How can you be Agile in "Rough Terrain" and under "Tight Boundary Conditions"

Industrial Experience Report

Peter Faßbinder
Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany
peter.fassbinder@siemens.com

Keywords: Agile Development, Large Systems, "Hybrid" Projects, Agile Requirements Management, Agile Testing, Agile Roles, Process Integration, Tailoring Concepts, Industry Experience.

Abstract: Ten years after the publication of the agile manifesto, the following statements still hold true: Defining an agile development process for a small co-located software team is straight forward; there are many theories, models and examples for this. However, integrating agile development into an overall standard process of a complex organization, that includes hardware and system development, and has large distributed projects, is still a major challenge: How do you integrate the agile development approach into the standard process? How much agility do you have to abandon to satisfy the boundary conditions of such an environment? What is the ideal process architecture to address the needs of the different project types? What compromises do you have to accept and where are the limits that you should not cross? This work provides possible answers to these questions, and describes suitable approaches to address the three key challenges faced when integrating agile development into a standard system development process. The results are based on experiences from many agile implementation projects within the Siemens AG.

1 INTRODUCTION

Last year the agile manifesto (Beck K. et. al., 2001) celebrated its tenth birthday. It condensed the essential ideas from lightweight development approaches as e.g. Extreme Programming (Beck K., 1999), Scrum (Schwaber and Beedle, 2001), and others.

While the agile development approches were originally designed for smaller development projects, many interpretations and case studies for scaling agile approaches have been reported over the past years (see e.g. Larman, Vodde, 2008; Larman, Vodde, 2010; Canditt S. et. al., 2010). Nevertheless, there is no "one size fits all" theory for using agile approaches within large projects (i.e. multiple subprojects) or complex organizations.

This work describes the experiences with using agile development approaches in large, heterogeneous, potentially distributed system development projects; in organizations, where the variety of project types require agile and "traditional" development (i.e. non agile approaches such as the waterfall or V-Model) to go side by side, sometimes even within one project. It presents the most important aspects encountered when deploying agile development in such complex and heterogeneous environments, and discusses the associated challenges and suitable solutions.

The following sections describe the three major challenges faced in this context in more detail. Based on the experiences from many agile implementation projects, these key challenges are:

- How can the agile development approach be integrated into the overall process landscape of the enterprise without jeopardizing the traditional boundary conditions? This is the process challenge.
- What are suitable approaches to integrate system requirements and system test within the agile mode of handling requirements and testing? This is the requirements and test challenge.
- How can agile roles be integrated into the overall organizational process with its many traditional roles? This can be labelled the role challenge.

2 THE PROCESS CHALLENGE

Large organizations within the industry are faced by

Faßbinder P..
How can you be Agile in "Rough Terrain" and under "Tight Boundary Conditions" - Industrial Experience Report.
DOI: 10.5220/0003970901170121
In *Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering* (ENASE-2012), pages 117-121
ISBN: 978-989-8565-13-6

many requirements that define the boundary conditions for their processes. Thereby, multiple external and internal stakeholders have to be considered in addition to the team of a specific project.

2.1 Process Stakeholders

Within industry, legislations, norms, and standards might require a standard process that covers a comprehensive set of process areas and satisfies specific safety criteria. The enterprise tool landscape might impose certain constraints on the data that needs to be provided by projects. Headquarters or higher organizational level might expect a specific form of planning and reporting.

On the other hand, process manager and process users need an efficient process that is easy to use and easy to maintain. They expect continuous incorporation of experiences and lessons learned from other projects within the organization. Since the process might be applicable for thousands of employees and hundreds of projects in large development departments, this poses a major challenge.

2.2 Boundary Conditions and Agile Principles

Due to the multiple stakeholders and diverse nature of development projects within a large, complex organization, creating suitable processes is a challenging task.

There will be projects where a transition to agile development is not possible or is not sensible. These can be, e.g., long running product lines where only minor maintenance is done, or hardware development, where short iterations are hardly achievable.

Or there might be large system projects with multiple subprojects, of which some want to develop agile to increase their efficiency, while other subprojects are not suitable for this approach. Also for these "hybrid projects" efficient process solutions have to be provided.

2.3 Integration Concepts

Faced with the challenge to provide the organization with a process framework that supports agile development, traditional development and hybrid projects, one can distinguish three different implementation approaches:

- Creation of a separate process for agile development (i.e. an agile process variant). Thereby, the traditional process remains unchanged.
- Agile development is described in a guideline as an add-on to the traditional process description. Also in this case the traditional process itself remains unchanged.
- Integration of agile development within the traditional process, i.e., the creation of one process with suitable tailoring mechanism that covers agile and traditional development approaches.

2.3.1 Process Variant for Agile Development

If you create a separate process for agile development, the impact on the traditional development process is minimal, since it will remain basically unchanged. On the other hand, the process for agile development can be optimized for agile and does not have to consider aspects from traditional development projects.

Benefits of this approach are especially the easy navigation for users of both process variants. The variants can be optimized and streamlined for their specific purposes. There are no ambiguities and no further agile tailoring aspects within the processes.

The disadvantages of two separate process variants for agile and traditional development are especially:

- Hybrid projects are not addressed. They have to define their own interpretation of the processes and how to combine them within their project.
- Similarities between the traditional and agile development process are not emphasized. Standardization and shared lessons learned across both variants are difficult to manage.
- The maintenance of the processes requires a high effort since two variants have to be kept up to date.

2.3.2 Agile Development as a Guideline

Another approach is the creation of a guideline for agile development. This means, the traditional development process of the organization remains valid also for agile development. All necessary adaptations to implement the agile principles are described in the guideline and have to be treated as an ad-on to the traditional process.

The advantages of this approach are especially its easy implementation and the unchanged traditional development process.

There are, however, numerous disadvantages

associated with this approach:

- The challenge to integrate the agile development into the overall process is transferred from the process team to every agile project team. This will result in multiple effort, interpretation variants, and missing integration of best practices.
- There is ample room for ambiguity within the agile projects. There is no guarantee, that mandatory elements from the standard process are implemented within an agile project.
- Agile roles are not defined in the standard process description, i.e., there are no officially defined responsibilities for agile.
- The agile development is not very visible within the process framework of the organization. This signals that traditional development remains the standard.

2.3.3 Integration of Agile and Traditional Process

The third implementation approach is the creation of one common process for all types of development projects, which incorporates traditional and agile streams. This approach requires a suitable process architecture and tailoring concept.

In this approach, the tailoring criteria can be implemented on three different levels within the integrated process (the recommended solution depends on the amount of differences within the workflows and activities):

- As agile workflows that are variants of a complete traditional workflow.
- As agile variants of activities or activity chains within a workflow.
- As tailoring options within an activity that differentiate e.g. between outputs or methods for agile and traditional projects.

In this approach, the similarities between the traditional and agile development process are emphasized. The workflows and the process elements, that are identical for agile and traditional development, have to be maintained only once. And hybrid projects have a clear, consistent basis to manage the overall system project and the individual subprojects.

The disadvantage is that this approach requires considerable effort for its implementation. The agile and traditional processes have to be integrated, based on a suitable process architecture and tailoring concept. If this is not done systematically, the workflow variants and tailoring options can make process navigation cumbersome.

2.4 Recommended Approach

The recommended integration approach depends on the objectives, the boundary conditions, the project types, and the transition strategy of the affected organization.

The preferred solution is typically the integration of the agile and traditional process. However, this is not the must suitable solution for all situations. If one has only a small number of agile projects, a guideline for these types of projects can be more appropriate. Also a guideline could be a sensible first step towards a more thorough transition to agile development.

On the other hand, if agile and traditional projects are clearly separated, and hybrid projects are not relevant, two process variants could be the most efficient approach for the organization, despite requiring process higher maintenance efforts.

As mentioned in the introduction, there is no “on size fits all” solution for every given situation. All the discussed integration concepts have their eligibility, and all approaches have been implemented successfully within process improvement projects.

3 THE REQUIREMENTS AND TEST CHALLENGE

If one wants to expand the agile development principles to large system projects, one has to look especially at the requirement engineering and test activities. These two parts of the development cycle are crucial in understanding agile system development, since they connect the system level with the component development within the subprojects.

3.1 The Requirements Dilemma

When looking at the requirements breakdown of large system projects, one is faced inevitably with a dilemma.

In traditional development, the system requirements are specified as comprehensive as possible at the beginning of the project. The system requirements, the derived system architecture, and the breakdown of the system requirements into the requirements for the different software and hardware components are specified during the early project phases.

If we want to expand the agile development approach from the component to the system level,

the expectation would be, that the detailing, i.e., the breakdown of the system requirements into component requirements, takes place in iterations. On the other hand, a comprehensive overview over the expected functionality of a component is required to set up a suitable and stable component architecture.

This raises the following question: how much information needs a component team upfront to enable efficient agile development, and how much information can the system level provide without giving up the agile principles of iterative refinement? An appropriate balance needs to be achieved between these two contradicting expectations.

3.2 Agile System Testing

The "back end" of the development cycle is easier to grasp. In principle, it is straight forward to expand the agile approach to system testing. System testing has to be done iteratively within the iteration cycles and the definition of done should be based on passed system tests.

In practice, it is however often not possible to integrate system testing completely within short iteration cycles. Reasons for this can be, e.g., complex systems, long test durations, manual tests, or that some components of the project follow the traditional development approach.

Therefore, one also has to find the right balance between what's desirable and what's possible when expanding agile principles to system testing.

3.3 Proposed Solution

A possible approach to handle the challenges faced when expanding agile development to large system projects, is the introduction of an intermediate level in-between the system project and the component development.

The intermediate level breaks the project into several development steps. The number and duration of these steps is driven by how much information a component team needs upfront, so that it creates only a minimum of waste when continuing with the development in the next step. The duration of the steps can vary within the project. They should be as short as possible, but are driven by the structure of the development object, the boundary conditions and the organizational constraints.

By this means, one can keep the short iterations and agile principles on component level without compromising them. And it is possible to integrate the agile development of e.g. software components with traditional development of e.g. hardware components.

On the overall project, the breakdown of the system requirements and the system integration and system testing will be done in larger steps. This then summarizes the outcome of several development iterations on component level. This means, the system level has been "agilized" as much as sensible.

As pointed out above, the right balance between what's desirable and what's possible and sensible has to be found. This differs from organization to organization. Typically, the "agilization grade" of the system level increases over time.

4 THE ROLE CHALLENGE

Agile development approaches focus on specific responsibilities. Scrum defines, e.g., only three roles (product owner, scrum master, development team). On the contrary, traditional processes within development departments have typically between 40 to 60 roles.

Do these roles become obsolete with the introduction of agile development? Can every activity be mapped to the three Scrum roles?

4.1 Role Concept

The answer is of course no. Especially large system projects require additional roles. Depending on the organizational set up, the project scope and the boundary conditions imposed by the multiple stakeholders, a suitable role concept has to be defined. This must address the needs of the organization and large system projects, without compromising agile principles more than necessary.

4.1.1 Project Organization

Agile development teams work best with seven plus or minus two team members. This means, that large system projects require an additional project organization on top of the agile development teams, including coordinating functions.

These coordinating functions can be, e.g., an overall project manager, a chief product owner, or a chief architect. Nevertheless, the self organizing aspects of the agile development teams should be preserved as much as possible. I.e., the responsibilities of the traditional roles have to be adjusted, enabling the agile roles to be implemented

as consistent as possible.

4.1.2 Supporting Roles

Other roles that are typically required in large system projects are an overall project quality manager, a business administrator, and a supplier manger. These roles can be taken from the traditional development process. In most cases they do not have to be changed, since their tasks will remain very similar to traditional projects.

The key success factor for combining agile with traditional roles into an overall role concept is the adjustment of the responsibilities of the traditional roles. In order to preserve the agile principles as much as possible, some of the traditional roles have to be abandoned, others changed significantly.

4.2 Organizational Aspects

Another aspect that needs to be considered when introducing agile development is the organizational set up. Especially the relation between product management and development has to be adjusted.

This is best understood when one looks at the product owner role defined in Scrum. This role combines aspects from the traditional roles of product manager and project manager. Since most organizations separate these two functions into separate functional units, this requires special attention.

The desirable option would be the integration of the product management and development departments. But this might not always be possible or sensible. Other organizational constraints might favour a separate structure.

In this case, one has to define how the relation between these two departments is handled in agile development projects. Typical questions that need to be addressed are: Who takes over the product owner role? How do the responsibilities change? How can R&D provide operational support for the product owner?

These topics need to be considered very carefully, since they are a crucial aspect for the success of agile development in large projects and complex organizations.

5 CONCLUSIONS

Introducing agile development in large projects or complex organizations is possible. However, there is no defined blueprint how this should be done.

Agile development was originally intended to suit small software development teams. Any extension to large system projects is a balance between following the agile principles as much as possible, yet satisfying the organizational constraints and requirements from the multiple stakeholders.

This is especially true for organizations, where projects are a mix of agile, traditional and hybrid development. In this case, the process has to address the needs of all these projects. Depending on the objectives and boundary conditions of the affected organization, the optimal integration approach varies.

The key challenges, that need to be addressed in this context, are the definition of a suitable process framework, the structuring of the requirements breakdown and system testing, and the definition of a suitable role concept.

REFERENCES

Beck K. et. al., 2001. Agile Manifesto. agilemanifesto.org.

Beck K., 1999. Extreme Programming Explained: Embrace Change. *Addison-Wesley Professional*. Boston, 1st edition.

Schwaber K., Beedle M., 2001, Agile Software Development with Scrum, *Prentice Hall*. NJ, 1st edition.

Larman C., Vodde B., 2008. Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum. *Addison-Wesley Professional*. Boston, 1st edition.

Larman C., Vodde B., 2010. Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum. *Addison-Wesley Professional*. Boston, 1st edition.

Canditt S. et. al., 2010. Das V-Modell XT mit Scrum inside. *OBJECTspektrum*. Troisdorf.