



Agile Software Development of Embedded Systems

Version : 1.0
Date : 2005.04.04
Pages : 24

Authors

Outi Salo
Minna Pikkarainen

Status

Final

Confidentiality

Public

Agile Deliverable D.4.4

Agile Deployment Model

Abstract

This document presents an Agile deployment model that aims at providing organizations with means to adopt and adapt agile methods in their software development context.



I T E A

INFORMATION TECHNOLOGY

FOR EUROPEAN ADVANCEMENT



TABLE OF CONTENTS

CHANGE LOG	2
APPLICABLE DOCUMENT LIST	2
1. Introduction.....	3
2. Agile Deployment Model.....	4
2.1 Agile Assessment.....	6
2.1.1 Purpose.....	6
2.1.2 Procedure of Agile Assessment.....	6
2.1.2.1 Focus Definition.....	7
2.1.2.2 Agility Evaluation	7
2.1.2.3 Data Collection Planning.....	8
2.1.2.4 Interviews.....	8
2.1.2.5 Analysis	9
2.1.2.6 Workshops and Learning.....	9
2.2 Deployment Planning.....	10
2.2.1 Purpose.....	10
2.2.2 Procedure	10
2.2.2.1 Analysis, Prioritization and Selection	11
2.2.2.2 Selection of Deployment Approach	11
2.2.2.3 Selecting of Pilot Projects.....	12
2.2.2.4 Planning of Feedback.....	12
2.3 Initial Deployment	12
2.3.1 Purpose.....	12
2.3.2 Procedure	12
2.3.2.1 Planning the Piloting.....	12
2.3.2.2 Tailoring the Base Process.....	13
2.3.2.3 Training.....	13
2.3.2.4 Preparing the Software Development Environment	13
2.4 Post-Iteration Workshop Method	13
2.4.1 Purpose.....	13
2.4.2 Procedure	14
2.4.2.1 Preparation	15
2.4.2.2 Collection of Experience.....	15
2.4.2.3 Planning of Improvement Actions.....	16
2.4.2.4 Piloting.....	17
2.4.2.5 Follow-Up and Validation.....	18
2.4.2.6 Packaging.....	18
2.5 Continuous Improvement of Organizational Practices	18
2.5.1 Purpose.....	18
2.5.2 Procedure	18
2.5.2.1 Analyze Feedback.....	19
2.5.2.2 Select/Adapt Organizational Software Development Practices	19
2.5.2.3 Update Organizational Processes.....	20
2.5.2.4 Disseminate the New/Altered Practices to Projects.....	20
3. Critical Success Factors in Adopting Agile Practices	20
4. Conclusions	21
5. References	22



AGILE document template
Deliverable ID: D4

Page : 2 of 24

Version: 1.0
Date : 04.04.05

Status : Final
Confid : Public

CHANGE LOG

Vers.	Date	Author	Description
0.1	28.2.2005	Outi Salo	Outline draft created.
0.2	4.3.2005	Outi Salo, Minna Pikkarainen	First Full Draft Creation.
0.3	7.3.2005	Outi Salo	Updated based on VTT internal review (except Agile Assessment part).
0.4	14.3.2005	Minna Pikkarainen	Agile Assessment part updates.
0.5	21.3.2005	Outi Salo & Minna Pikkarainen	Updated based on ITEA review/ESI/MariaElisa Gallo. Critical success factors –chapter added based on the text from ESI.
0.6	29.3.2005	Outi Salo & Minna Pikkarainen	Updated based on ITEA review/Philips/Ko Dooms.
1.0	4.4.2005	Outi Salo	Language checked.

APPLICABLE DOCUMENT LIST

Vers.	Title, author, source, date, status	Identification
0.31	Framework for Agile Software Development in Embedded Systems, Tuomo Kähkönen, v. 0.31	
	Agile Toolbox	
0.5	Agile Assessment Framework, Minna Pikkarainen	
	Guidelines for Applying the Agile Patterns (Elisa Gallo and Teodora Bozheva)	

1. INTRODUCTION

This document describes the agile deployment model that is part of the agile software development framework of the Agile-ITEA project (see Framework for Agile Software Development in Embedded Systems document).

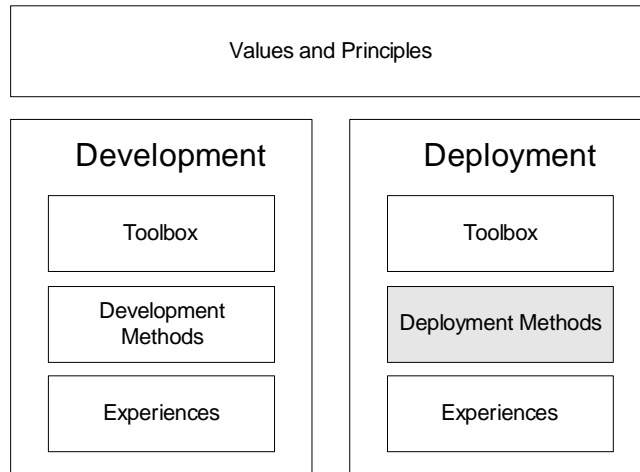


Fig. 1. Framework for Agile Software Development (Kähkönen 2005)

The Agile deployment model contains different methods and procedures for software development organizations to adopt and to adapt new (agile) software development practices, methods, and tools. Many of the deployment practices presented in the model also apply to the traditional mode of software development. However, the agile approach may have influence on how the practice or method should be applied. Thus, emphasis is put on how the deployment is carried out in an agile software development context. Especially Agile Assessment (section 2.1) and Post-Iteration Workshop method (section 2.4) are designed for agile environment. The agility of this deployment model can also be seen in its flexibility of enabling multiple possibilities (both agile and more traditional ways) for adopting and adapting agile practices, methods, and tools in organizations.

The detailed description of agile practices, methods, and tools can be found in the agile toolbox documentation (see Fig. 1).

The Agile deployment model consists of five different phases: 1) agile assessment framework, 2) deployment planning, 3) initial deployment, 4) post-iteration workshop method, and 5) continuous improvement of organizational practices.

	AGILE document template Deliverable ID: D4	Page : 4 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

2. AGILE DEPLOYMENT MODEL

The various agile methods and experiences of their use have been described in several publications in literature, e.g. (Grenning 2001; Greene 2004) (Paetch, Eberlein et al. 2003) (Macias, Holcombe et al. 2003). However, those publications do not provide a sufficient and adequate answer to the six main challenges, which, according to the case experiences, are:

1. How to evaluate the agility of the software product development?
2. How to find the most suitable agile practices to fit the organization's projects?
3. How to pilot the agile practices in organizations?
4. How to improve the agile software development practices?
5. How to tailor agile practices to fit the needs of the projects?
6. How to tailor agile practices, methods and tools to fit the needs of the organization?

The aim of the deployment model is to provide answers to the questions above.

The five phases of the Agile deployment model are: 1) agile assessment framework, 2) deployment planning, 3) initial deployment, 4) post-iteration workshop method, and 5) Continuous improvement of organizational practices. Further information and details on agile assessments can be found in (Pikkarainen 2005).

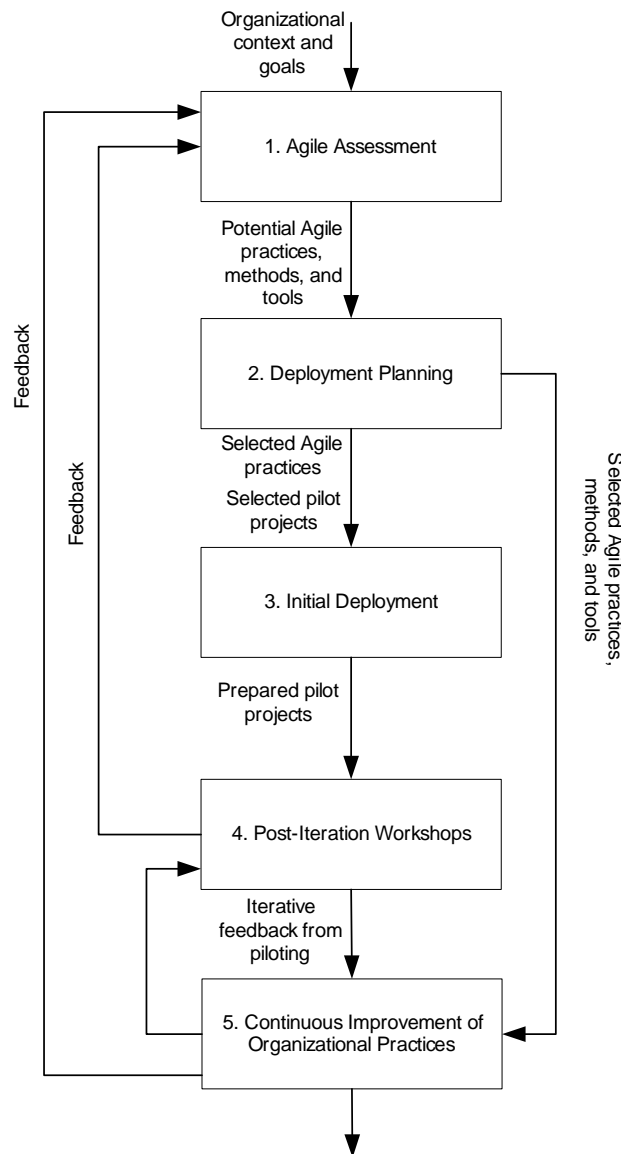



Fig. 2. Agile Deployment Model

Agile assessment provides a way for identifying the agile methods, practices and tools that could be used in an organization (i.e., fit-for-use analysis). The deployment planning contains the selection of the agile practices and planning of the approach for deployment, such as piloting. The initial deployment phase prepares the (pilot) projects to change the way of their daily software development. It includes, for example, training of the project team and preparing the tools of software development. The post-iteration workshops provide means for gaining feedback from the pilot projects iteratively during the software development. The outputs of the agile assessments as well as the post-iteration workshops can be used for continuous improvement of organizational practices.

Fig. 2 illustrates the alternative ways for an organization to conduct agile deployment. In the following, some examples are listed on conducting the phases of agile deployment:

	AGILE document template Deliverable ID: D4	Page : 6 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

- 1) An organization can conduct an agile assessment for finding out potential new Agile practices and validate the selected new methods in pilot projects before deploying them in organizational practices. (Phases 1-2-3-4-5)
- 2) An organization can conduct an agile assessment to find out potential new practices, plan their deployment in an organization and embed them into the organizational practices without validation in advance. (Phases 1-2-5)
- 3) An organization can conduct an agile assessment to find out potential new practices, plan their deployment in an organization and embed them into the organizational practices and validate the process enhancements afterwards. (Phases 1-2-5-4-5)
- 4) An organization can pilot agile practices/processes in their organizational context. (Phases 2-3-4-5)
- 5) An organization adopt post-iteration workshops in their software development projects and, based on the process knowledge from the project teams make continuous improvements on the organizational practices (Phases 4-5 iteratively or Phases 4-1-2-3-4-5 or Phases 4-1-2-5)

Each phase of the agile deployment model consists of steps that define the procedure in more detail. In the following sub-sections, the purpose and content of each phase of agile deployment model in more detail.

2.1 AGILE ASSESSMENT

Many organizations have already utilized agile methods and principles in their software development (Greene 2004; Highsmith 2004). However, few organizations can take a specific agile method (e.g. XP or Scrum) and use it as such. The purpose is rather to apply the most suitable agile practices as a part of the organization's current SW development practices (Manhart and Schneider 2004), based on the organization's needs. This would need evaluation charting the areas of improvement of the organization's current software development and mapping the most suitable agile practices.

Agile Assessment (see research data on (Pikkarainen, Kähkönen et al. 2005; Pikkarainen and Passoja 2005)) is a method for software development evaluation. In the first phase of deployment, it is focused on finding the most suitable agile practices for the SW development in an organization. In the second phase, the aim is on evaluating the utility of the used agile practices in an organization.

2.1.1 Purpose

The purpose of the Agile Assessment is:

1. to analyse the practices, methods, and tools that are currently used in an organization or, for instance, in one project, and to identify how/what agile practices, methods, and tools could be used to make activities more effective.
2. to evaluate the utility of agile practices, methods and tools in use
3. to support the deployment of agile practices, methods and tools.

Agile Assessment (i.e., fit-for-use analysis) can be the first part of deploying agile practices, methods, and tools in an organization. However, it can also be used as a mechanism for gaining evidence on the utility of the used agile methods, practices and tools before their large scale adoption in other projects.

2.1.2 Procedure of Agile Assessment

The agile assessment approach is based on well-known software process improvement paradigms (e.g. QIP(Kan, Basili et al. 1994)) which have a strong theoretical and practical background in improving the project performance. Some elements of Digsoyr and Moe a Participative Approach for Process Assessment (Dybå, Dingsøyr et al. 2004) and Extreme Programming (XP) Evaluation Framework (Williams 2004) are also utilized in the Agile Assessment Framework implementation. The key point in the participative assessment model is the purpose of making both the managers and the employees accept the results of the assessment as relevant and useful. Agile Assessment has the same purposes. It, however, attempts at providing concrete results quickly, using a cyclic evaluation approach. The Agile Assessment Framework consists of the following steps: 1) Focus Definition, 2). Agility Evaluation, 3) Data Collection Planning, 4) Data Collection, 5) Analysis and Workshops and learning phases (Fig. 3).

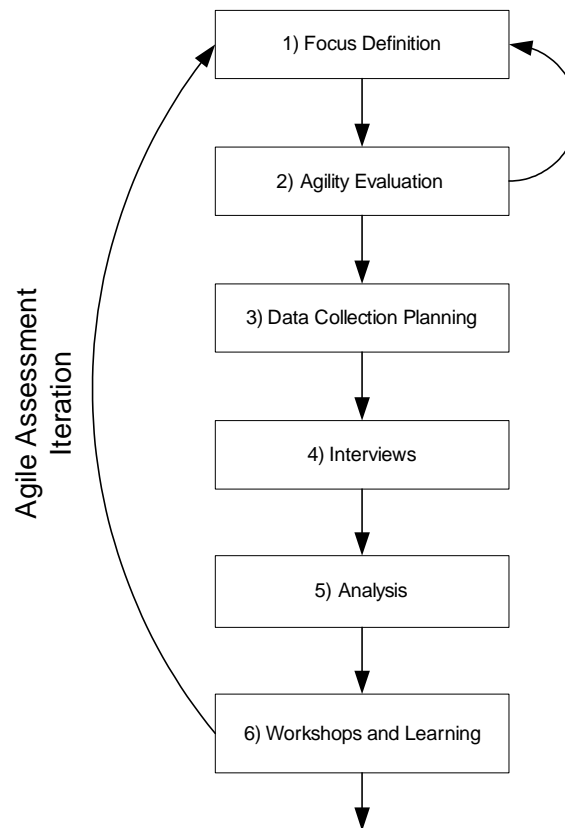


Fig. 3. An Agile Assessment Framework

2.1.2.1 **Focus Definition**

Agile Assessment is started with the focus definition phase which, basically, is not much different from the traditional assessment goal definition phases (e.g. Assessment Initiation in participative approach (Dybå, Dingsøy et al. 2004)). The aims in the focus definition are:

- 1) Clarify the business strategy, goals and their links to the agile assessment. Define the target process/processes for the agile assessment and process owner(s).
- 2) Define the business area for the agile assessment (i.e. 1-n project(s), 1-n team(s), etc.) and people who will be interviewed.
- 3) Define the agile assessment team in the organization (generally, process owners, i.e. who will participate in the analysis session and workshops, contact person in the organization).
- 4) Define the users of the assessment and how the results are going to be used
- 5) Define the scope of the assessment in terms of organizational units

2.1.2.2 **Agility Evaluation**

The second phase of the Agile Assessment is Agility Evaluation. It is a starting point for a deeper agile project assessment giving an appraisal whether the agile methods would be suitable and effective in the evaluated projects. The purpose of the agility evaluation is to give context information that helps the selection of the project for the agile assessment, as well as the comparison between the different agile and plan-driven teams, and explains the variations in results when different teams use similar agile practices.

	AGILE document template Deliverable ID: D4	Page : 8 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

In practice, it is done concurrent with the focus definition. The purpose of this phase is to define what agile methods and principles the organization already uses. The agility level of the evaluated project also affects defining the assessment focus. In the situations where the evaluated project mainly uses the plan-driven (Boehm and Turner 2003) software development practices, the purpose is to define the most suitable agile practices to the evaluated projects and organization context. If the evaluated project has a high agility level and already adopts agile practices, methods, and tools, the goal can be rather to get the evidence on the utility of the used agile practice in the current project situation.

The level of Agility in a project can be evaluated using the Boehm and Turner dimensions, for example. The five factors are the following (Boehm and Turner 2003):

- 1) **Size:** The Number of people of the team,
- 2) **Criticality:** Product's safety criticality (in agile development potential difficulties can be caused by the simple design and lack of documentation),
- 3) **Dynamism:** Degree of requirements and change in technology,
- 4) **Personnel:** The skill and experience of the team, and
- 5) **Culture:** The agile project culture should support the developers' freedom to make the technical solutions for obtaining high motivation for the development team.

2.1.2.3 Data Collection Planning

The Agile Assessment data can be collected using qualitative interviews, collaborative workshops and quantitative metrics. Interviews are planned individually, depending on the goals of the organizations and target project(s) the agility of the evaluated project. Interviews can be based on the assessment approaches such as CMMI or SPICE even if the used agile practices and principles should be included as a part of the structured interview questions. In the situations where the project's agility level is low, the assessment is mainly based on the traditional plan-driven development standards (e.g. CMMI). Instead, in the situations where the agility level of the evaluated project is high (and the project mostly uses agile practices) the interview structure should be mainly based on the agile, not plan-driven software development practices.


According to the case study experiences, also the *collaborative workshop* is a very efficient way to 1) find how/what agile practices, methods, and tools are currently used in an organization, and 2) to define the effectiveness and usefulness of agile practices, methods, and tools in use, 3) define the use of agile practices, and their utility in the different software development phases and iterations. The goal of the collaborative workshop is to find each participant's opinions on the utility of the agile practices in each agile software development phase. The collaborative workshop consists of four steps:

- 1) Workshop preparation
- 2) Data Collection
- 3) Analysis
- 4) Discussion based on interview results

2.1.2.4 Interviews

Interviews (Fig. 2) are based on the existing interviewing techniques (e.g. Focused Interviewing (Merton, Fiske et al. 1956)). The main purpose of the agile assessment is to discuss possible agile practices and their utility in the company. Thus, the agile assessment includes a set of questions which are meant to clarify which agile practices, methods, and tools could be exploited in an organization and how. The questions have been grouped according to the SW development phases and CMMI goals.

The purpose is to have an open discussion on the organizational improvement needs and the benefits of selected agile practice, and problems in the current situation. Questions are focused for each interviewee based on his/her area of domain knowledge. For example, the questions for the project developers are focused on the development steps (iteration planning, iterations), whereas the managers' expertise can be on the project exploitation and iteration planning phases.

	AGILE document template Deliverable ID: D4	Page : 9 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

2.1.2.5 Analysis

The analysis is based on the knowledge of agile practices, methods, and tools and how they have been used in similar organizations or situations before.

The analysis of the most suitable agile practices that may resolve the current problems in an organization requires much background information on the benefits of agile practices in different situations. The required information depends on the agility of the evaluated project and the deployment phase studied.

In the situations where the project agility is low and the goal can be to make a fit-for-use analysis for the agile practices, methods, and tools in general. The analysis is implemented using the following procedure:

- 1) Current state analysis using the CMMI, for example, and agile practices including analysis of current agile practices, methods, and tools in use. Note that only the processes included in the assessment focus are analyzed. The purpose is not to make a CMMI assessment as such but to find the problems in the current state that can be improved using agile practices, methods, and tools.
- 2) Definition of the improvement needs
- 3) Agile practice mapping with the project improvement needs
- 4) Definition of the improvements where the selected agile practices could bring some benefits

In the situation where the agility is high (e.g. the project uses some selected agile method such as XP or Mobile-D) and the goal can be, for instance, to clarify the utility of the used agile practices, the analysis is implemented using the following steps:

- 1) Current state analysis of the used agile practices in different software development phases and iterations
- 2) Efficiency and utility analysis for the current agile practices utilizing the predefined efficiency assumptions

2.1.2.6 Workshops and Learning

The main results of Agile Assessment are defined in the *workshops* (Fig. 4) where the agile practices are analyzed together with the development team.

If the purpose is to collect data, increase the developers learning and analyze the results, the Workshop is implemented in the three main steps which are

- 1) Data collection, where the developers and project managers define agile practices, benefits, problems, traditional practices and new potential practices for the worksheets without discussion.
- 2) Result analysis, which is done together with the workshop team, and
- 3) An interview based evaluation on the suitability (improvements and strengths) of the agile practice which are presented by the agile assessor and discussed together utilizing the analysis results of the previous partitions. An example of the workshop results is illustrated in Fig. 4.

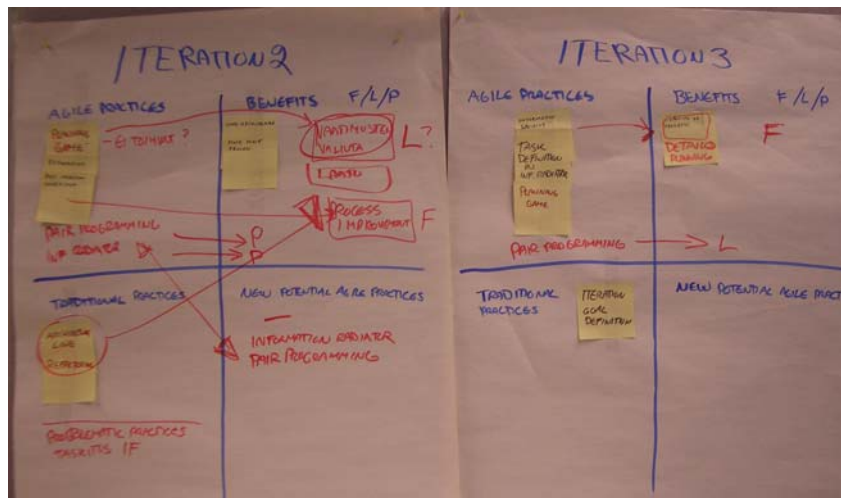


Fig. 4. Agile assessment results of iterations 2 and 3

If the goal of the workshop is to present the interview based agile assessment results, the workshops can be prepared to present possible problem solution alternatives fitted in the organization's current process descriptions and guidelines based on the agile assessment results.

2.2 DEPLOYMENT PLANNING

2.2.1 Purpose

The purpose of the Deployment Planning is to analyze and prioritize the process improvement opportunities in an organization which have been identified, for example, in the agile assessments. Also, the improvements that are to be piloted or adopted in an organization are selected. Another task of deployment planning is to select the deployment approach. For example, the agile assessment results may be taken directly into use in the organization or an organization may prefer piloting of the new practices at first. In this case, the pilot projects should be identified in deployment planning phase.

2.2.2 Procedure

The deployment planning phase consists of steps: 1) Analysis, prioritization and selection of (new) agile practices, 2) Selection of deployment approach, and 3) Selection of pilot projects.

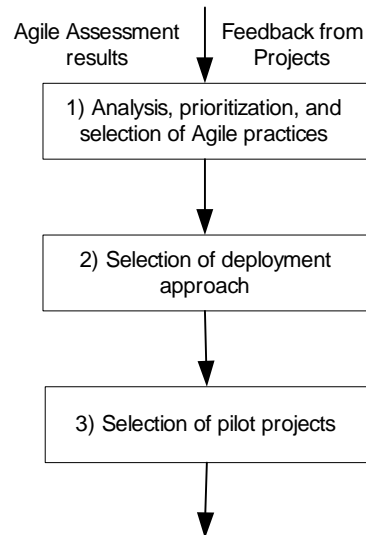


Fig. 5. Deployment Planning Procedure

2.2.2.1 Analysis, Prioritization and Selection

In this phase, the improvement ideas and identified potential agile practices are analyzed and prioritized by organization process managers or at improvement meetings. Then, the most high-priority improvements are selected for the next cycle of organizational process improvement.

2.2.2.2 Selection of Deployment Approach

The selection of the deployment approach means the decision on how the selected agile practices, methods, and tools will be deployed in an organization (see Figure 2). There are several alternatives, such as:

- 1) An organization can conduct an agile assessment for finding out potential new Agile practices and validate the selected new methods in pilot projects before deploying them in organizational practices.
- 2) An organization can conduct an agile assessment to find out potential new practices, plan their deployment in an organization and embed them into the organizational practices without validation in advance.
- 3) An organization can conduct an agile assessment to find out potential new practices, plan their deployment in an organization and embed them into the organizational practices and validate the process enhancements afterwards.
- 4) An organization can pilot agile practices/processes in their organizational context.
- 5) An organization adopts post-iteration workshops in their software development projects and, based on the process knowledge from the project teams makes continuous improvements in the organizational practices.

Especially in small organizations, it may be possible to adopt new practices directly into organizational use based on, for example, agile assessment results. Usually, however, the organizations want to pilot the practices in selected projects before the improvements are taken in the larger scale use.



2.2.2.3 *Selecting of Pilot Projects*

If piloting is selected as a deployment approach in an organization, the pilot projects should be identified at this point. In the selection, it should be taken into consideration what the typical project that would use the piloted practice in the future is.

2.2.2.4 *Planning of Feedback*

The feedback mechanisms should be planned for pilot projects.

2.3 INITIAL DEPLOYMENT

2.3.1 Purpose

Initial deployment phase aims at providing organizations a controlled manner to adopt new practices in (pilot) projects.

2.3.2 Procedure

The initial deployment phase consists of four steps: 1) planning the piloting, 2) tailoring the base process, 3) training, and 4) preparing the software development environment. The projects that will adopt the new (agile) practices have been defined earlier in the deployment planning phase.

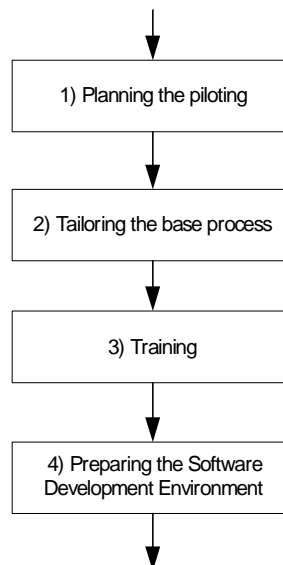



Fig. 6. Initial Deployment phase

2.3.2.1 *Planning the Piloting*

The piloting should be carefully planned for each of the pilot projects that are adopting the selected agile practices. The following issues should be taken into consideration:

- 1) How is the suitability of the method measured (during the piloting)?
- 2) How will the feedback from the (pilot) projects be stored and analyzed?

This Agile deployment model suggests a post-iteration workshop method (2.4) for gaining rapid feedback from the (pilot) projects on how successfully they have adopted and how they have adapted the selected methods during the project. In the “planning of piloting” task, it should be ensured that there are

	AGILE document template Deliverable ID: D4	Page : 13 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

organizational mechanisms available for the project teams to collect and store the relevant feedback in an appropriate format from projects to the organizational level.

2.3.2.2 Tailoring the Base Process

The base process should be tailored for each of the projects that will adopt new practices. In other words, it should be defined on how the adopted practices, methods, and tools will affect the software development process of the project team. Also, the piloting itself may require additions to the daily procedures, such as collection of metrics data in conducting post-iteration workshops, for example. In case there are documented process descriptions available in the project, they should be updated accordingly.

2.3.2.3 Training

The training step aims at introducing the new/tailored practices, methods, and tools for the software development team of the pilot projects. It should be clearly illustrated what is being adopted and how the change will affect the daily work of the project team. Beside the software development practices, the training should also include an introduction to the other alterations in the process, such as collection of metrics and conducting post-iteration workshops in a project.

The training step includes the careful and project specific planning of the training, as well as the training itself. As a result, the software developers should have clear understanding on why their daily work is changed and how the adopted practices will affect their software development process.

Also, coaching (e.g. in XP) can be used during agile software development projects. It aims at providing the software developers additional expertise on the adopted practices whenever required in a project.

2.3.2.4 Preparing the Software Development Environment

The technical environment of the software developers may need to be updated as a result of the new practices, methods, and tools that are being adopted (e.g., new tools and altered document templates). In this phase all the updates needed in the software development environment should be made. This also includes the installation of new mechanisms that are needed (e.g. feedback forms or database tools) for gaining feedback on the piloted practices.

2.4 POST-ITERATION WORKSHOP METHOD

2.4.1 Purpose

One of the principles of agile software development (<http://agilemanifesto.org/principles.html>) proposes that “at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly”. Therefore, an iterative adaptation of the used process is fundamental in agile software development mode. The short development cycles of agile software development provide conveniently continuous and rapid loops for iterative learning, enhancing of the process and for systematic assessment of the effects of the improvements in the projects.

However, this kind of iterative project level learning can also be utilized in adopting and gaining feedback on the software development practices in an organization. For example, a software development project may adopt new (agile) practices, methods, and tools (e.g., test-driven development, pair-programming, etc.) or even entirely new software development process (e.g., XP or Mobile-D). This project can then be used as a pilot project, where the Post-Iteration Workshops method [(hereafter referred as PIW) (see case study results from (Salo, Kolehmainen et al. 2004) (Salo 2005) (Salo 2004) and (Salo and Abrahamsson 2005))] is then used for gaining feedback on the adopted practices, for example on:

- 1) how valuable/suitable the individual practices are found (in the specific project)
- 2) how the adopted practices can be improved (to better suit the specific project)

By adopting the PIW method more widely in the organizational software development process, valuable knowledge, and validated improvement opportunities and insights of various projects can be gained in order to continuously improve the practices beyond single projects, i.e. organizational software development

process. Also, for the individual project teams the PIW method provides an opportunity to adapt the base process to better suit its needs and, within organizational limitations, change the way the daily work is conducted.

In summary, it can be said that PIW method has two main purposes:

- 1) Adapting and improving the practices of an individual project team
- 2) Gaining process knowledge/feedback from projects to be utilized on the organizational level

2.4.2 Procedure

The PIW method consists of six steps: 1) preparation, 2) collection of experience, 3) planning of improvement actions, 4) piloting, 5) follow-up and validation, and 6) packaging. Each step recurs iteratively during an agile software development project. The preparation step is conducted at the beginning of the project and prior to each PIW session. The actual PIW session consists of collection of experience, planning of improvement actions, and follow-up and validation steps. This event is situated at the beginning of the iteration, when the project team gathers together to discuss the problems and obstacles in their daily software development and to seek solutions for improving the way their daily work is conducted. The improvement solutions are then piloted in the following iteration to experiment and validate the agreed improvement actions. Finally, the packaging step includes the storing of the results of each PIW session for both project and organizational purposes.

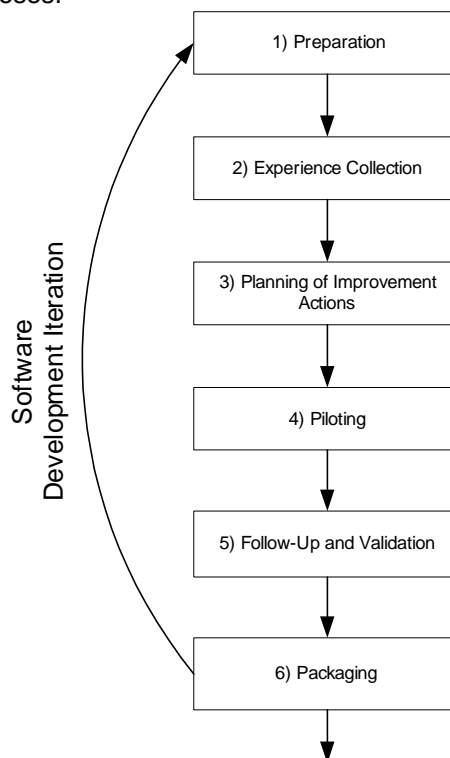



Fig. 7. The Six Steps of PIW method

Naturally, if a PIW is conducted at the end of the last iteration, the piloting as well as the follow-up and validation steps cannot be conducted. In this case, the workshop provides feedback directly to the organizational level that can then pilot and validate potential improvement suggestions in other projects later on.

The central stakeholder of the PIW method is the software development team, who is in the vital role of putting all their knowledge and experience into improving and adapting their software development process. Also, they are the ones who will gain the immediate benefits from the workshops in the form of improved

	AGILE document template Deliverable ID: D4	Page : 15 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

daily working practices. The PIW session is lead by a facilitator who is in charge of keeping the session as effective as possible to save the valuable time of the software developers for their main task. The facilitator also has multiple tasks before, during and after the actual PIW sessions. He/she leads the preparation of the workshops, managing the PIW session and packaging the results. The facilitator should be someone outside of the project team, e.g. from the quality department of an organization, as one of his/her important duties is to be a vital software process improvement link between the organizational and project levels. The facilitator also needs to have real authority and knowledge of the work in two ways: to provide the organizational limits for tailoring the process and other relevant process knowledge for the project team in its process improvement work, as well as the ability to ensure the information flow from the PIWs to the organizational learning activities. Also, a secretary is needed for noting down the important outputs during the PIWs. In practice, this responsibility will likely fall to one of the project team members.

2.4.2.1 Preparation

The preparation step includes, firstly, the preparation needed prior to the project that will conduct the series of PIWs. The Agile software process model with iterative software development cycles provides a suitable rhythm for holding PIW sessions in a project. Thus, the density and the schedule of the workshops should be planned beforehand. Also, if one purpose of holding the PIWs is to gain feedback for the organizational level (e.g. piloting of new practices), the feedback and packaging mechanisms for the organizational level should be carefully defined at this point. The preparation should also include careful planning of the workshop itself to ensure its suitability and effectiveness in the given context. Various techniques may be used in different steps of PIW depending on different factors such as the composition and size of the software development team (Kerth 2001).

During the project, the preparation task includes the setting of the equipment for the PIW session and preparation other workshop material, such as the analysis and visualizations of the metrics data to be used in the "follow-up and validation" step. However, the preparation task may also include improving the PIW session itself by tailoring the selected PIW techniques during the project.

2.4.2.2 Collection of Experience

The collection of experience step is the first activity of the actual PIW session that takes place after a completed iteration. The central idea of PIWs is to base the process improvements on the obstacles and problems that are identified by the software development team. A few techniques in agile software development context have been suggested to derive experiences from the project team and generating them into process improvements. For example, Dingsøy and Hanssen (Dingsøy and Hanssen 2002) suggest the KJ method (Scupin 1997) for collecting the positive and negative experiences on flap-sheet and structuring them separately into labeled groups for further analysis (Fig. 8).

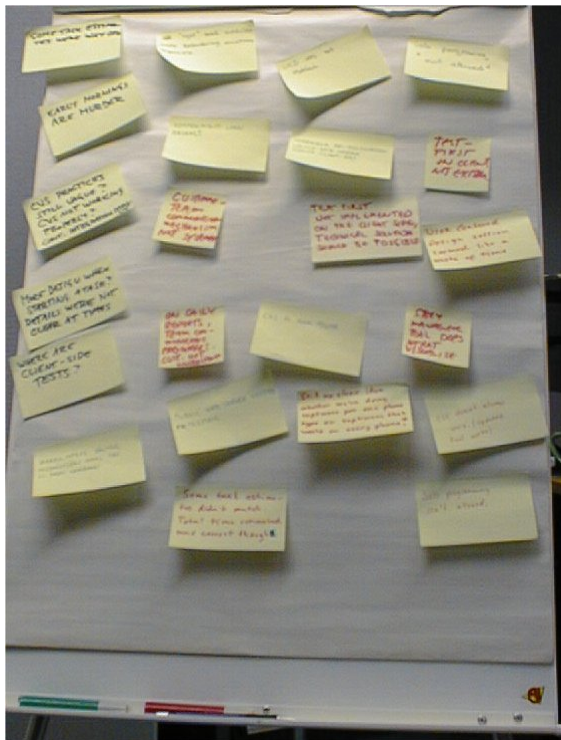


Fig. 8. Grouping of findings

The groups generated from negative findings provide means for visualizing different problem areas and can be, later on, used as a tool in finding solutions. Also, if the process knowledge of the PIWs is to be utilized on the organizational level, the grouping plays an important role in classifying and filtering the information later on. Thus, the categories for grouping should be organizationally fixed.

It should be noted, however, that the project's successes and problems may also have been encountered by other organizational stakeholder groups. The PIW session is a possible place to discuss and analyze this feedback as well, providing that there are established structures to this knowledge flow.

2.4.2.3 Planning of Improvement Actions

Improvement actions for an incipient iteration are planned in a PIW session after the obstacles and problems have been identified. Thus, the starting point for planning the improvement actions is the set of negative experiences of the project team from the previous iteration that are grouped into problem areas. The domain knowledge of the software developers is then used to formulate useful and functional process enhancements for the incipient iteration.

Different techniques, such as Root Cause Analysis (Bergman, Fundin, et al. 2002) or focus group technique (Kerlinger and Lee 2002) can be chosen for finding the causes and solutions for problems that have emerged. Focus group technique is based on a free and open discussion led by a facilitator. Its advantages have been reported to be low costs and quickness in generating the ideas of the respondents, as well as flexibility and effectiveness (Kerlinger and Lee 2002). The facilitator – regardless of the selected problem solving technique - plays an important role in leading the way of the software development team to effectively use their knowledge in formulating feasible and collectively agreed solutions to be tried out in the next iteration. When selecting a technique, it should be considered whether the goal is to focus on a few central problems or cover the problems that have emerged more widely. Also, it should be noted, that as new knowledge and team learning emerge during the problem solving session, new problems may come up.

In order to pilot the agreed process improvements during the software development activities of the incipient iteration, they first need to be explicitly defined. Issues, such as 1) the exact problem, 2) the concrete action

to be taken to improve the situation, and 3) the responsibilities and schedule of carrying out the improvement action, should be defined. Preferably, the improvement actions should be small enough to implement in the beginning iteration. Also, the software development team should collectively agree on the planned improvement actions and the facilitator should confirm that they are within possible organizational limitations for the process tailoring.

Furthermore, each improvement action should include a plan on how its implementation is followed-up and validated after piloting. The means for validating each process enhancement should be carefully planned. Often, the enhancements may be validated using merely the qualitative data and relying on the experiences and knowledge of the project team. However, also quantitative validation may be in order. When considering quantitative validation issues, such as the availability of the required metrics, organizational requirements, and effort needed for collecting additional metrics comparing to the added value it brings, should be taken into consideration. Different techniques, such as GQM (Basili 1994) may be used for defining the metrics needed for quantitative validation. Also, the responsibilities and schedules for validation tasks (e.g., data collection and analysis) should be defined at this point.

A structured document template (Table 1) [see also (Salo 2005)] can be used in supporting the planning, managing and storing of the improvement actions in PIWs. The goal of the template is to provide guidance for the facilitator and the project team on the issues to be considered for every agreed improvement action. The document is filled out by, for example, the secretary of the PIW session and revised and re-revised by the facilitator after the session. It should be possible for the project team members to ensure the implementation of the action points. The action point list provides the organizational level with feedback on how the project teams have adapted their practices.

Table 1. Action Point Template


Problem Area: Planning and Estimation				
Problem:	Action Point	Actor	Validation plan	Validation
Task estimations too inaccurate in release 1. => release delayed.	Analyze task data to evaluate the cause of delays in each task.	Tracker	Team interprets the data in the next PIW to find ways to improve.	Data revealed too big and indistinct tasks. NEW AP =>.
Too big and indistinct tasks.	Task will be split to max. 4 hour size.	Project Manager	Interpret the metrics data in the next PIW to see if effort estimations improved.	Smaller task size improved the effort estimation. 4 hours is a good size for task when possible.
	Analysis of effort estimations	Tracker		

The basic idea of the template was first to link the improvement actions to the corresponding problem areas (i.e. "Problem Area" field) that were formed in the grouping of the negative experiences. The "Problem" field then specifies the problems that emerged when discussing the negative experiences within the specific problem area and withholds the summary of an underlying problem. The aim was to provide sufficiently detailed, explicit knowledge of the issue so that it could also be understood by other readers of the document. "Action point" field should clearly define the specific action to be taken in the next iteration to solve the problem, whereas the "Actor" defines the responsibilities for the task. "Validation plan" describes how and when the follow-up and validation for the action point will be conducted and the "Validation" –field, the results of the planned validation after piloting (i.e., updated in the ensuing PIW).

2.4.2.4 Piloting

The piloting step is conducted during the software development iteration. It includes the implementation of the plans for process improvement, the collection of measurement data for validation of process improvements and the preparation of the data analysis needed for validation purposes.

In the Agile learning mode, the planned improvement actions should be piloted as soon as possible, preferably in the beginning iteration. This makes the software developers' effort put on the software process improvement activities visible and increases the motivation and satisfaction of the software developers both in their daily work and in the improvement itself (Salo 2004).

	AGILE document template Deliverable ID: D4	Page : 18 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

2.4.2.5 Follow-Up and Validation

The follow-up and validation of the piloted process improvements are done during a PIW session. This needs to be done to ensure if each improvement action has been taken place as planned or if it needs to be postponed to the next iteration. Also, the effectiveness of each of the piloted process improvements should be systematically validated for various reasons. First, the project itself should control whether the process actually is improved and take further actions accordingly. Secondly, the organizational level needs the information on how and why the individual projects adapted their base process, and how successful the enhancements were in the given context. The validation should be conducted using the experiences of the project team (qualitative validation) and analyzed metrics (quantitative validation) as planned for each improvement action.

2.4.2.6 Packaging

The systematic storing of the PIW results should be done regarding the purposes of both project and organizational levels. For one, the project team needs to record, for example, the process improvement plans to manage the implementation and validation of the SPI actions throughout the project. The action point template (see section 2.4.2.3) may be used for storing the PIW knowledge. However, depending on the organizational needs, the storing, filtering and analyzing of the extensive PIW knowledge may require also other mechanisms, such as database storages.

Also the organizational level requires feedback on piloting and adapting the software development practices in projects i.e., why, what, when, how and how successfully different project teams in specific project context have improved and adapted their process iteratively. Thus, the organizational level needs the packaged PIW knowledge, as defined in the knowledge creation theory (Nonaka and Hirotaka 1995), for formalizing the knowledge of individual project teams and transforming it into explicit knowledge in order to create new understanding (Nonaka and Hirotaka 1995). For example, Basili suggests a use of experience base that “provides a mechanism for evaluating the recorded experience, helping us to decide what and how to reuse, tailor and analyze” (Basili 1989, p. 475). Thus, if an organization applies PIWs systematically in multiple projects and wants to ensure an effective utilization of the knowledge it provides, an organizational knowledge system for the projects to store their PIW results iteratively should be established.

The storing of the data and knowledge from a PIW session should be conducted immediately after the PIW session when the facilitator still remembers the important details well. However, the validation data for the planned improvement actions should be added in the PIW session after piloting.

2.5 CONTINUOUS IMPROVEMENT OF ORGANIZATIONAL PRACTICES

2.5.1 Purpose

The continuous improvement of organizational practices is done to make sure that the new (agile) practices that have been found useful are employed in the organization. Also, the goal is to continuously seek new ways to improve the software development in an organization by adopting, adapting and empirically evaluating new practices.

2.5.2 Procedure

The phase of continuous improvement of organizational practices consists of four steps: 1) analyze the feedback, 2) select/adapt organizational software development practices, 3) update organizational processes, and 4) disseminate the new/altered practices to projects (Fig. 9).

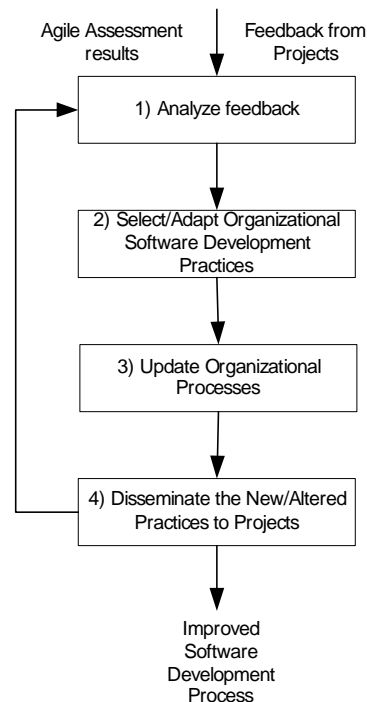


Fig. 9. Steps of Continuous Improvement of Organizational Practices

2.5.2.1 Analyze Feedback

In this agile deployment model, the organizational level gains feedback for adopting and adapting the existing software development process from two directions:


- 1) Agile assessments
- 2) Individual projects (post-iteration workshops)

This knowledge should be carefully analyzed in order to find any potential improvements that can be made in the organizational software development process. The improvement can mean adopting a totally new tool or practice in the organization's projects but also adapting the existing software development practices, methods, and tools.

2.5.2.2 Select/Adapt Organizational Software Development Practices

Evolving the organizational processes is a task that should be done gradually in small steps. The software process improvement is a continuous process where potential improvement opportunities are identified, prioritized and selected in a cyclic manner. Thus, the software process improvements with high priority should be selected to be embedded in the organizational software development practices for the next improvement cycle.

The agile practices found need also to be tailored to suit the organization processes and development practices. For example, an electronic process guide is an effective way to make the created process models, examples and templates available for the software development projects (Dybå, Moe et al. 2004; Kurniawati and Jeffrey 2004). Still, some of the main problems in the software process improvement (SPI) work are how to create commitment towards SPI involving all parts of the organization (Arent and Iversen 2000), and how to develop the working process guides which meet the needs of different software development projects (Arent and Norbjerg 2000). To overcome these challenges Dingsøy and Moe (Dingsøy and Moe 2004) have proposed a process workshop, which is a promising approach to define an electronic process guide in

	AGILE document template Deliverable ID: D4	Page : 20 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

small software companies. It is a light agile way to develop processes based on the organization's needs, offering rapid feedback from the future process users and face-to-face communication between the experienced developers and the improvement team (Fowler and Highsmith 2001). Rapid7 (Kylmäkoski 2003) is another promising approach for the electronic process guide development. Its purpose is to define the ready documents in several workshops together with the relevant stakeholders. The Rapid7 method has two key benefits compared to the Dingsøy and Moe (Dingsøy and Moe 2004) Process Workshop Approach. They are the deeper result analysis and the aim at developing the final version of the document (e.g. electronic process guide) during the workshops.

2.5.2.3 Update Organizational Processes

Once the organizational process improvements have been selected, their effects on the existing processes and practices should be carefully identified. The updates should be done accordingly to, for example, the organizational process descriptions, tools, and templates.

2.5.2.4 Disseminate the New/Altered Practices to Projects

It should be ensured that the software development projects are aware and familiar with the new or adapted practices that they should adopt in their daily software development. Depending on the organization and the size of the process improvement, this may require, for example, training of the software developers.

3. CRITICAL SUCCESS FACTORS IN ADOPTING AGILE PRACTICES

The text of this chapter is taken on permission of ESI from "Guidelines for Applying the Agile Patterns" document (Gallo and Bozheva 2005). Gallo and Bozheva have identified following critical success factors in application of the agile practices:

- 1) Developers. Most developers respond to the proposed introduction of an agile process with the appropriate combination of skepticism, enthusiasm, and cautious optimism. Other developers, however, either resist the change or overzealously jump into the project without enough forethought. Either reaction can cause problems. The best situation is when the piloting team is formed by experienced and motivated developers and the project has a reasonable duration (6-8 months) allowing the demonstration of the results and analysing the pros and cons of the new process in a relatively short time.
- 2) Testers. Writing source code is the primary activity in any development process, but it is especially important in agile processes. Therefore, the agile processes do not have separate coding and testing phases and the implementation and testing patterns are that much interrelated. The testers, if different from programmers, work more closely with them earlier in an agile process than in other processes. Thus, the testers and other nonprogrammers must be very carefully integrated into any agile project in which they participate.
- 3) Two issues have to be kept in mind, if the tester team is different from the programmers. First, writing a unit test from someone else may result in a useful test, but it almost certainly loses some of the white-box advantages inherent in self-written unit tests. Second, if a tester knows enough about programming and you need another programmer, hire the tester.
- 4) Management. The upper-management concerns generally fall into four categories: How can we promise new features to the customers? How can we track progress? How will the agile process impact the other groups? When does the project end?
- 5) Many managers are reluctant to surrender the feeling of control that, for instance, a Gantt chart gives them, thus they often present challenges to teams/organizations who wish to move toward an agile process.
- 6) Since the management support is extremely important for the success of an agile team, the only advice to hesitating managers is "Try it out for a while, collect record on on-time delivery, and let's see".
- 7) Customer. When a product is developed for a number of customers, it is helpful to assign the role of the customer's representative to a team member. This customer's representative will be responsible for meeting the real customers and discussing the status of the software developed and features to be further implemented with them. However, it has to be taken into consideration that meeting several customers located in different places takes time, during which the customer's representative is not present in the office, i.e. he/she cannot make a pair with another developer, if necessary. A possible solution is that the customer's representative does not

	AGILE document template Deliverable ID: D4	Page : 21 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

participate in the team as a developer. Nevertheless, it is important that the customer's representative takes part in the development team meetings at least once a week to avoid letting the development go in the wrong direction.

- 8) When a customer does not commit him/herself sufficiently in the project, someone from the development company should act as internal customer, otherwise most efforts will be likely to fail.
- 9) Team. The application of the agile practices requires intensive communication and team work. The co-located teams typically make decisions much more quickly than the distributed ones. Thus, the organizations should resist distributed development at least for the first few months after initiating an application of agile patterns.
- 10) If distributed teams must be combined, bringing as many people as possible together for the first week or two of the project can increase the likelihood of success. During this joint work, the team members have to define the key software product components and the interfaces between them, and to distribute the work between the teams in such a way as to minimize the dependencies between them.
- 11) Barry Boehm's principle of top talent, "use better and fewer people", is central to an agile process. Although the difference in productivity between the best and the worst programmers on a team may exceed the documented ratio of 10:1, the productivity difference matters most when the programmers are working on tasks essential to the software delivery. Therefore, the members of teams applying agile patterns should have about the same level of experience and new developers have to be involved in projects gradually.

4. CONCLUSIONS

Agile SW development and agile methodologies [e.g. XP (Beck 2000) or Scrum (Schwaber and Beedle 2002)], used to improve organization or project team ability to manage projects, have been widely discussed in literature (Greene 2004). Changing customer requirements, dynamic market situation and new technical challenges generate more demands for the product development (Beck 2000). Therefore, the agile principles (Beck, Beedle, et al. 2001) and practices (such as XP Planning Game (Beck 2000), and the rapid incremental process cycles can be seen as one solution for efficient product development.

The various agile methods and experiences of their use have been described in several publications in literature, e.g. (Grenning 2001; Greene 2004) (Paetch, Eberlein et al. 2003) (Macias, Holcombe et al. 2003). However, those publications do not provide a sufficient and adequate answer for the six main challenges, which, according to the case experiences are:

- 1) How to evaluate the agility of the software product development?
- 2) How to find the most suitable agile practices to fit the organization projects?
- 3) How to tailor agile practices to fit the needs of the projects?
- 4) How to pilot the agile practices in organizations?
- 5) How to improve the agile software development practices?
- 6) How to tailor agile practices, methods, and tools to fit the needs of the organization?

The aim of the Agile Deployment Model is to provide software development organizations with means to answer the above questions. The five phases of agile deployment model are: 1) agile assessment framework, 2) deployment planning, 3) initial deployment, 4) post-iteration workshop method, and 5) continuous improvement of organizational practices. The phases of Agile Deployment Model can be applied in various sequences in order to enable various alternatives for adopting agile practices, methods, and tools in an organization.

Agile assessment provides a way to identify the most potential agile practices, methods, and tools in an organization (i.e., fit-for-use analysis). The deployment planning includes the selection of the agile practices for piloting and identifying the projects that will adopt the selected practices. Also, the mechanism for collecting feedback from the pilot projects should be planned at this stage. Initial deployment phase prepares the (pilot) projects to change the way of their daily software development. It includes, for example, training of the project team and preparing the tools for further software development. Post-iteration workshops provide means for gaining feedback from the pilot projects iteratively during the software development. The outputs

	AGILE document template Deliverable ID: D4	Page : 22 of 24
		Version: 1.0 Date : 04.04.05
		Status : Final Confid : Public

of the agile assessments as well as the post-iteration workshops can be used for continuous improvement of the organizational practices.

5. REFERENCES

- Arent, J. and J. Iversen (2000). Project Assessments: Supporting Commitment, Participation, and learning in Software Process Improvement. 33 rd Hawaii Ubternational Conference on System Sciences.
- Arent, J. and J. Norbjerg (2000). Software process improvement as organizational knowledge creation: a multiple case analysis. System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference, IEEE CNF.
- Basili, V. R. (1989). Software Development: A Paradigm for the Future. COMPSAC '89, Orlando, Florida.
- Basili, V. R. (1994). The Goal Question Metric Approach. Encyclopedia of Software Engineering, John Wiley & Sons, Inc. 2: 528-532.
- Beck, K. (2000). Extreme Programming Explained: Embrace Change, Addison Wesley Longman, Inc.
- Beck, K., M. Beedle, et al. (2001). Manifesto for Agile Software Development. **2002**.
- Bergman, B., A. Fundin, et al. (2002). Beyond Root-Cause Analysis. Annual Reliability and Maintainability Symposium, The International Symposium on Produc Quality and Integrity, Seattle, WA, IEEE.
- Boehm, B. and R. Turner (2003). Balancing Agility and Discipline. Balancing Agility and Discipline -A Guide for the Perplexed, Addison Wesley.
- Dingsøy, T. and G. K. Hanssen (2002). Extending Agile Methods: Postmortem Reviews as Extended Feedback. 4th International Workshop on Learning Software Organizations (LSO'02)), Chicago, Illinois, USA.
- Dingsøy, T. and N. B. Moe (2004). The Process Workshop: A Tool to define Electronic Process Guides in small software companies. Software Engineering Conference, 2004, Australian, IEEE MCF.
- Dybå, T., T. Dingsøy, et al. (2004). Process Improvement In Practice, A Handbook for IT Companies. Trondheim, Kluwer Academic Publishers.
- Dybå, T., N. B. Moe, et al. (2004). An empirical Investigation of Factors Affecting Software Developer Acceptance and Utilization of Electronic Process Guide. 10 th International Symposium on Software Metrics (Metrics'04).
- Fowler, M. and J. Highsmith (2001). "Agile Manifesto." Software Development.
- Gallo, E. and T. Bozheva (2005). Guidelines for Applying the Agile Patterns, ESI (European Software Institute).
- Greene, B. (2004). Agile Methods Applied to Embedded Firmware Development. Agile Development Conference, Salt-Lake city.
- Grenning, J. (2001). "Launching XP at a Process-Intensive Company." IEEE Software **18**: 3-9.
- Highsmith, J. (2004). Agile Project Management, Creating innovative products, Addison-Wesley.
- Kan, S. H., V. R. Basili, et al. (1994). "Software Quality: An Overview from the perspective of total quality management." IBM Systems Journal **33**(1).
- Kerlinger, F. N. and H. B. Lee (2002). Foundations of Behavioral Research, Harcourt College Publishers.
- Kerth, N. L. (2001). Project Retrospectives: A Handbook for Team Reviews, Dorset House Publishing.
- Kurniawati, F. and R. Jeffrey (2004). The long-term effwcts of an EPG/ER in small Software organization. ASWEC, Australian Software Engineering Conference, Australian, IEEE Computer Society.
- Kylmäkoski, R. (2003). Efficient Authoring of Software documentation Using RaPiD7. ICSE 2003, IEEE.
- Kähkönen, T. (2005). Framework for Agile Software Development in Embedded Systems, Nokia: 44.
- Macias, F., M. Holcombe, et al. (2003). A Formal Experiment Comparing Extreme Programming with Traditional Software Construction. The Fourth Mexican International Conference on Computer Science, Mexican.
- Manhart, P. and K. Schneider (2004). Breaking the Ice for Agile Development of Embedded Software: An Industry Experience report. 26 th International Conference of Software Engineering.
- Merton, R. K., M. Fiske, et al. (1956). The Focused Interview, The Free Press.
- Nonaka, I. and T. Hirotaka (1995). The Knowledge-Creating Company, Oxford University Press, Inc.
- Paetch, F., A. Eberlein, et al. (2003). Requirements Engineering and Agile Software Development. Twelfth IEEE international workshop on Enabling Technologies:Infrastructure for Collaborative Enterprices, Computer Society.
- Pikkarainen, M. (2005). Agile Assessment Framework, VTT Technical Research Centre of Finland.
- Pikkarainen, M., T. Kähkönen, et al. (2005). Agile Project Assessment Using an Extreme Evaluation Framework -A Case Study. Under review in Agile 2005, Conference.



AGILE document template
Deliverable ID: D4

Page : 23 of 24

Version: 1.0
Date : 04.04.05

Status : Final
Confid : Public

- Pikkarainen, M. and U. Passoja (2005). An Approach for Assessing Suitability of Agile Solutions:A Case Study. The Sixth International Conference on Extreme Programming and Agile Processes in Software Engineering, Sheffield University, UK.
- Salo, O. (2004). Improving Software Process in Agile Software Development Projects: Results from Two XP Case Studies. EUROMICRO 2004, Rennes, France, IEEE Computer Society Press.
- Salo, O. (2005). Learning in Agile Software Development Environment. 7th International Workshop on Learning Software Organizations, Kaiserslautern, Germany.
- Salo, O. and P. Abrahamsson (2005). "A Post-Iteration Workshop Approach for Agile Software Process Improvement: Implications from a Multiple Case Study." Under Review Process.
- Salo, O., K. Kolehmainen, et al. (2004). Self-Adaptability of Agile Software Processes: A Case Study on Post-Iteration Workhops. 5th International Conference on Extreme Programming and Agile Processes in Software Engineering (XP 2004), Garmisch-Partenkirchen, Germany, Springer.
- Schwaber, K. and M. Beedle (2002). Agile Software Development With Scrum. Upper Saddle River, NJ, Prentice-Hall.
- Scupin, R. (1997). "The KJ Method: A Technique for Analyzing Data Derived from Japanese Ethnology." Human Organization **56**(2): 233-237.
- Williams, L. A. (2004). "Extreme Programming Practices: What's on Top." Agile Project Management Advisory Service **5**(12).