basis, both in terms of time and money.

Plan-driven approaches are best suited when developers can determine the requirements in advance—including via prototyping—and when the requirements remain relatively stable, with change rates on the order of 1 percent per month (Bohem 2002). When requirements change more often than on a monthly basis, it becomes increasingly difficult to keep requirements complete, consistent, testable, and traceable. It is still vital to have good documentation when developing safety-critical embedded software. Plan-driven approaches scale well to large, stable projects, but a bureaucratic, plan-driven organization that requires an average of one person-month just to get a project approved and started might not be considered efficient on smaller projects (Bohem 2002).

The capability maturity model has been widely adopted in the software community (see Mathiassen et al. 2002; Paulk 2001; Paulk et al. 1995). CMM is a method for evaluating the maturity of the software development process of organizations. The Software Engineering Institute, responsible for the CMM, argue that predictability, effectiveness, and control of an organization's software processes lead to software quality improvements as the organization moves up the five levels that comprise the standard.

- Level 1: Initial (processes are usually ad hoc and chaotic and products and services that work frequently exceed the budget and schedule of their projects).

- Level 2: Repeatable (software development success is repeatable, focusing more on project management to track cost and schedule).

- Level 3: Defined (engineering processes are described in standards, procedures, tools, and methods, while the organization's set of standard processes are established and improved over time).

- Level 4: Managed (product and process quality is controlled using statistical and quantitative techniques to find ideal management measurements for the software development).

- Level 5: Optimizing (process performance is continuously improved using innovative technology).

ISO/IEC 15504, sometimes referred to as SPICE (www.sqi.gu.edu.au/spice), is a suite of standards for software process assessment. ISO/IEC 15504 is similar to CMM, relying on levels of process development, use, and improvement to categorize an organization's increasing capability to follow software development processes that strive to increase product or service quality. ISO/IEC 15504 defines a measurement framework for the assessment of process capability on six levels.

- Level 0: Incomplete process (there is a general failure to achieve the purpose of the processes).

- Level 1: Performed process (the purpose of the process is generally achieved, but may not be strictly planned and tracked; products conform to specified standards and requirements).

- Level 2: Managed process (the performed process is planned, tracked, and adapted).

- Level 3: Established process (the process is performed and managed using a defined process based on good software engineering principles).

- Level 4: Predictable process (the defined process consistently operates within defied control limits to achieve its defined process goals).

- Level 5: Optimizing process (the process is optimized to meet current and future business goals).

At the time of the study, VTEC was preparing for an assessment of their processes toward ISO/IEC 15504, which had also triggered their interest in possibly incorporating an agile software development process such as XP into their defined process framework.

2.2 Agile Approaches

In the beginning of the 1990s, practitioners were starting to find the heavy initial requirements documentation, as well as architecture and design development steps, of plan-driven methods frustrating and even impossible (Williams and Cockburn 2003). Several alternative development methods were gaining public attention. Each of these had a different mixture of new ideas, old ideas and transformed old ideas, and where brought together under the "Manifesto for Agile Software Development" (www.agilemanifesto.org).

These agile methods were developed in various places at different times, but agile proponents agree that the central aspects of agile methods are simplicity and speed (Abrahamsson et al. 2003). The agile method Scrum was developed for managing the software development process in an unstable environment (Schwaber and Beedle 2001). Scrum leaves open to the developers the choice of specific software development techniques, methods, and practices for the implementation process. Less time is spent trying to plan and define tasks, and less time is spent on management reports. More time is spent with the project team. Another widely known agile method is Extreme Programming (XP) (see Beck 1999b, 2000). XP is based on four principles: simplicity, communication, feedback, and courage. XP is designed for use with small teams who need to develop software quickly in an environment of rapidly changing requirements. It is in XP that our case study organization has particular interest, as they are interested in likely consequences from an introduction of XP within their plan-driven framework. There are 12 key practices in XP, listed in Table 1.

Table 1. The Twelve Key Practices in XP (from *Extreme Programming Explained*, K. Beck, 1999)

The Planning Process	Pair Programming
Small Releases	Collective Ownership
Metaphor	Continuous Integration
Simple	Forty-hour Week
Refactoring	On-site Customer
Testing before coding	Coding

Since it has been argued that XP sometimes suffers from weak management control (Vriens 2003), a blend of Scrum and XP known as XP@Scrum has gained in popularity. This approach employs the management principles of Scrum and uses them together with the engineering principles of XP. Several critical people factors are emphasized for agile methods: amicability, talent, skill, and communication (Cockburn and Highsmith 2001). This is not to say that agile methods require uniformly high-capability people. Many agile projects have succeeded with people of mixed experience, as have plan-driven projects. The main difference is that agile methods achieve much of their agility by relying on the tacit knowledge embodied in the team, rather than always striving to collect and document the knowledge (Boehm 2002). There are risks involved in relying on this tacit knowledge, as unrecognized shortfalls may lead to irrecoverable architectural mistakes. Nevertheless, the effectiveness achieved when running smoothly is hard to deny. Agile methods require customers that are committed, collaborative, knowledgeable, representative, and operate with dedication to the development team in order to reach full potential. When little documentation is produced, formally reviewed, and agreed upon by all parties, the involvement of the customer is something that can not be ignored or allowed to slip.

2.3 Combining the Agile Approach in a Plan-Driven Context

Paulk et al. (1995) identify some interesting questions on agile- and plan-driven methods within software development. Can we combine selected agile practices with our traditional plan-oriented practices? How much change is necessary when transitioning to and using agile methods? How can agile practices improve the quality of our products? As the pace in the introduction of technology, tools, and techniques has picked up over the last decade, many CMM and ISO 9000 organizations are now reviewing how their process framework can be adapted to better suit a more frequently changing environment. One way of doing this has been to adopt some agile practices, looking to increase their efficiencies without having to abandon their plan-oriented strategies and (importantly) putting their plan-oriented certifications in jeopardy. Boehm (2002) argues that both agile and plan-driven have a home ground of project characteristics in which each works best, and the other will have difficulties. Further,

he argues that hybrid approaches that combine both methods are feasible, and even necessary for projects that combine a mix of agile and plan-driven home ground characteristics. The project leader behind the development of the CMM for software regards agile methods like XP to be compatible with CMM up to level three (Paulk 2001). Vriens (2003) describe a successful move from a plan-driven software development process to XP@Scrum, while still becoming approved for CMM level two. Another example of how agile thinking can come into a plan-driven context is illustrated by Mathiassen et al. (2002) and their in-depth exploration of how software process improvement (SPI) can become more agile, while retaining a plan-driven perspective. Using four case studies as an empirical foundation for their arguments, they identify several critical success factors for improving plan-driven software processes to better cope with rapidly changing environments. Key among these is the involvement of management in the continuous SPI efforts and the organizational determination to go through with them. Furthermore, the goals for improving the software development process must be in line with business goals in order to be implemented fruitfully. As our host organization VTEC are operating from a process designed to be assessed toward the ISO/IEC 15504 standard, we are, in this paper, assisting them in exploring how ISO/IEC 15504 can be used together with XP.

Before starting any move from a plan-driven to an agile approach there are many factors to consider, and organizations must carefully assess their readiness when doing so. Nerur et al. (2005) present key issues in making such decisions and we present an adapted and expanded version of these key issues in our discussion of the findings from the VTEC case study. Nerur et al., as does Boehm, advise caution when moving from a plan-driven to an agile development approach. "While the opportunities and benefits that agile methodologies afford make them attractive, organizations should be circumspect in embracing them or in integrating them with existing practices" (Nerur at al. 2005, p. 77). Organizations not heeding this warning risk making ill-judged decisions regarding the changes this might bring, how to proceed in implementing them, and what the overall cost versus benefit might be.

3 RESEARCH APPROACH

The intent of this study is to assess the current state of process use within VTEC with the specific purpose of identifying the effect that a combination of plan-driven and agile approaches would have on the organization. On a general level, this corresponds to the growing interest from industry to follow certified software development processes such as ISO and CMM, but at the same time retain the high level of flexibility that is perceived as vital in rapidly changing environments. To answer this, we have collected a wide array of data, ranging from on-site observation and field notes, to formal repertory grid sessions, informal interviews, and workshops. As the use of repertory grids has been seminal to our work, we spend section 3.1 on outlining the basic elements and assumptions of the technique, before we present our seven-phase approach in section 3.2.

3.1 Personal Construct Theory and the Repertory Grid Technique

Members of organizations attempt to make sense of their environment by interpreting events, actions, and objects through their personal view of the world. In IS research, this process has previously been explored through, for instance, *Weltanschauung* (Churchman 1971), cognitive maps (Weick and Bougon 2001), technological frames (Orlikowski and Gash 1994), and mental models (Daniels et al. 1995). In this work, we have addressed this through exploring the perceptions held by key personnel in the software development process at VTEC, using the notion of personal constructs. In his work on personal construct theory, Kelly (1955) argues that individuals strive to make sense out of the world by constantly testing their experiences against an evolving network of hypotheses.

> Constructs are used for predictions of things to come, and the world keeps on rolling on and revealing these predictions to be either correct or misleading. This fact provides the basis for the revision of constructs, and, eventually of whole construct systems (Kelly 1955, p. 14).

Kelly emphasizes that we use our own personal constructs to understand and interpret events taking place around us. By organizing these experiences into a personal construct system, we make sense of the present situation and try to anticipate future states. In essence, Kelly argues that it is natural for us as humans to consider the nature of our surroundings through personal constructs that are bipolar (i.e., that we define experiences in dualities such as <*Tall—Short*> or <*Nice—Rude*> if we are describing someone). The repertory grid technique is a structured yet highly flexible and participant-driven way to identify and assess our personal constructs (Olsson and Russo 2004).

The main components in repertory grids are elements, constructs, the links between them, and the sense-making process of analyzing them.

- Elements are entities within the domain of investigation and could be different things depending on the character of the domain. For instance, if the targeted domain is the general perception of different car brands, the choice of elements could be known brands like Volvo, Toyota and Ford.

- Constructs describe the character of the elements from the research participant's perspective. There are four alternative approaches in eliciting constructs (Tan and Hunter 2002). The approach used here is considered the classical way, and is known as the triadic sort form.

- Having obtained elements and constructs, the respondents are then asked to rate each of the elements using the elicited (in our case, but otherwise supplied) constructs.

- Techniques for analyzing the grids created include content analysis (Moynihan 1996), rearranging (Bell 1990), transforming (Shaw and Thomas 1978), and the original analysis of content and structure (Kelly 1955). However, in this paper, we

have chosen to focus on how the repertory grid technique also can be used as an efficient tool for eliciting and presenting personal constructs that, in our case, enabled us to perform in-depth group workshops as well as individual discussions with the respondents to identify the actual links between de facto development and the intended (plan-driven) software development process. These links represent the bricolage of perceptions, tools, and techniques used at VTEC to cope with the changing environment and projects of varying size and nature (some incorporating both software and hardware development).

As an example, we present the display grid of Respondent C, showing his perceptions on the various stages of their current process (Requirement, Formal Review, etc., on the bottom). On the left and right side of the figure are the constructs, and in between them are the rankings that Respondent C gave for their current development stages. For instance, looking at the construct <*Stimulating – Boring*>, Respondent C finds *Implementation* and *Formal Review* stimulating (a rating of one) while the *Documentation* and *Quality Assurance* stages are perceived as *Boring* (rating 5).

For the sake of completeness, we have included the display grids of all our respondents in the appendix.

3.2 Data Collection

We approached the study by defining clear phases for data collection and analysis. The data needed for this study was first an assessment of the actual (i.e., the enacted rather than prescribed) development process used at VTEC. Having obtained an understanding of this plan-driven process, we wanted to compare the actual process with the prescribed plan-driven process and XP practices to outline likely conclusions about how using XP would affect the organization. Continuously through the phases, we reviewed related literature from the adoption of agile processes, in particular focusing on XP.

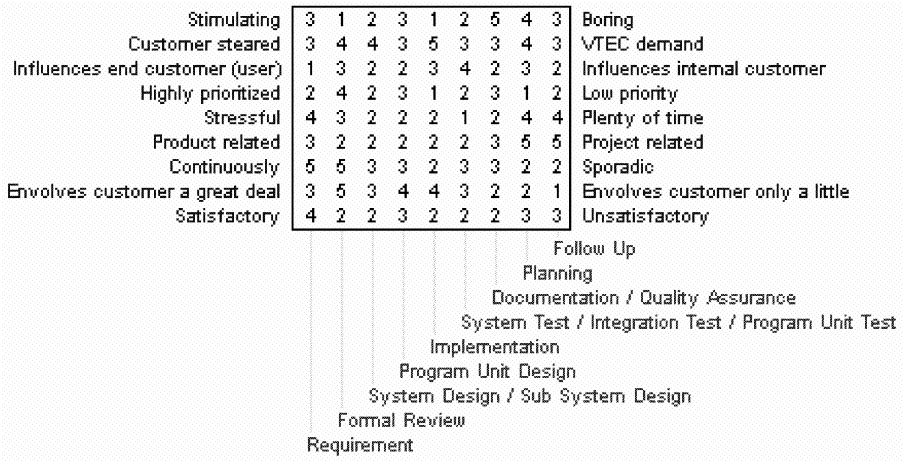


Figure 1. Display Grid for Respondent C

Table 2. The Six Phases of Our Research Approach

Phase A	**Contacting people for participating in interviews, planning the interviews, and choosing elements**. VTEC was asked to provide us with access to a group of representative users of VTEC's development process being considered as experienced developers or project leaders. We asked that this reference group have experience with various types and sizes of projects. The group formed as our reference group consisted of four people, which we refer to as respondents A, B, C, and D. Respondents A, B, and C are project leaders with a background as software developers; respondent D is a software developer.
Phase B	**Performing recorded interviews.** These recorded interview sessions were performed individually with the participants in the reference group. Prior to the interviews, we grouped the elements from Phase A into triads for eight different settings in such a way that every element would be reviewed at least twice.
Phase C	**Eliciting constructs and rating the elements.** To be able to compare the grid results between the participants of the reference group, and also to secure the quality of the constructs, we decided to adapt the use of the repertory grid technique in one aspect. The practiced method describes how the constructs are elicited during the interview sessions and further how this elicitation is immediately followed by the respondent's rating (Tan and Hunter 2002). Our adaptation was to separate the interview sessions from the rating and to identify the constructs from the interview data after having conducted all interviews. A few days after the interviews, the participants received a matrix to fill in their ratings, using the full set of constructs elicited.
Phase D	**Analyzing grids and interviews about VTEC's current development process.** As we have relatively few respondents, we choose to do an interpretative content analysis of the grids rather than rely on numerical facts. After this, we selected elements and constructs of particular interest for an in-depth discussion.
Phase E	**XP workshop at VTEC.** Equipped with a theoretical understanding of XP based on previous research, together with experiences from an organization already using XP and a good understanding of the current development process at VTEC, we performed an XP workshop where all members of VTEC were invited to participate. Participation was on a voluntary basis, but we were happy to see that the interest in XP was not limited to our reference group. Aside from presenting XP and previous experience from XP use, we used most of the time for open discussion among the participants.
Phase F	**Analyzing results from the workshop.** After the workshop (Phase E), we met with the participants of the reference group individually for follow-up interviews, again guided by our review of XP together with our grid results from phase D and feedback from phase E. We concentrated on the following practices: user stories, pair-programming, testing, collective code ownership, and keeping the design simple. After having transcribed workshop and follow-up data, we reviewed the accumulated material from two perspectives: a theoretical perspective and a practical perspective. Although in this paper we report primarily on the theoretical implications found from this study, we touch on some key practical aspects in our discussion of the findings below.

4 DISCUSSION OF THE FINDINGS

In this section we present and discuss our findings. To categorize and compare the bricolage of process use at VTEC (in short, we will refer to this as a bricolage perspective), we rely on the key issues outlined recently by Nerur et al. (2005) when moving from a plan-driven software development to agile approaches (i.e., a plan-to-agile perspective). Furthermore, we expand the Nerur et al. work by incorporating the work by Mathiassen et al. (2002) into the comparison. As Mathiassen et al. focus on achieving agility within the traditional plan-driven development standards—primarily using CMM to drive their discussion—this represents an agile-in-plan perspective. Finally, based on our findings, we draw some initial conclusions regarding adoption of XP within a plan-driven software development process (at VTEC), using Paulk (1999, 2001) and the XP@Scrum adoption into CMM (Vriens 2003) to compare and relate our conclusions. This creates a fourth perspective for achieving a more agile approach to software development: the XP at VTEC perspective. In future research, we hope to expand the XP at VTEC perspective with studies of organizations incorporating XP with well-established plan-driven approaches. Naturally, this perspective may also be expanded by considering agile methods other than XP. In Table 3, we present an overview of these four perspectives on agile development. We then spend the remainder of this section discussing this overview. We strongly suggest that the reader refer to the table when reading the text (numbered references in brackets between the table summaries and the outlined text are provided to facilitiate this), as the table is tightly coupled to the four key issues of management and organizational, people, process, and technology, including the subset of issues defined by Nerur et al.

4.1 Management and Organizational

When planning to change processes within an organization, one needs to involve considerations of the impact upon the organizational culture (Nerur et al. 2005). Mathiassen et al. use the concept *diffusion* when describing how to put changes into practice, and suggest the employment of an implementation workshop to achieve lasting changes. While culture in a plan-driven organization tends to be process oriented, the culture using XP is more social and team oriented (van Loon 2004). At VTEC, the development process is partly traditional, according to a plan-driven process, and partly agile in the sense that documentation has a lower priority than delivered code and functionality.

> *From the start all that is prioritized is the functionality. Not until late in the project the does customer ask for documentation, for example when problems turn up. Maybe we are prioritizing the wrong things and agree to develop too much; we should perhaps tell the customer that we can develop half the scope and spend the other half on documentation....but on the other hand, the customers do value functionality!* (Respondent D)

Table 3. An Overview of the Four Identified Perspectives to Agile Development

Perspectives		Agile-in-plan	Plan-to-agile	Bricolage	XP@plan
Key references		Mathiassen et al. (2002)	Nerur et al. (2005) VTEC case	VTEC	VTEC, Paulk (1999, 2001), Vriens (2003)
Management & Organizational	*Organizational culture*	Diffusion through workshop.	Change of culture is difficult.	Culture is receptive to a well argued and communicated case.	Throw communicative workshops early in the change effort. Coordinate with business goals [1].
	Management Style	Effective management for SPI initiatives.	Relinquish authority when adapting agile.	Not a large step from the bricolage practiced.	Continuous risk for power struggles from the management need to relinquish authority [2].
	Organizational Form	A resolute organizational effort	Traditional approach is formalized while agile is flexible.	Suitable for an agile shift.	Satisfy the level of management involvement required by the standards [3].
	Management of Software Development Knowledge	Knowledge creation through state-of-the-art comparisons.	Power shift from management to team.	Tacit knowledge communicated through socialization (=agile).	Define areas where knowledge creation is of interest, and assign responsibility to an SPI effort [4].
	Reward Systems		Suitable design of reward system is required.	No real incitement to stimulate an agile approach.	Team efforts and sharing must be recognized and rewarded [5].
People	*Working effectively in a team*	Team focus in SPI.	Might be an overwhelming experience.	A positive attitude to team work, uncertainty about working	Create an atmosphere that facilitates the team to develop respect, confidence and integrity [6].
	High level of competence		A culture of elitism might be created.	Focused on advanced engineering, requires skilled developers.	An innovative environment attracts competence [7].
	Customer relationships: commitment, knowledge, proximity, trust and respect	Involving customers in joint-venture activities.	Dedicated and knowledgeable customers might be hard to find.	The customer does not participate enough, and early feedback is not possible.	Important to probe the customers' ability to participate in an agile project before it is launched [8].

Perspectives		Agile-in-plan	Plan-to-agile	Bricolage	XP@plan
Key references		Mathiassen et al. (2002)	Nerur et al. (2005) VTEC case	VTEC	VTEC, Paulk (1999, 2001), Vriens (2003)
Process	*Change from process-centric to feature-driven, people-centric approach*	SPI efforts must be in line with business goals.	Takes time to change.	Leaving control and quality assuring is the largest difficulty.	Coordinate change effort with business goals, and carefully interpret process attributes [9].
	Short, iterative, test-driven development that emphasize adaptability		One of the largest barriers. Involves major alterations.	Test environment for an embedded development project needs to be simulated.	Establish customer cooperation where responsiveness is a natural part [10].
	Managing large, scalable projects		Little data exists.	Involved only with small projects of 4-15 people.	Unsuitable for large projects [11].
	Selecting an appropriate agile method		Select with care.	Quality assurance and management control are crucial.	Combination of agile and plan-driven methods [12].
Technology	*Appropriateness of existing technology and tools*	Communication channels in a social system in order to perceive common understanding.	Mainframe technologies may find it difficult to assimilate agile methods compared to OO development techniques.	The tool Clear Case is used for version handling.	Must be assessed [13].
	New skill sets-refactoring, configuration management, JUnits		Tools play a critical role in successful implementation. People must also be trained to use these tools successfully.	Tools like "Safer C" to review the code and to find discrepancies against MISRA C. Coding style guidelines.	Education about practices, tools and modeling techniques [14].

Changes in this environment might prove to be difficult as the employees are used to doing things their own way (rather than in accordance to a prescribed way). Thus, the employees have already made the move away from the original plan-driven process toward what they feel is more fitting. Still, introducing requirements to meet common goals and working together according to XP (within the existing plan-driven environment) is likely to require some strong argumentation [1].

Managing an SPI effort is different from other organizational changes (Mathiassen et al. 2002). Because improving software processes will take a long time, perhaps years, it calls for effective and strong management. While working with plan-driven development processes where an authorized, coordinating project leader has control, the same role in an agile project resigns control of decisions to the team. Hence, changing from command-and-control management to leadership-and-collaboration will be noticeable in transforming from a plan-driven to an agile approach. "The biggest challenge here is to get the project management to relinquish the authority he/she previously enjoyed" (Nerur et al. 2005, p. 76). The management style at VTEC again consists of a bricolage-like mix between plan-driven and agile approaches without heavy control of every step in the entire process. Instead there are more experienced development leaders supporting the team members of the project that hold much trust from management as well as project coworkers. Therefore, there need not be any problems with relinquished authority at VTEC, but rather recognition of an approach already used. XP is oriented toward developers, while traditional development processes have more of a management orientation. As mentioned earlier, combining Scrum with XP (Vriens 2003) gain a set of management practices that can be used to produce a wrapper for the XP engineering principles. Nevertheless, there is a potential conflict (Vriens 2003) in management style between an agile approach and the requirements with a plan-driven approach [2], as becoming agile is becoming more self-organized, which is contradictory to the commitment to centralized leadership required to satisfy the standards (particularly in the higher levels of CMM and ISO/IEC 15504) [3].

The organization at VTEC is suitable to use agile approaches as they are rather more adaptive and knowledgeable than they are bureaucratic and formalized, which otherwise characterizes a plan-driven organization (Nerur et al. 2005).

> *When necessary we develop the requirements and later receive feedback upon them from the customer, and sometimes even add requirements ourselves.* (Respondent D)

According to Mathiassen et al., SPI should emphasize knowledge creation and describe how the experience from software development practitioners is elicited into the SPI effort for comparison with current state-of-the-art theories, which are then fed back into the development work. The key factors for this are systematic evaluation and state-of-the-art knowledge. In traditional development, the knowledge is explicit in the form of documentation. Since agile approaches encourage lean thinking and little documentation, much of the knowledge stays tacit in the heads of the team members. Nerur et al. argue that this potentially can shift the power from the management to the development teams, and suggests that a decision is made on which knowledge should be codified and which may remain tacit. However, there also exist opinions among practitioners that since all code is collectively owned, no sole person possesses the knowledge about anything, and that this is power to management since individuals become replaceable. At VTEC, the priority is upon delivering working code before documentation. The knowledge is tacit and communicated among the team members through socialization (Nonaka 1994). This indicates that moving to agile will not be an obstacle when managing software development knowledge, but rather that some areas of particular interest should be selected for reporting of tacit knowledge in an SPI effort [4].

Adapting XP requires that management is attentive to other factors than individual technical competence when establishing the level of individual improvement with the staff, such as sharing competence and communicating. Nerur et al. suggest that a suitable design of reward systems is necessary to succeed with agile methodologies. For our case study, respondent C in particular highlights how people today (through the existing reward system) might be encouraged to withhold accomplished competence instead of sharing, thus creating more leverage negotiations with management on wages. To encourage communication and team efforts this must be rewarded to a greater extent than are individual accomplishments when moving to agile [5].

4.2 People

Working in an agile way means being member of a team, where communication and social interaction play an important role. Law and Charron (2005) describe that, in an integrated workforce, people act selflessly and contribute to the greater good which makes the team more than the sum of its parts. Descriptions of an ideal XP working area will describe a place were creative ideas are being exchanged between, and tested by, dedicated developers in a calm but stimulating environment. To reach this state there are many practices to embrace. According to Vriens (2003), and to Nerur et al., the ideas of shared learning, reflection workshops, pair-programming, and collaborative decision making might be overwhelming to programmers accustomed to solitary activities. At VTEC, there are strong-willed individuals, used to exchanging ideas and arguing for them. They also have a positive attitude to pair programming. However, adapting refactoring and collective code ownership must be introduced with care at VTEC.

> *I would like to receive an explanation if someone changed the code I had written!* (Respondent C)

A bigger issue at VTEC is the habit of applying very flexible working hours. To practice pair programming, refactoring, and 40-hours weeks successfully, XP depends (possibly to a larger extent than a plan-driven organization) upon the presence of the team. Mathiassen et al. do not explicitly address the effectiveness of a team in their guidelines to a successful SPI effort, but do spend significant effort on describing the usefulness of workshops for project team members. Clearly the team is in high focus when working on changes to an organization and we foresee an even larger demand for the respect, confidence, and integrity of the team members [6].

An agile team depends to a higher extent on the competence of the team members than traditional development that strictly follows expert-reviewed documentation (Boehm 2002; Nerur at al. 2005). Cockburn and Highsmith (2001) express their belief that agile development teams focus on individual competency as a critical factor in project success. The experience and education level at VTEC is high. Vriens reports that the innovative nature of XP@Scrum development has also proven to be an effective way to attract highly qualified engineers to the organization and we foresee a similar pull by adapting XP at VTEC [7].

When an organization is working on improving its software operation, customers will be seriously affected. It is therefore important to involve the aspects of customers, or even better, to initiate joint activities aimed at improving the customer-supplier relationships (Mathiassen et al. 2002). Nerur et al. recognize that finding a highly dedicated and knowledgeable customer might be difficult. Among VTEC's customers, competence varies and they mostly have limited ways to become involved in the development process due to time constraints of their own. Probing and understanding the type of customer thus becomes very important [8].

4.3 Process

Agile methodologies rely on people and their creativity rather than on processes, and changing from a process-centric to a feature-driven people-centric approach is one of the largest barriers to cross (Nerur 2005). Leaving a plan-driven development model would result in several difficulties for VTEC as the vehicle industry has a long tradition in working according to standards and assuring quality through plan-driven development processes. To have the change effort coincide with business goals is, therefore, particularly crucial to VTEC [9], as this is recognized as an important success factor when changing existing processes (Mathiassen et al. 2002).

Vriens suggests implementing an independent quality assurance group for obtaining the requirements from the ISO 9000 and CMM standards. Further, he says that pair programming is effective in achieving a high level of assurance regarding conformance to standards, though it doesn't really give management (enough) visibility into non-conformance issues. ISO/IEC 15504 at capability level 3 states that "responsibilities and authorities for performing the process are assigned and communicated," while in XP there is an assigned customer and tester; all other responsibilities are assigned to the development team as a whole, and at any time different people can assume different responsibilities and authorities. This is not incompatible with ISO/IEC 15504 but highlights the need for an assessor to carefully interpret the process attributes of the standard.

One problem identified within the development process at VTEC is the lack of relevant feedback on releases. Customers require releases during a development project, but take too long to supply feedback for it to be relevant to the development.

> *The idea of short releases as XP means would not be easy to apply today as the customer does not really have the time to test and evaluate them.* (Respondent D)

We believe, however, that the development process both at VTEC and other organizations that considers adapting to agile engineering principles could gain from more established cooperation with the customer where development of requirements and responsiveness to changes is a natural part [10]. Vriens (p. 2) states that "Both customers and programmers appreciate working iteratively, making small increments. The customer appreciates business value, and programmers value the learning by experience."

Agile methodologies differ in terms of team size. There is not much data collected upon the efficacy of agile approaches with regard to large projects since agility usually

is targeted toward small- to medium-sized teams expected to be colocated, typically with less than 10 members. The current software development projects at VTEC consist of 4 to 15 developers, and there are currently no larger projects, so an agile approach would be feasible and even suitable in terms of size [11]. Paulk et al. (1995) express that as systems grow, some XP practices become more difficult to implement. XP is targeted toward small teams working on small- to medium-sized projects. As projects become larger, emphasizing a good architectural "philosophy" becomes increasingly critical to project success.

To VTEC, we feel that the main issues when selecting an agile method would be to keep the quality assurance level they have today along with the possibility to assess for ISO/IEC 15504. In his article, Vriens shows that using agile approaches for the software engineering and plan-driven approaches for management control is a way to go. Van Loon (2004) gives an overview of the relationships between XP practices and ISO/IEC 15504 processes, and concludes that XP is compatible with ISO up to level two before it needs additional management. There are many agile methods to select from when choosing the appropriate one, and while they all agree with the agile manifesto, they obviously are not the same in every aspect. Nerur et al. recognize that the methods differ in team sizes, mechanisms for rapid feedback and change, and that an organization must decide which is most suitable to its needs. Again, in VTEC, there are many indications that using an agile approach in combination with keeping some traditional plan-drive processes (although carefully selected) would be an attractive alternative [12].

4.4 Technology (Tools and Techniques)

Failing to understand the needs of the target user groups is a common mistake of the SPI effort. Mathiassen et al. (pp. 258) talk about communication channels in a social system in order to perceive this common understanding of the users needs, and argue that personal communication channels (as opposed to mass media channels) are generally most effective when it comes to convincing a target group of an innovation's value.

Nerur et al. express that an organization's existing technology can impact the efforts to migrate to agile methodologies. Companies that rely solely on mainframe technologies may find it difficult to assimilate agile methods compared to those that use object-oriented development techniques. The development language at VTEC is C and is not strictly object-oriented. There could be small difficulties at the beginning when adapting to an agile methodology. Regarding common code ownership, recall that respondent C wanted an explanation if the code were changed. However, none of the respondents thought this would be impossible for them to adopt, but rather argue that it be done with respect and communicated among the team.

Vriens notes a general high satisfaction applying XP, but thinks that XP does not give much help regarding documentation, modeling, and the use of UML and design patterns, and thus asks for further research on these areas. Tools do play a critical role in successful implementation of a software development methodology (Nerur 2005), meaning that the sharing of knowledge also applies to hands-on aspects for using XP and other agile methods. Organizations planning to adopt agile methodologies must look for, and invest in, tools that support and facilitate rapid iterative development, versioning and configuration management, and refactoring [13]. Employees must also

be trained to use these tools successfully and committed to the fact that in XP, writing test cases first has everything to do with architecture and design (Vriens 2003) [14].

5 CONCLUSIONS

In this paper, we set out to expand the key issues identified by Nerur et al. (2005) as particularly relevant when an organization moves from a plan-driven to an agile software development process. In doing so, the case study at VTEC provided us with a picture of the bricolage of work practices that the organization and individual project members had drifted into using, rather than always following the prescribed development process. Triggered by a desire to become a licensed ISO/IEC 15504 developer, VTEC was particularly interested in, if possible, integrating XP into their high-level plan-driven (but updated) development process. By analyzing our data using the Nerur et al. key issues, and comparing this with the Mathiassen et al. (2002) exploration of how a plan-driven organization may become more agile (even without adopting any of the typical agile methodologies), we arrived at a set of characteristics for what would be necessary to have a combined XP and plan-driven approach. While we intend to continue to probe the feasibility and limitations of our characteristics by looking at other organizations (as well as continuing to study the VTEC case) that also share this interest, we would much welcome other researchers to report findings related to this study to the research community.

References

Abrahamsson, P., Warsta, J., Sipponen, M. T., and Ronkainen, J. "New Directions on Agile Methods: A Comparative Analysis," in L. Clarke, L. Dillon, and W. Tichy (eds.), *Proceedings of the 25th International Conference on Software Engineering*, Los Alamitos, CA: IEEE Computer Society Press, 2003, pp. 244-254.

Beck, K. "Embracing Change with Extreme Programming," *IEEE Computer* (32), 1999a, pp. 70-77.

Beck, K. "Extreme Programming Explained," Reading, MA: Addison-Wesley, 1999b.

Beck, K 2000. "Extreme programming explained: embrace change," *Addison-Wesley*, Reading MA.

Bell, R. C. "Analytic Issues in the Use of Repertory Grid Technique," *Advances in Personal Construct Psychology* (1), 1990, pp. 25-48.

Boehm, B. "Get Ready for Agile Methods, with Care," *Computer* (35:1), 2002, pp. 64-69.

Churchman, C. *The Design of Inquiring Systems—Basic Concepts of Systems and Organization*, New York: Basic Books Inc., 1971.

Ciborra, C. U., Braa, K., Cordella, A., Dahlbom, B., Failla, A., Hanseth, O., Hepsø, V., Ljungberg, J., Monteiro, E., and Simon, K. A. *From Control to Drift: The Dynamics of Corporate Information Infrastructures,* Oxford, England: Oxford University Press, 2000.

Cockburn, A., and Highsmith, J. "Agile Software Development: The People Factor," *IEEE Computer*, November 2001, pp. 131-133.

Daniels, K., de Chernatony, L., and Johnson, G. "Validating a Method for Mapping Managers Mental Models of Competitive Industry Structures," *Human Relations* (49:9), 1995, pp. 975-991.

Kelly, G. *The Psychology of Personal Constructs, Volumes 1 and 2*, London: Routledge, 1995.

Law, A., and Charron, R. "Effects of Agile Practices on Social Factors," in *Proceedings of 27th International Conference on Software Engineering: Workshop on Human and Social Factors of Software Engineering*, St. Louis, MO, May 15-21, 2005, pp. 1-5.

Mathiassen, L., Pries-Heje, J., and Ngwenyama, O. *Improving Software Organizations*, Boston: Addison-Wesley, 2002.

Moynihan, T. "An Inventory of Personal Constructs for Information Systems Project Risk Researchers," *Journal of Information Technology* (11), 1996, pp. 359-371.

Nerur, S., Mahapatra, R., and Mangalaraj, G. "Challenges of Migrating to Agile Methodologies," *Communications of the ACM* (48:5), May 2005, pp. 72-78.

Nonaka, I. "A Dynamic Theory of Organizational Knowledge Creation," *Organization Science* (5:1), 1994, pp. 14-37.

Orlikowski, W. J., and Gash, D. C. "Technological Frames: Making Sense of Information Technology in Organizations," *ACM Transactions on Information Systems* (12:2), April 1994, pp. 174-201.

Olsson, C. M., and Russo, N. "Applying Adaptive Structuration Theory to the Study of Context-Aware Applications," in B. Kaplan, D. P. Truex, D. Wastell, A. T. Wood-Harper, and J. I. DeGross (eds.), *Information Systems Research: Relevant Theory and Informed Practice*, Boston: Kluwer Academic Publishers, 2004, pp. 735-741.

Paulk, M. "Analyzing the Conceptual Relationship between ISO/IEC 15504 (Software Process Assessment) and the Capability Maturity Model for Software," *International Conference on Software Quality,* Cambridge, MA, 1999 (available online at http://www.telnet.com.tn/Doc/ISO_15504_Maturity_Model_for_Software.pdf).

Paulk, M. "Extreme Programming from a CMM Perspective," *IEEE Software* (18:6), November-December 2001, pp. 19-26.

Paulk, M., Curtis, B., Chrissis, M. B., and Weber, C. V. *The Capability Maturity Model: Guidelines for Improving the Software Process*, Reading, MA: Addison-Wesley, 1995.

Schwaber, K., and Beedle, M. *Agile Software Development with Scrum*, Upper Saddle River, NJ: Prentice Hall, 2001.

Shaw, M. L. G., and Thomas, L. F. "FOCUS on Education: An Interactive Computer System for the Development and Analysis of Repertory Grids," *International Journal of Man-Machine Studies* (10), 1978, pp. 139-173.

Suchman, L. A. *Plans and Situated Actions: The Problem of Human-Machine Communication*, New York: Cambridge University Press, 1987.

Tan, F. B., and Hunter, M. G. "The Repertory Grid Technique: A Method for the Study of Cognition In Information Systems," *MIS Quarterly* (26:1), 2002, pp. 39-57.

Van Loon, H. *Process Assessment and ISO/IEC 15504*, Berlin: Springer, 2004.

Vriens, C. "Certifying for CMM Level 2 and ISO 9001 with XP@Scrum," in *Proceedings of the Agile Development Conference*, Salt Lake City, UT, June 25-28, 2003, pp. 120-124 (available online at http://www.agiledevelopmentconference.com/2003/files/R8Paper.pdf).

Weick, K. E., and Bougon, M. G. "Organizations as Cognitive Maps: Charting Ways to Success and Failure," in K. E. Weick (ed.), *Sensemaking in Organizations*, Beverly Hills, CA: Sage Publications, 2001, pp. 308-329.

Williams, L., and Cockburn, A. "Agile Software Development: Its about Feedback and Change," *IEEE Computer* (36:6), June 2003, pp. 39-43.

About the Authors

Helene Dahlberg received a bachelor's degree from the IT University of Gothenburg in 2005 and since graduating has been employed as development engineer at Volvo Technology Corporation AB (VTEC) at the department of Electronics and Software. She takes a special

interest in achieving and improving processes for cost effective development of quality software. Helene can be reached at it2dahe@ituniv.se.

Francisco Solano Ruiz received a bachelor's degree from the IT University of Gothenburg in 2005 and since graduating has been employed as system developer at Elanders AB. One of his special interest lies in helping organizations to structure their information and developing cost effective web applications to store their information. He can be reached at francisco.solano@elanders.com.

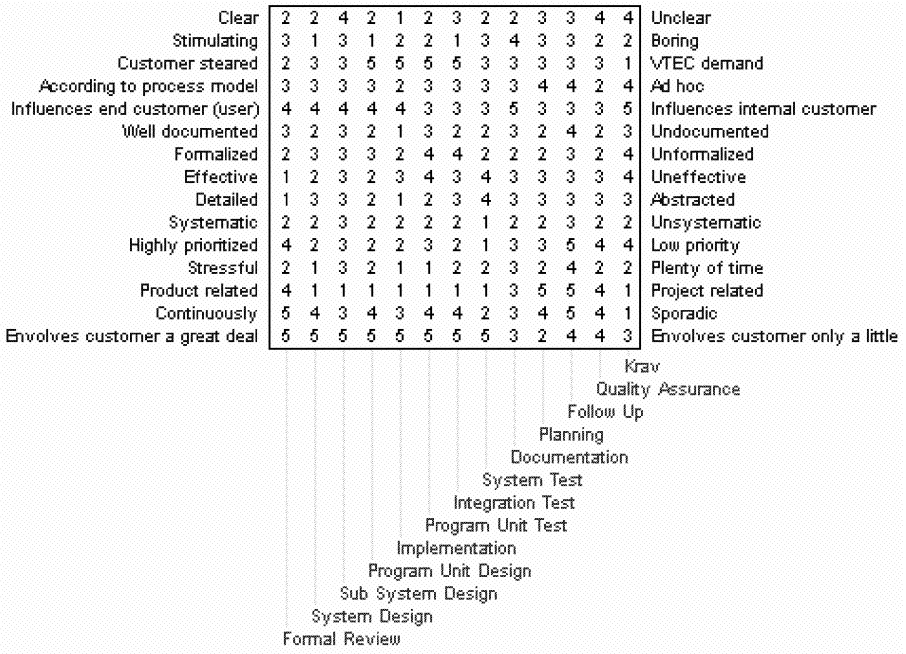
Carl Magnus Olsson is a Ph.D. student at the University of Limerick, Ireland. He conducted the research reported here in collaboration with the IT University, Gothenburg, Sweden, and holds a guest researcher position at the Operations Management and Information Systems Department of Northern Illinois University, USA. He was a participant of the ICIS Doctoral Consortium in Seattle, 2003, and has published conference papers at IFIP WG 8.2, IFIP WG 8.6, and International Conference on Information Systems. He can be reached at cmo@ul.ie.

Appendix: Display of Repondent Grids with Ratings of Constructs Against Elements

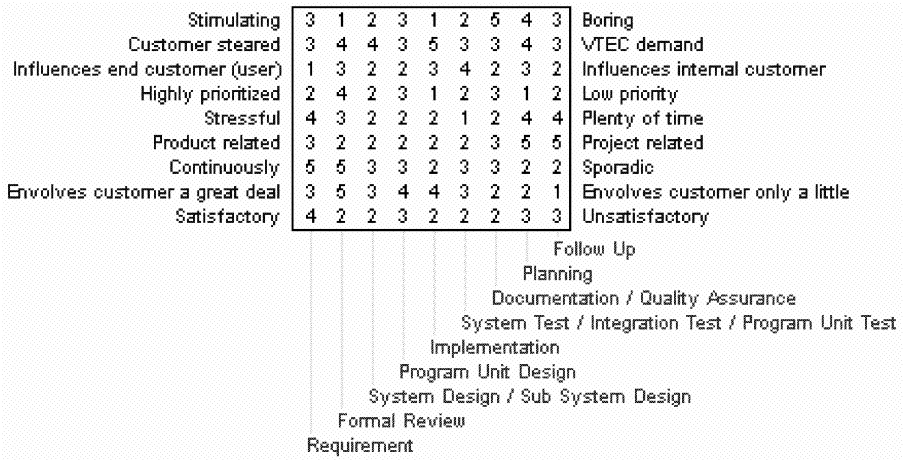
Clear	2	2	3	4	4	2	4	3	2	2	2	4	4	Unclear
Stimulating	3	4	2	2	2	2	3	3	2	3	2	4	3	Boring
Customer steared	2	4	3	4	4	3	3	2	2	2	2	4	4	VTEC demand
According to process model	5	5	3	3	3	2	2	2	2	2	2	3	3	Ad hoc
Influences end customer (user)	1	5	2	3	4	1	3	2	1	2	3	3	3	Influences internal customer
Well documented	3	5	2	2	4	1	4	3	2	2	2	4	4	Undocumented
Formalized	4	3	2	2	4	5	4	3	2	2	2	4	4	Unformalized
Effective	4	5	2	2	2	1	2	2	2	3	2	3	3	Uneffective
Detailed	2	2	3	3	3	1	2	2	2	4	3	4	4	Abstracted
Systematic	3	2	2	2	2	2	4	4	2	2	2	3	2	Unsystematic
Highly prioritized	2	5	2	3	3	1	2	2	2	2	?	4	3	Low priority
Stressful	5	2	5	5	5	3	3	2	2	2	4	2	2	Plenty of time
Product related	1	5	2	2	3	4	4	3	1	1	3	4	4	Project related
Continuously	4	5	3	3	3	1	1	2	2	5	4	4	4	Sporadic
Envolves customer a great deal	2	5	2	4	5	5	4	3	2	1	1	3	3	Envolves customer only a little

Quality Assurance
Follow Up
Planning
Documentation
System Test
Integration Test
Program Unit Test
Implementation
Program Unit Design
Sub System Design
System Design
Formal Review
Requirement

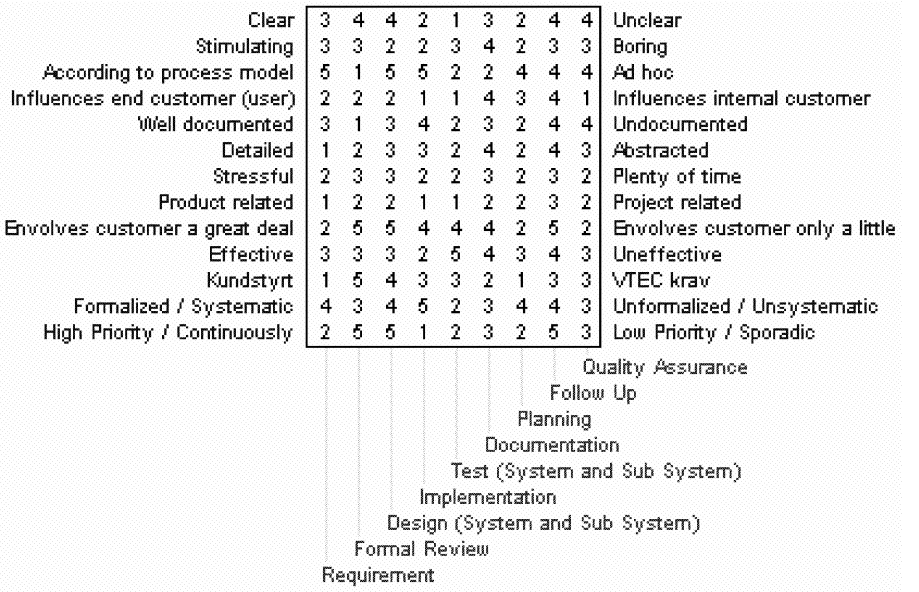
Grid from Respondent A



Grid from Respondent B



Grid from Respondent C



Grid from Respondent D