Multi-Dimensional Context-Aware Adaptation of Service Front-Ends

Project no. FP7 – ICT – 258030

Deliverable 3.4.1 Agile Methodology Description (R1)

Due date of deliverable: 31/08/2011

Actual submission to EC date: 30/08/2011

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination level		
PU	Public	Yes

Document Information	
Lead Contractor	SAP AG
Editor	SAP AG
Revision	V1.0 (29/08/2011)
Reviewer 1	UCL
Approved by	
Project Officer	Jorge Gasós

Contributors	
Partner	**Contributors**
SAP	**Joerg Rett, Safdar Ali, Anna Lewandowski, Boris Kneisel**
Université catholique de Louvain	**Jean Vanderdonckt, Vivian Genaro Motti**
W4	**Jean-Loup Comeliau, Nicolas Bodin**
ISTI-CNR	**Fabio Paternò, Carmen Santoro**

Changes			
Version	**Date**	**Author**	**Comments**
0.1	10/06/2011	SAP	Initial Draft
0.2	20/06/2011	Contributors	First Contributions
0.3	29/06/2011	SAP	First Consolidation
0.4	01/07/2011	UCL	Reviewers Comments
0.5	29/07/2011	Contributors	Final Contributions
0.6	05/08/2011	SAP	Final Consolidation
0.7	12/08/2011	UCL	Reviewers Comments
0.8	19/08/2011	SAP	Final version
1.0	29/08/2011	TID	Sent Version

Executive Summary

This deliverable presents the first release of the Serenoa agile methodology description, supporting the development of the authoring tool within task T4.5.

The state-of-the-art of agile methodologies like Scrum and important terms and definitions like (e.g. Product Owner, Scrum Master ...) are discussed. After this, each partner in the development team for the authoring tool (T4.5 team) will explain their former experiences with agile methodologies (prerequisites). Then, the agile use-case, i.e. the development of the authoring tool within task 4.5 will be described. This deliverable will describe which roles have been assigned to the members of the T4.5 team, which Sprint patterns have been applied and which kinds of tools have been used. Results are presented through the integration of the agile process on the example of processing some Backlog Items and a survey of the team members on the implementation of the agile process. The document ends with conclusions and some future works to be done.

Table of Contents

1 Introduction

1.1 Objectives

This task aims at defining an Agile methodology based on existing Agile frameworks (including widely established standards e.g. Scrum) that supports developing the SFE components like algorithms, runtime, and tools. The methodology will build upon the agile modes and pre-requisites of the partners, describe the agile tools used, and reflect the performance of the process. The transparency generated by working in and documenting the agile process will be used to steer the implementation of the agile cross-partner, cross-location software development process.

1.2 Audience

This document has a public dissemination level, so theoretically it is open to public consultation by the general public. However, a key audience is represented by Project reviewers and officer, as well as any researcher/scientist who could be interested in the topics addressed by SERENOA.

1.3 Related documents

- Deliverable 1.1.1 Requirement Analysis (R1), which is the first report of the Serenoa which was developed and delivered by applying agile methodology
- The future deliverable 4.5.1 Authoring Environment and Development Tools will be the first product yielding a software prototype developed by applying agile methodology.

1.4 Organization of this document

The “Abstract” Section provides some summary information about the main content of the document. Then, Section 1 (“Introduction”) gives some related information about this Deliverable (e.g. objectives, planned audience, ...). In section 2 the state-of-the-art of agile methodologies like Scrum are discussed and important terms and definitions like (e.g. Product Owner, Scrum Master ...) are discussed. After this, in section 3 each partner in the development team for the authoring tool (T4.5 team) will explain their former experiences with agile methodologies (prerequisites). Then, in section 4 the agile use-case, i.e. the development of the authoring tool within task 4.5 will be described. Section 5 will describe which roles have been assigned to the members of the T4.5 team, which Sprint patterns have been applied and which kinds of tools have been used. Then, in section 6 results are presented through the integration of the agile process on the example of processing some Backlog Items and a survey of the team members on the implementation of the agile process. The document ends with section 7 with conclusions and some future works to be done.

2 Agile Methodologies

This section presents some possible methodologies for agile development. First, a review of these methodologies was performed in order to identify the possibilities, and compare them. By defining specific criteria, we could then evaluate the strengths and weaknesses of each methodology. Finally, by analysing the data collected, we were able to justify our choice of the Scrum method.

2.1 V-Model

The V-Model consists of an improved version of the Waterfall model. It decomposes the system in a series of tasks that are then organized sequentially. The main drawback of this approach is the impossibility of performing Backtracking, which prevents the changes in the requirements.

This approach is organized in two parts, as Figure 1 illustrates: the left branch defines the tasks organized in a sequence. The right branch consists of test and evaluation of the task previously defined.

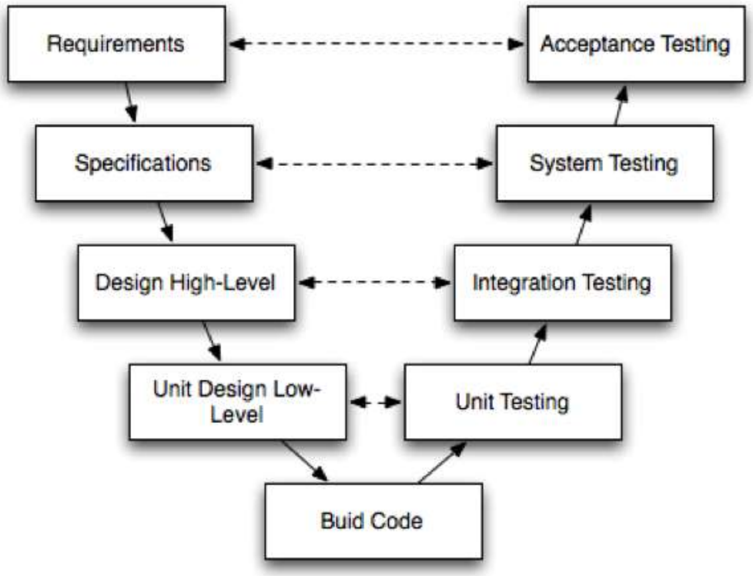


Figure 1: V-Model

The advantages of this model are: simplicity and organization. However, the main constraint is the impossibility of dynamic changes.

2.2 Rational Unified Process

The Rational Unified Process (RUP) is based on the Spiral model, and its main characteristics are that it is iterative and incremental. After each iteration the executable version of the project is tested and evaluated. The new versions are incremented to the previous ones.

The incremental phases are divided in 4 consecutive parts: Inception, Elaboration, Construction and Transition, as Figure 2 illustrates.

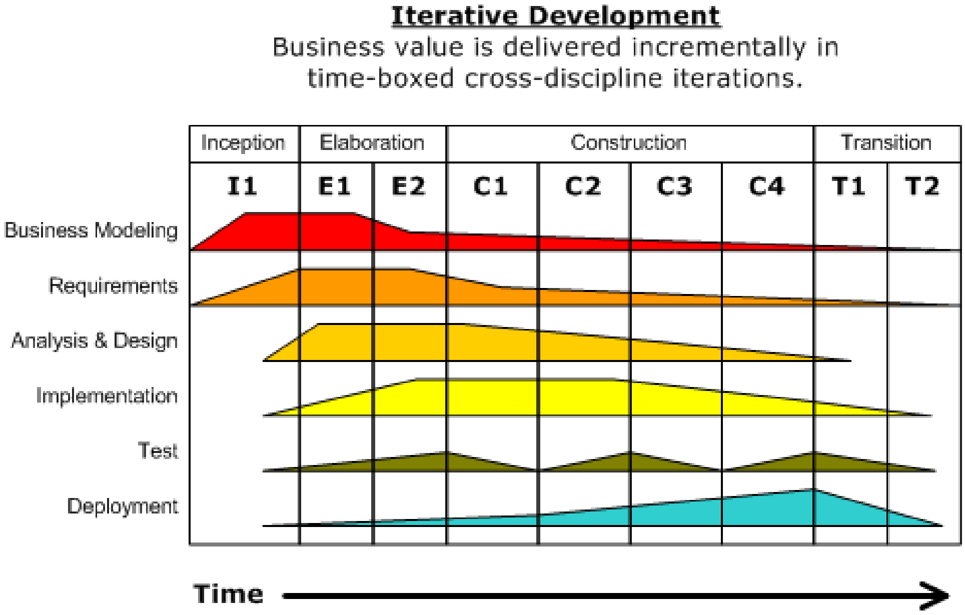


Figure 2: Phases and roles in Rational Unified Process (RUP)

- During the *Inception* there is the definition of the scope of the project, as well as costs and budgets, definition of constraints and features, considering the user needs and creating use case models in UML
- The *Elaboration* defines the architectural approach, plans, functionalities and the possible risks
- During the *Construction* the project is implemented iteratively according to the requirements previously defined
- The last phase: *Transition*, consists in transforming the project into a product, allowing customers to test it and provide their feedback

This method organizes all the development cycle by defining roles and responsibilities for each stakeholder. It is indicated for medium and large projects in which a large number of persons are involved.

The six best practices in the RUP process framework are:

- The iterative development: usually the sprints last 30 days (but there is some flexibility)
- The collaboration: the requirements are managed by the entire team and controlled by the Product Owner (see section 2.3.1)
- The architecture-centric approach: considers and selects carefully the architecture
- The visual modelling: usually UML is a common practice
- The progressive quality: iterative and incremental development ensure quality, once after each sprint the functionality is tested, measured and demonstrated (quality control is observed from different perspectives)
- The management of changes: changes are presented in the end of a sprint as part of the set of goals

2.3 Scrum

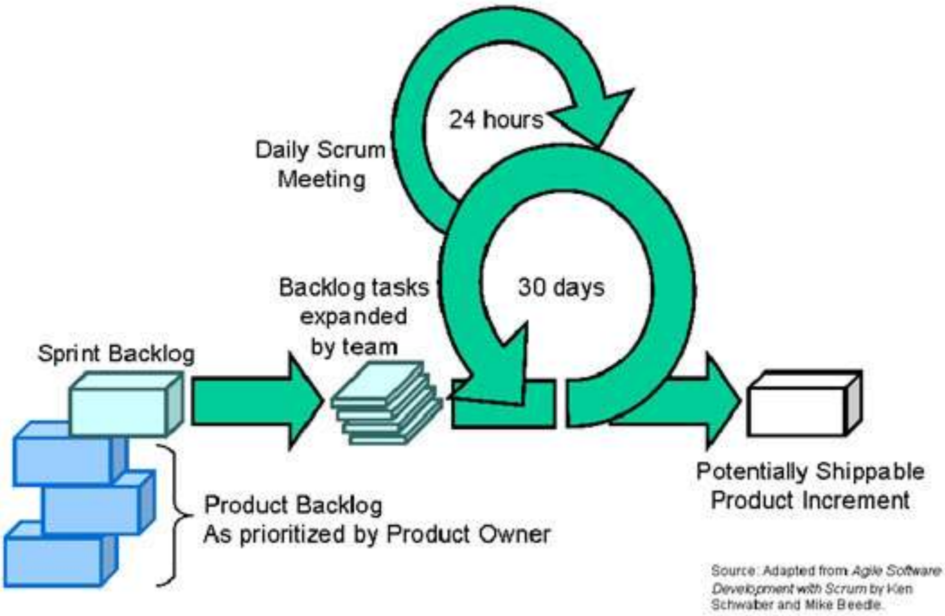
Scrum is centred on the persons; the customer must have all the needs satisfied. Its most important characteristic is the fact that changes in the requirements are considered in a dynamic way. Thus the customer is allowed to test and provide feedback for the project as soon as possible.

This method is also iterative and incremental, and it has short development cycles. Scrum adopts four principles:

- It prioritizes the persons and their interaction
- It prioritizes the functionality

- It considers collaboration
- It is flexible and adaptable to changes

This method is very dynamic and allows quick reactions for the changes in the requirements. In such a way, it is recommended for highly dynamic environments characterized by fast changing requirements where development teams are required to build complex products.


24 hours
Daily Scrum Meeting
Backlog tasks expanded by team
30 days
Sprint Backlog
Product Backlog As prioritized by Product Owner
Potentially Shippable Product Increment

Source: Adapted from *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.

Figure 3: Scrum adapted from Agile Software Development with Scrum by Ken Schwaber and Mike Breede

In practice as single-source-of-truth for referencing the Scrum Guide (Schwaber 2011) became well-cited and highly-acknowledged[1].

The following section will present definitions for entities of Scrum taken from the Scrum Guide (Schwaber 2011). Scrum consists of Scrum Teams and their associated roles, events, artifacts, and rules. First the Scrum Team composed of the roles, Product Owner, Development Team and Scrum Master will be described. Then Scrum Events like the Sprint, the Sprint Planning Meeting, the Daily Scrum, the Sprint Review Meeting and the Sprint Retrospective are presented. As Scrum Artifacts the Product Backlog and the Sprint Burndown Chart are described. Finally the “Definition of Done” and the importance of the Done criteria is presented.

2.3.1 The Product Owner

The Product Owner is responsible for maximizing the value of the product and the work of the Development Team. How this is done may vary widely across organizations, Scrum Teams, and individuals. The Product Owner is the sole person responsible for managing the Product Backlog. Product Backlog management includes:

- Clearly expressing Product Backlog Items;
- Ordering the items in the Product Backlog to best achieve goals and missions;
- Ensuring the value of the work the Development Team performs;
- Ensuring that the Product Backlog is visible, transparent, and clear to all, and shows what the Scrum Team will work on next; and,
- Ensuring the Development Team understands items in the Product Backlog to the level needed.

The Product Owner may do the above work, or have the Development Team do it. However, the Product Owner remains accountable.

[1] An additional collection of ScrumPapers edited by Jeff Sutherland and Ken Schwaber can be found at http://jeffsutherland.com/ScrumPapers.pdf

The Product Owner is one person, not a committee. The Product Owner may represent the desires of a committee in the Product Backlog, but those wanting to change a backlog item's priority must convince the Product Owner.

For the Product Owner to succeed, the entire organization must respect his or her decisions. The Product Owner's decisions are visible in the content and prioritization of the Product Backlog. No one is allowed to tell the Development Team to work from a different set of priorities, and the Development Team isn't allowed to act on what anyone else says (Schwaber 2011).

2.3.2 The Development Team

The Development Team consists of professionals who do the work of delivering a potentially releasable Increment of "Done" product at the end of each Sprint. Only members of the Development Team create the Increment.

Development Teams are structured and empowered by the organization to organize and manage their own work. The resulting synergy optimizes the Development Team's overall efficiency and effectiveness. Development Teams have the following characteristics:

- They are self-organizing. No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality;
- Development Teams are cross-functional, with all of the skills as a team necessary to create a product Increment;
- Scrum recognizes no titles for Development Team members other than Developer, regardless of the work being performed by the person; there are no exceptions to this rule;
- Individual Development Team members may have specialized skills and areas of focus, but accountability belongs to the Development Team as a whole; and,
- Development Teams do not contain sub-teams dedicated to particular domains like testing or business analysis.

2.3.3 The Scrum Master

The Scrum Master is responsible for ensuring Scrum is understood and enacted. Scrum Masters do this by ensuring that the Scrum Team adheres to Scrum theory, practices, and rules. The Scrum Master is a servant-leader for the Scrum Team. The Scrum Master helps those outside the Scrum Team understand which of their interactions with the Scrum Team are helpful and which aren't. The Scrum Master helps everyone change these interactions to maximize the value created by the Scrum Team (Schwaber 2011).

The Scrum Master serves the Product Owner in several ways, including:

- Finding techniques for effective Product Backlog management;
- Clearly communicating vision, goals, and Product Backlog items to the Development Team;
- Teaching the Development Team to create clear and concise Product Backlog items;
- Understanding long-term product planning in an empirical environment;
- Understanding and practicing agility; and,
- Facilitating Scrum events as requested or needed.

The Scrum Master serves the Development Team in several ways, including:

- Coaching the Development Team in self-organization and cross-functionality;
- Teaching and leading the Development Team to create high-value products;
- Removing impediments to the Development Team's progress;
- Facilitating Scrum events as requested or needed; and,
- Coaching the Development Team in organizational environments in which Scrum is not yet fully adopted and understood.

The Scrum Master serves the organization in several ways, including:

- Leading and coaching the organization in its Scrum adoption;

- Planning Scrum implementations within the organization;
- Helping employees and stakeholders understand and enact Scrum and empirical product development;
- Causing change that increases the productivity of the Scrum Team; and,
- Working with other Scrum Masters to increase the effectiveness of the application of Scrum in the organization.

2.3.4 The Sprint

The heart of Scrum is a Sprint, a time-box of one month or less during which a Done, useable, and potentially releasable product Increment is created. Sprints have consistent durations throughout a development effort. A new Sprint starts immediately after the conclusion of the previous Sprint.

Sprints contain and consist of the Sprint Planning Meeting, Daily Scrums, the development work, the Sprint Review Meeting, and the Sprint Retrospective.

During the Sprint:

- No changes are made that would affect the Sprint Goal;
- Development Team composition and quality goals remain constant; and,
- Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

Each Sprint may be considered a project with no more than a one-month horizon. Like projects, Sprints are used to accomplish something. Each Sprint has a definition of what is to be built, a design and flexible plan that will guide building it, the work, and the resultant product.

Sprints are limited to one calendar month. When a Sprint's horizon is too long the definition of what is being built may change, complexity may rise, and risk may increase. Sprints enable predictability by ensuring inspection and adaptation of progress toward a goal at least every calendar month. Sprints also limit risk to one calendar month of cost (Schwaber 2011).

2.3.5 Sprint Planning Meeting

The work to be performed in the Sprint is planned at the Sprint Planning Meeting. This plan is created by the collaborative work of the entire Scrum Team.

The Sprint Planning Meeting is time-boxed to eight hours for a one-month Sprint. For shorter Sprints, the event is proportionately shorter. For example, two-week Sprints have four-hour Sprint Planning Meetings.

The Sprint Planning Meeting consists of two parts, each one being a time-box of one half of the Sprint Planning Meeting duration. The two parts of the Sprint Planning Meeting answer the following questions, respectively (Schwaber 2011):

- What will be delivered in the Increment resulting from the upcoming Sprint?
- How will the work needed to deliver the Increment be achieved?

2.3.6 Daily Scrum

The Daily Scrum meeting is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours. This is done by inspecting the work since the last Daily Scrum and forecasting the work that could be done before the next one.

The Daily Scrum is held at the same time and place each day to reduce complexity. During the meeting, each Development Team member explains:

- What has been accomplished since the last meeting?
- What will be done before the next meeting?
- What obstacles are in the way?

The Development Team uses the Daily Scrum to assess progress toward the Sprint Goal and to assess how progress is trending toward completing the work in the Sprint Backlog. The Daily Scrum optimizes the probability that the Development Team will meet the Sprint Goal. The Development Team often meets immediately after the Daily Scrum to re-plan the rest of the Sprint's work. Every day, the Development Team should be able to explain to the Product Owner and Scrum Master how it intends to work together as a self-organizing team to accomplish the goal and create the anticipated increment in the remainder of the Sprint.

The Scrum Master ensures that the Development Team has the meeting, but the Development Team is responsible for conducting the Daily Scrum. The Scrum Master teaches the Development Team to keep the Daily Scrum within the 15-minute time-box.

The Scrum Master enforces the rule that only Development Team members participate in the Daily Scrum. The Daily Scrum is not a status meeting, and is for the people transforming the Product Backlog items into an Increment.

Daily Scrums improve communications, eliminate other meetings, identify and remove impediments to development, highlight and promote quick decision-making, and improve the Development Team's level of project knowledge. This is a key inspect and adapt meeting (Schwaber 2011).

2.3.7 Sprint Review Meeting

A Sprint Review Meeting is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog if needed. During the Sprint Review, the Scrum Team and stakeholders collaborate about what was done in the Sprint. Based on that and any changes to the Product Backlog during the Sprint, attendees collaborate on the next things that could be done. This is an informal meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration.

This is a four-hour time-boxed meeting for one-month Sprints. Proportionately less time is allocated for shorter Sprints. For example, two week Sprints have two-hour Sprint Reviews.

The Sprint Review includes the following elements:

- The Product Owner identifies what has been "Done" and what has not been "Done";
- The Development Team discusses what went well during the Sprint, what problems it ran into, and how those problems were solved;
- The Development Team demonstrates the work that it has "Done" and answers questions about the Increment;
- The Product Owner discusses the Product Backlog as it stands. He or she projects likely completion dates based on progress to date; and,
- The entire group collaborates on what to do next, so that the Sprint Review provides valuable input to subsequent Sprint Planning Meetings.

The result of the Sprint Review is a revised Product Backlog that defines the probable Product Backlog items for the next Sprint. The Product Backlog may also be adjusted overall to meet new opportunities (Schwaber 2011).

2.3.8 Sprint Retrospective

The Sprint Retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint.

The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint Planning Meeting. This is a three-hour time-boxed meeting for one-month Sprints. Proportionately less time is allocated for shorter Sprints.

The purpose of the Sprint Retrospective is to:

- Inspect how the last Sprint went with regards to people, relationships, process, and tools;
- Identify and order the major items that went well and potential improvements; and,
- Create a plan for implementing improvements to the way the Scrum Team does its work.

The Scrum Master encourages the Scrum Team to improve, within the Scrum process framework, its development process and practices to make it more effective and enjoyable for the next Sprint. During each

Sprint Retrospective, the Scrum Team plans ways to increase product quality by adapting the Definition of “Done” as appropriate.

By the end of the Sprint Retrospective, the Scrum Team should have identified improvements that it will implement in the next Sprint. Implementing these improvements in the next Sprint is the adaptation to the inspection of the Scrum Team itself. Although improvements may be implemented at any time, the Sprint Retrospective provides a dedicated event focused on inspection and adaptation (Schwaber 2011).

2.3.9 Product Backlog

The Product Backlog is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product. The Product Owner is responsible for the Product Backlog, including its content, availability, and ordering.

A Product Backlog is never complete. The earliest development of it only lays out the initially known and best-understood requirements. The Product Backlog evolves as the product and the environment in which it will be used evolves. The Product Backlog is dynamic; it constantly changes to identify what the product needs to be appropriate, competitive, and useful. As long as a product exists, a Product Backlog also exists (Schwaber 2011)

The Product Backlog lists all features, functions, requirements, enhancements, and fixes that constitute the changes to be made to the product in future releases. Product Backlog items have the attributes of a description, order, and estimate.

The Product Backlog is often ordered by value, risk, priority, and necessity. Top-ordered Product Backlog items drive immediate development activities. The higher the order, the more a Product Backlog item has been considered, and the more consensus exists regarding it and its value.

Higher ordered Product Backlog items are clearer and more detailed than lower ordered ones.

Requirements never stop changing, so a Product Backlog is a living artefact (Schwaber 2011).

2.3.10 Product Burndown Chart

At any point in time, the total work remaining to reach a goal can be summed. The Product Owner tracks this total work remaining at least for every Sprint Review. The Product Owner compares this amount with work remaining at previous Sprint Reviews to assess progress toward completing projected work by the desired time for the goal. This information is made transparent to all stakeholders. Various trend burndown, burnup and other projective practices have been used to forecast progress. These have proven useful (Schwaber 2011).

2.3.11 Sprint Backlog

The Sprint Backlog is the set of Product Backlog items selected for the Sprint plus a plan for delivering the product Increment and realizing the Sprint Goal. The Sprint Backlog is a forecast by the Development Team about what functionality will be in the next Increment and the work needed to deliver that functionality.

The Sprint Backlog defines the work the Development Team will perform to turn Product Backlog items into a “Done” Increment. The Sprint Backlog makes visible all of the work that the Development Team identifies as necessary to meet the Sprint Goal.

The Sprint Backlog is a plan with enough detail that changes in progress can be understood in the Daily Scrum. The Development Team modifies Sprint Backlog throughout the Sprint, and the Sprint Backlog emerges during the Sprint. This emergence occurs as the Development Team works through the plan and learns more about the work needed to achieve the Sprint Goal.

As new work is required, the Development Team adds it to the Sprint Backlog. As work is performed or completed, the estimated remaining work is updated. When elements of the plan are deemed unnecessary, they are removed. Only the Development Team can change its Sprint Backlog during a Sprint. The Sprint Backlog is a highly visible, real-time picture of the work that the Development Team plans to accomplish during the Sprint, and it belongs solely to the Development Team (Schwaber 2011).

2.3.12 Sprint Burndown Chart

At any point in time in a Sprint, the total work remaining in the Sprint Backlog items can be summed. The Development Team tracks this total work remaining at least for every Daily Scrum. The Development Team tracks these sums daily and projects the likelihood of achieving the Sprint Goal. By tracking the remaining work throughout the Sprint, the Development Team can manage its progress (Schwaber 2011).

2.3.13 The Done criteria

When the Product Backlog Item is described as Done, everyone must understand what Done means. Although this varies significantly per Scrum Team, members must have a shared understanding of what it means for work to be complete, to ensure transparency. This is the “Definition of Done” for the Scrum Team and is used to assess when work is complete on the product Increment.

The same definition guides the Development Team in knowing how many Product Backlog items it can select during a Sprint Planning Meeting. The purpose of each Sprint is to deliver Increments of potentially shippable functionality that adhere to the Scrum Team’s current Definition of “Done.”

Development Teams deliver an Increment of product functionality every Sprint. This Increment is useable, so a Product Owner may choose to immediately release it. Each Increment is additive to all prior Increments and thoroughly tested, ensuring that all Increments work together.

As Scrum Teams mature, it is expected that their Definition of “Done” will expand to include more stringent criteria for higher quality (Schwaber 2011).

3 Agile experience within Serenoa

This chapter describes the levels of experience with agile methodologies that already exist within the consortium. We have chosen the development of the authoring tool defined by task 4.5 as our agile use case. Currently the three partners ISTI, W4 and SAP are working on the development of this tool. The partner ISTI had not used agile methodologies so far. Thus, the following sections will describe the experiences of W4 and SAP.

3.1 Agile background of the SAP team

The SAP team started to use agile methods and techniques that consider the Scrum-framework as early as the fourth quarter of 2010. The team decided to stick to “plain vanilla” Scrum[2]. A full-day training was provided to all Scrum Team-members and extra training covering special topics of the role of the Product Owner was provided to the Product Owner on the fly, while setting-up the Backlog. Apart from the Scrum Master none of the team-members had run a project in Scrum-mode before. Concerning project-initiation, after the training-session, a meeting was held to decide the Sprint planning for Sprint01.

3.1.1 Roles

The role of the Scrum Master was assigned to an experienced and externally certified Scrum Master (CSM) from a different group of the organization. This ensured an unbiased view-point due to a different reporting-line as the other team members. In order to keep overhead low, the Product Owner role was assigned to the Project Leader who had been nominated to the EC and acted as an official SAP-contact for consortium. Besides their roles as Scrum Master and Product Owner, both persons also acted with a certain capacity as members of the Development Team (see Table 1). The Development Team consisted of four additional persons with different degrees of capacity and domain expertise, like development, UI design, user experience, communication, business development and analysis. As the total number of Scrum Team members (6) exceeded the average number of full time employees requested for funding (2.5) of the research project, this can be seen as a SAP-Invest. As it can be seen in Table 1, the members of the team were diverse in terms of domain expertise and level of involvement (capacity) in the project.

Table 1: Role, domain expertise and involvement of the SAP team

Role	Domain Expertise	Capacity[3]
ScrumMaster	User Experience	25% + 10%
Product Owner	Development	50% + 50%
Development Team	Development	100%
Development Team	User Interface Designer	100%
Development Team	Communication and Business development	20%
Development Team	Business analysis	20%

All team-members were collocated in one building, sharing an office with the main EC-funded persons. This office was declared the Sereona Team Space, where all Scrum-related meetings (Planning Meeting, Review/Demo Meeting, Retrospective, Daily Scrum) were held.

[2] “Plain Vanilla” Scrum (Schwaber 2004, Schwaber 2007) prescribes a two-part Sprint Planning Meeting. During the first part of the meeting, the Product Owner is supposed to meet with the team to explain sprint backlog priorities, as well as to clarify any functional issues. The team is then supposed to retreat (without the Product Owner) to estimate tasks, as well as to “sign-up” for work. By the end of the second step of the planning session, all work is supposed to be estimated and assigned.

[3] Relative to a full time employee

3.1.2 Sprint Patterns

Aiming to maximize the speed-of-response-to-expected-changes, in the starting phase of the project, the team decided for a sprint-duration of one week. This decision was due to the foreseeable uncertainty during the first phase of the project where set-ups and processes need to be defined and clarified.

In order to run an efficient pattern the team decided to block the Mondays as the meeting days. Additional Scrum meeting patterns consisted in:

- Monday (M), 09:30-10:30h Review/Demo Meeting (sprint N)
- Monday (M), 10:30-11:30h Retrospective (sprint N)
- Monday (M), 15:30-16:30h Planning Meeting (sprint N+1)
- Every work-day except Monday 14:00-14:15h, Daily Scrum

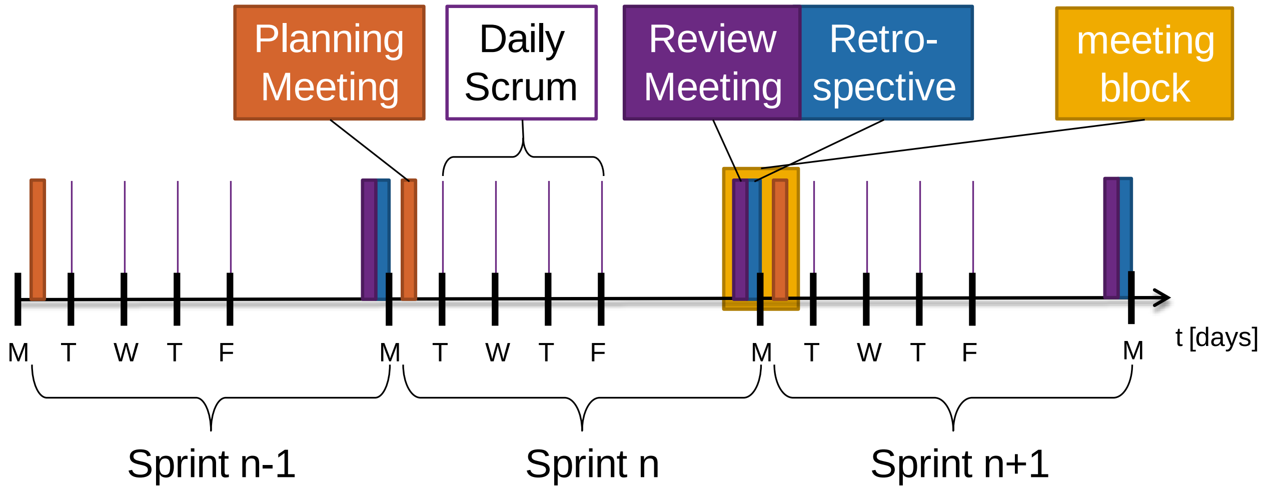


Figure 4: Sprint patterns of the SAP team showing Sprints, Planning and Review Meetings and Retrospective which are weekly and daily scrum meetings.

During the Daily Scrum, the team maintains the Sprint Burndown Chart (half-page flip-chart paper hung-up in the Team Space) – to collect completed tasks (small PostIts) into a special section of the task-board (full-page flip-chart in landscape fashion hung-up in the team-space) and burn them in a Daily Scrum only, if the last task of a Backlog Item is Done (Backlog Items are represented by larger PostIts). By this “Backlog-complete on burn-down” paradigm the team ensures, that the sprint-burn-down is also valid for the Product Owner, since it actually reports real value-progress and not just “time-consumed” (convertible into “money” using people’s hourly-rates & tariffs).

In the Review Meetings at the end of each Sprint each team member shares recent results they have achieved in the last sprint to track progress. The accountable Product Owner signs-off respective Backlog Items according to pre-defined acceptance-criteria (developed with one key-player / domain-expert of the team per Backlog Item in Sprint-preparation-sessions and shared with the whole team in Sprint-planning-sessions at the beginning of a respective sprint).

In the Retrospective, the SAP team gives a feedback on what went good and what went wrong. Each team member states his feedback by answering the following five questions.

1. What were valuable take-aways, what was a positive personal learning?
2. What were things that we fell short on?
3. What was the core of the Sprint, the dominating issue?
4. What were the issues to point-out, the personal observations?
5. What was cool-thumbs-up, what were positive things to keep?

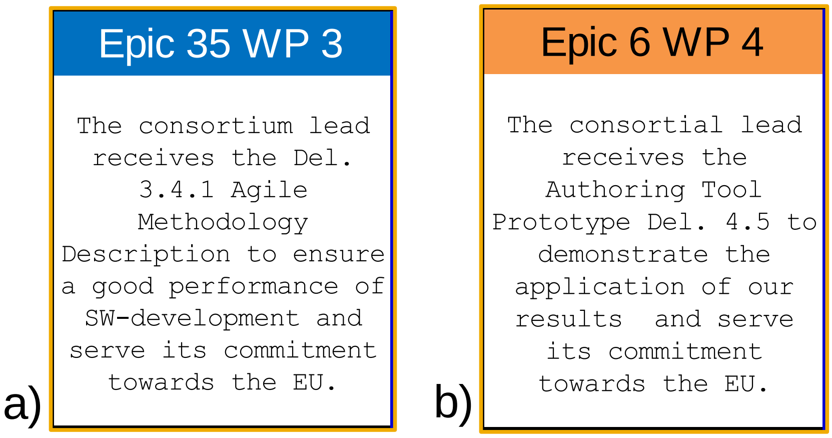
During the Planning Meeting, the Product Owner presents the Backlog Items to the team. After the Product Owner role has left the room the team starts to estimate the effort (the time) that is needed to complete each Backlog Item. The team might choose to estimate some of the Backlog Items collaboratively. The estimation is then conducted using the so called agile Planning Poker is played, this has been mentioned as the best approach to agile teams for estimation (Grenning 2002). With this approach the team members present their estimation simultaneously, by drawing poker cards. After a short sequence of statements and re-estimations the team converges to a certain value. The SAP team experienced, that in all cases where Planning Poker was played, the final-team estimation was better than the initial single team member estimation.

In order to stay within the one hour boundaries of the Sprint Planning Meeting the Product Owner usually meets with the team members beforehand. These pre-alignment sessions (“(“Sprint-preparation meetings”) were used to discuss the upcoming Backlog Items, settle first estimations for the effort and agree on the acceptance (Done) criteria. Through these meetings the Product Owner can estimate quite well how many Backlog Items fit into the upcoming Sprint.

3.1.3 Tools

Backlog Items were crafted by the Product Owner according to the work package description in the Description of Work (DoW) for the project. The final goal is very often to ship a deliverable as shown in Figure 1Figure 5 a) on the example of deliverable D3.4.1 “Agile Methodology Description (R1)”. Another goal is the completion of tasks like T4.5 “Authoring environment and development tools” shown in Figure 5 b). Both, deliverables and tasks, are usually shaped for an effort of six months or more. Thus, they are too big to serve as a Backlog Item to be completed in a single Sprint. To overcome this problem the Product Owner has chosen the format of Epics described in the format of user stories as shown in Figure 5.

User stories are a lightweight technique for expressing software requirements (Cohn 2004). A User Story is a brief description of functionality as viewed by a user or by a customer of the system. User Stories are free-form, and there is no mandatory syntax. However, it can be useful to think of a story generally fitting this form: “As a <type of user>, I want <capability> so that <business value>.”(Cohn 2006). As an example the user story of the epic for writing this deliverable is shown in Figure 5.

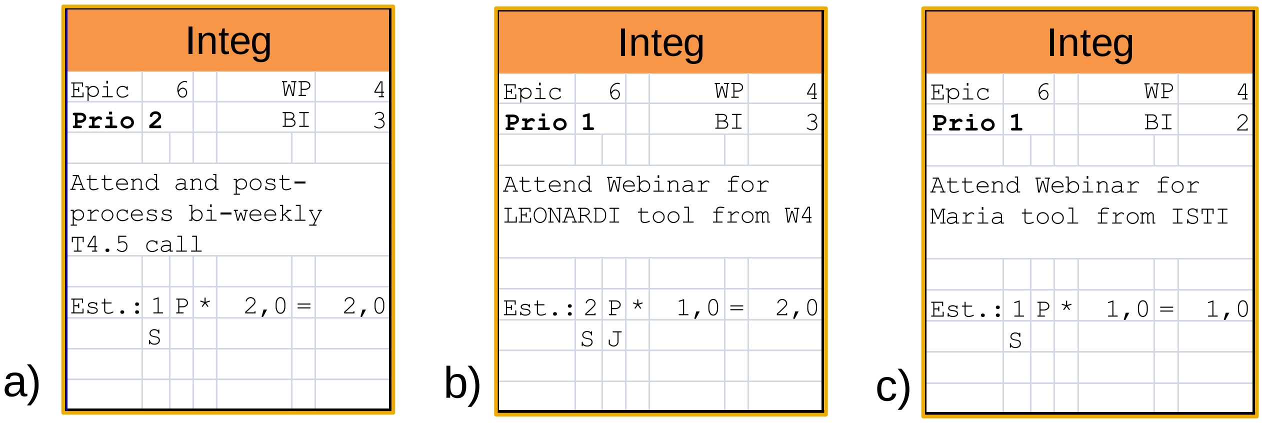


The consortium lead receives the Del. 3.4.1 Agile Methodology Description to ensure a good performance of SW-development and serve its commitment towards the EU.

The consortial lead receives the Authoring Tool Prototype Del. 4.5 to demonstrate the application of our results and serve its commitment towards the EU.

Figure 5: Description of Epics using the form of user stories. a) Epic concerning the writing of this deliverable 3.4.1 “Agile Methodologies as part of work package 3 b) Epic concerning the development of “Authoring tool” as part of work package 4.

The Epics can be seen as objects of a higher hierarchical level. Usually several Backlog Items over several Sprints are associated to on Epic. As an example the Backlog Items shown in Figure 6 are all associated to Epic 6. Figure 6 shows, that this association is reflected on the first white line of the Backlog Item as well as the relationship to the work package.



a)

Integ

Epic	6	WP	4
Prio	**2**	BI	3

Attend and post-process bi-weekly T4.5 call

Est.:	1	P	*	2,0	=	2,0
	S					

b)

Integ

Epic	6	WP	4
Prio	**1**	BI	3

Attend Webinar for LEONARDI tool from W4

Est.:	2	P	*	1,0	=	2,0
	S	J				

c)

Integ

Epic	6	WP	4
Prio	**1**	BI	2

Attend Webinar for Maria tool from ISTI

Est.:	1	P	*	1,0	=	1,0
	S					

Figure 6: Layout of the Backlog Items of the SAP team. The first line holds the index of the epic (6) and the related work package (4). The second line specifies the priority level (1) and the index of the Backlog Item (all BIs are taken from different Sprints, so Indices might be missing or doubled). The middle part describes the objective. The lines below represent the estimated effort in storypoints and the team member who has committed to work on this BI.

The second line reflects one of the important tasks of the Product Owner, i.e. to prioritize the Backlog Items. As a rule the team members need to start working on high priority Backlog Items first.

3.1.4 Interactions with entities outside the Scrum Team

The consortium lead of the Serenoa project is one major line of interaction for the SAP team. Request from the consortium trigger the creation of Backlog Items by the Product Owner. The consortium lead of Serenoa has in the agile settings the role of a Customer, who receives major releases of prototypes and reports them in form of deliverables. In this sense the project officers act as higher-level Costumers.

End-Users are another important line of interaction, as the SAP team leads deliverables related to requirement analysis. End-user engagement is managed using a special SAP customer engagement initiative within which regular sync-sessions are provided to customers to include them into info-loops regarding progress made based on conducted interviews with them, these interviews ultimately may converge in usability testing.

3.2 Agile background of the W4 team

For the last decade, W4 has been strongly involved with agile project methodologies. First inspired by XP's (eXtreme Programming) best practice rules for programming and later by Scrums' iterations, in which the customer takes a proactive role, W4's products are designed to support projects using agile methodologies.

There are four main ways in which W4 is actively involved with agile methodologies in its everyday operations:

1. W4 explicitly provides products for supporting agile methodologies
2. W4's internal R&D uses the Scrum methodology for developing products
3. W4's consultants are commonly embedded in customer development teams applying agile methodologies
4. W4 provides *Agilia*, an application for managing agile projects

3.2.1 Products by W4 which support agile methodologies

As a software editor, W4 distributes two main products, namely LEONARDI and BUSINESS FIRST. Both are software suites for designing and executing composite, collaborative business applications. W4 is usually involved in B2B communication where its products are used by development teams, that are in charge of delivering final applications to their own customers. W4's products are based (1) on model-driven concepts and (2) on BPM techniques.

One direct goal emphasized by W4 to its users is that its technology provides appropriate ways for supporting agile project methodologies. Specifically, this challenge is met by:

- Making the model the main focus of the development process

- Avoiding code generation

Unlike other actors in the model-driven engineering world who use the model-driven concepts to automate code generation from the model (thus following OMG's MDA -Model Driven Architecture- recommendations), W4's approach relies on an engine-based mechanism. With W4's products for example, the model is directly interpreted to generate the GUI on the fly: there is no code generation. This principle boosts agility because it shortens the development cycle: the only code that is needed is not redundant with the model, but specializes inherited behaviours.

Indeed, W4's field experience has shown that both XP's *timeboxing* cycles and Scrum's *sprints* are traditionally not easy to deal with in a MDA environment. The main reason is that in an ideal world, MDA aims at generating 100% of the application code automatically from the model. The model consists of PIMs (Platform Independent Models) that are automatically converted into PSMs (Platform Specific Models). The generated code is then deployed to run the business application. This process introduces complexity in agile iterations (sprints in Scrum) because it then becomes hard to keep both the model and the code in sync. When doing so, there is a strong risk of iterating on the code and not anymore on the model, which then contradicts model driven principles.

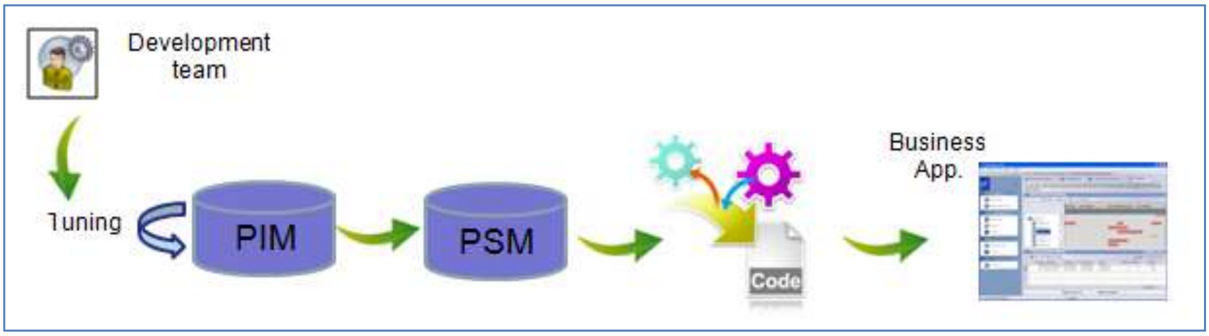


Figure 7: Development with traditional MDA makes iterations on code and model uneasy

Therefore, in order to follow the manifesto for agile software development's recommendation in a more efficient way (see http://agilemanifesto.org/), W4's view of model-driven engineering is closer to Scott W. Ambler's AMDD, in which «*models just barely good enough*» are used from the beginning of the project, and then improved, iteration after iteration. W4's conviction is that model-driven concepts serve agility in many ways when they are appropriately applied, leading the way to more adaptable applications (applications "*built for change*", as the Forrester analysts puts it), able to change quickly, either for technological or functional reasons.

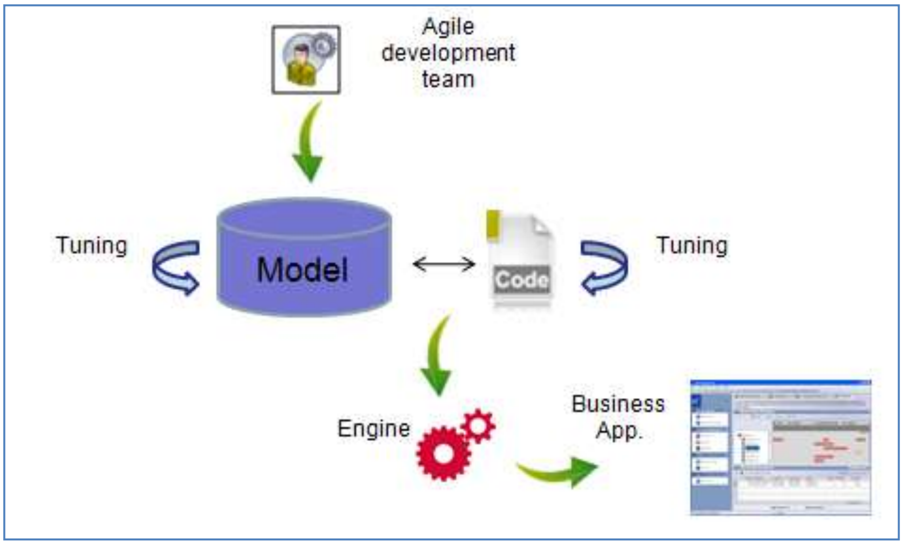


Figure 8: W4' MDE fits well agile development by avoiding redundancy between model and code

With this goal in mind, W4's technology is aimed at putting model driven engineering at the service of agility, mainly by:

- providing efficient communication tools that can be used by agile development teams, using the model as the main way of involving early the customer and get constructive feedback;
- reducing complexity by decoupling business aspects from technological ones: while the team and the customer focus on business activities (business model, rules and processes), the product is in charge of technical integration. This allows the team to spend more time on core business aspects and less on technical aspects;

3.2.2 Scrum methodology applied by W4's R&D team

W4's R&D team is composed of about 15 software engineers. Internally, product releases are consistently implemented following the Scrum methodology. W4 has found that Scrum is the best fit methodology to obtain the satisfying results, especially when short deadlines must be met.

In this R&D context, a scrum master orchestrates the team (usually the R&D manager) and ensures that its members are appropriately committed in the project (and not only involved, see below). He creates a favourable context for development.

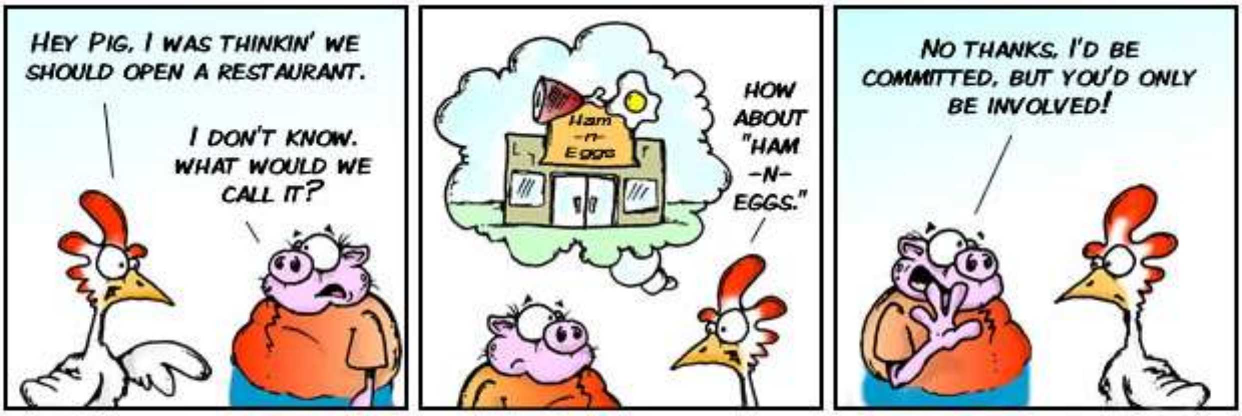

HEY PIG, I WAS THINKIN' WE SHOULD OPEN A RESTAURANT.
I DON'T KNOW. WHAT WOULD WE CALL IT?
Ham -n- Eggs
HOW ABOUT "HAM -N- EGGS."
NO THANKS, I'D BE COMMITTED, BUT YOU'D ONLY BE INVOLVED!

Figure 9: Ken Schwaber's story of pig and chicken explaining commitment in Scrum

The role of the project owner is often taken by the person in charge of quality assurance at W4, who accepts or rejects the work achieved during the different sprints. He is the one who defines the priorities based on the deadlines and the product roadmap. Stand-up meetings are held daily to improve communication and foster the team's involvement.

3.2.3 Participation in agile methodology teams for W4's costumer development projects

By offering its professional services, W4 also participates in various projects led by its different customers; these projects belong to multiple contexts. Consequently, W4 consultants are commonly included in agile development teams (usually applying the scrum methodology), in which they can hold any role needed in such methodologies, i.e., scrum master, Product Owner or team member.

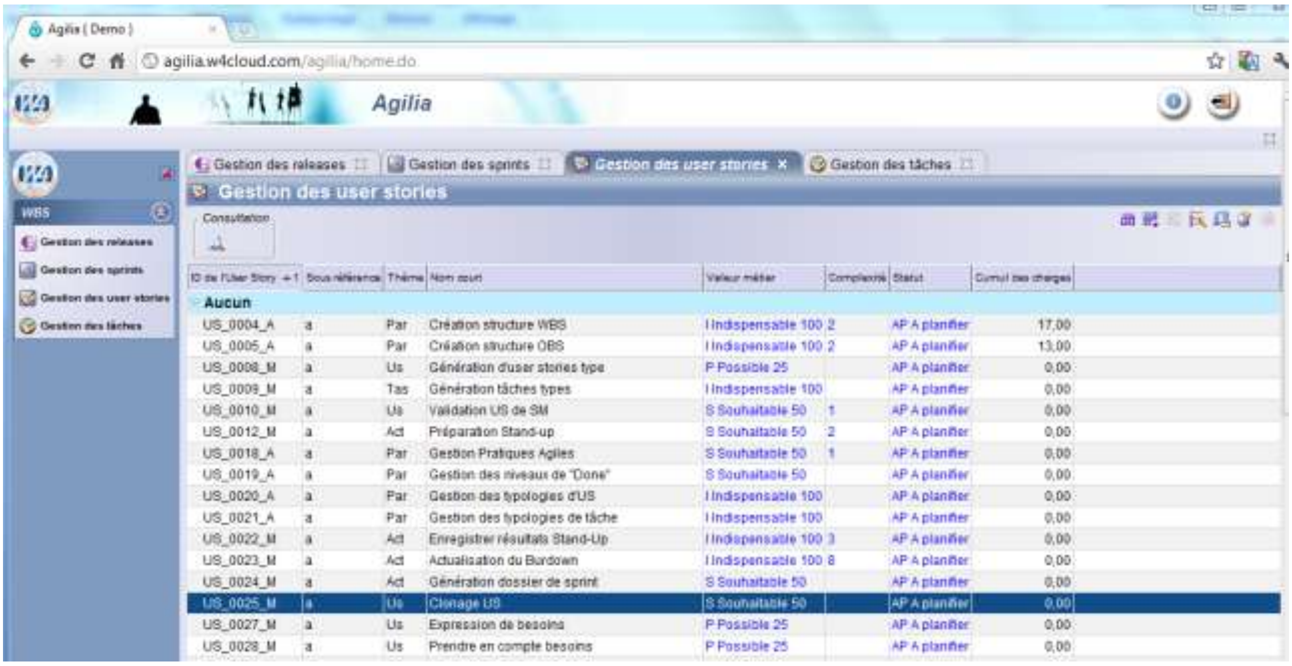
Involved with agile methodologies, the teams use model-driven concepts as a foundation; the teams are composed by professional service consultants (about 10 members) that provide W4 continuous feedback about specific project needs. This direct contact with the market is an excellent opportunity to keep improving W4' products and make them more fit to agile development. Therefore, consultant's reports are also used to feed W4's products roadmaps.

This field experience was also used by W4 to develop *Agilia*, an application dedicated to managing products according to agile methodologies.

3.2.4 Agilia, a free application by W4 for conducting agile projects

W4 offers, as a free add-on to its products, *Agilia*, an application for managing and reporting purposes in agile (scrum-like) projects. This application aims at helping project managers to adopt agile methodologies, to edit their project data as work unfolds, and to visualize key project indicators. It is an efficient way for them to take ownership of their application.

The application offers, for example, forms to edit data for user stories, support to prepare for action, support to release planning, sprint planning, sprint review, sprint retrospective and after daily stand-up meetings. It provides features for role management (team member, Product Owner and scrum master), task management, tasks update, data workload (estimated, effective, left) and graphical views of key project indicators, such as the number of user stories, the complexity curve, the business value curve or the calculation of the velocity.



Gestion des user stories

ID de l'User Story ↑1	Sous référence	Thème	Nom court	Valeur métier	Complexité	Statut	Cumul des charges
Aucun							
US_0004_A	a	Par	Création structure WBS	I Indispensable 100	2	AP A planifier	17,00
US_0005_A	a	Par	Création structure OBS	I Indispensable 100	2	AP A planifier	13,00
US_0008_M	a	Us	Génération d'user stories type	P Possible 25		AP A planifier	0,00
US_0009_M	a	Tas	Génération tâches types	I Indispensable 100		AP A planifier	0,00
US_0010_M	a	Us	Validation US de SM	S Souhaitable 50	1	AP A planifier	0,00
US_0012_M	a	Act	Préparation Stand-up	S Souhaitable 50	2	AP A planifier	0,00
US_0018_A	a	Par	Gestion Pratiques Agiles	S Souhaitable 50	1	AP A planifier	0,00
US_0019_A	a	Par	Gestion des niveaux de "Done"	S Souhaitable 50		AP A planifier	0,00
US_0020_A	a	Par	Gestion des typologies d'US	I Indispensable 100		AP A planifier	0,00
US_0021_A	a	Par	Gestion des typologies de tâche	I Indispensable 100		AP A planifier	0,00
US_0022_M	a	Act	Enregistrer résultats Stand-Up	I Indispensable 100	3	AP A planifier	0,00
US_0023_M	a	Act	Actualisation du Burdown	I Indispensable 100	8	AP A planifier	0,00
US_0024_M	a	Act	Génération dossier de sprint	S Souhaitable 50		AP A planifier	0,00
US_0025_M	a	Us	Clonage US	S Souhaitable 50		AP A planifier	0,00
US_0027_M	a	Us	Expression de besoins	P Possible 25		AP A planifier	0,00
US_0028_M	a	Us	Prendre en compte besoins	P Possible 25		AP A planifier	0,00

Figure 10: W4's *Agilia* application for managing agile projects (tab showing user stories)

4 The agile use case: Task 4.5 Authoring tool

This section gives a short overview on task T4.5 "Authoring environment and development tools" which was also chosen as the name-giver for the team and as a use-case for applying the agile methodology. The current state of the development is presented on the example of two quite recent developments. One is the strategy of following two paths of development: a web-based approach and a plug-in based approach. The second is the description of the design process for a first mock-up. A more thorough discussion on the prototype will follow in deliverable D4.5.1 "Authoring Environment (R1)".

The objective of task 4.5 is to develop an authoring environment and its accompanying analysis tools. The environment and its tools aims to facilitate and make it more efficient for designers and programmers to design, build and deploy adaptive SFEs using the languages provided by work package 3. The authoring tool will help the designers, engineers and web authors to easily create context-sensitive SFEs for different platforms, which may also use different interaction modalities, e.g. graphics, voice, touch etc. The authoring tools will provide support for editing not only the model-based descriptions at both abstract and concrete levels, but also the context-dependent transformations rules.

4.1 Types of Authoring Tools

Two types of authoring tools are planned to be developed under Task 4.5:

- An IDE plug-in that can be integrated in one of the most widely used IDEs, i.e. Eclipse in the research community for the software development;
- A web based application, which runs within a HTML5-compliant web browser and operates on models that are held on the web server side. Ideally, this version would allow live concurrent editing by multiple users, so that they could see and discuss the changes that are remotely executed, in real time. The target delivery platforms would be desktop, tablet and mobile devices that support HTML5.

There are certain advantages of using HTML5 palette/canvas based editor; for instance it inherently provides features, such as: layout, drag & drop, and the means to edit properties of a control, or to remove it from the canvas. HTML5 supports many new syntactical features, i.e. the <video>, <audio>, <header> and <canvas> elements, as well as the integration of SVG content. These features are designed to make it easy to include and handle multimedia and graphical content on the web without having to resort to proprietary plug-ins and APIs.

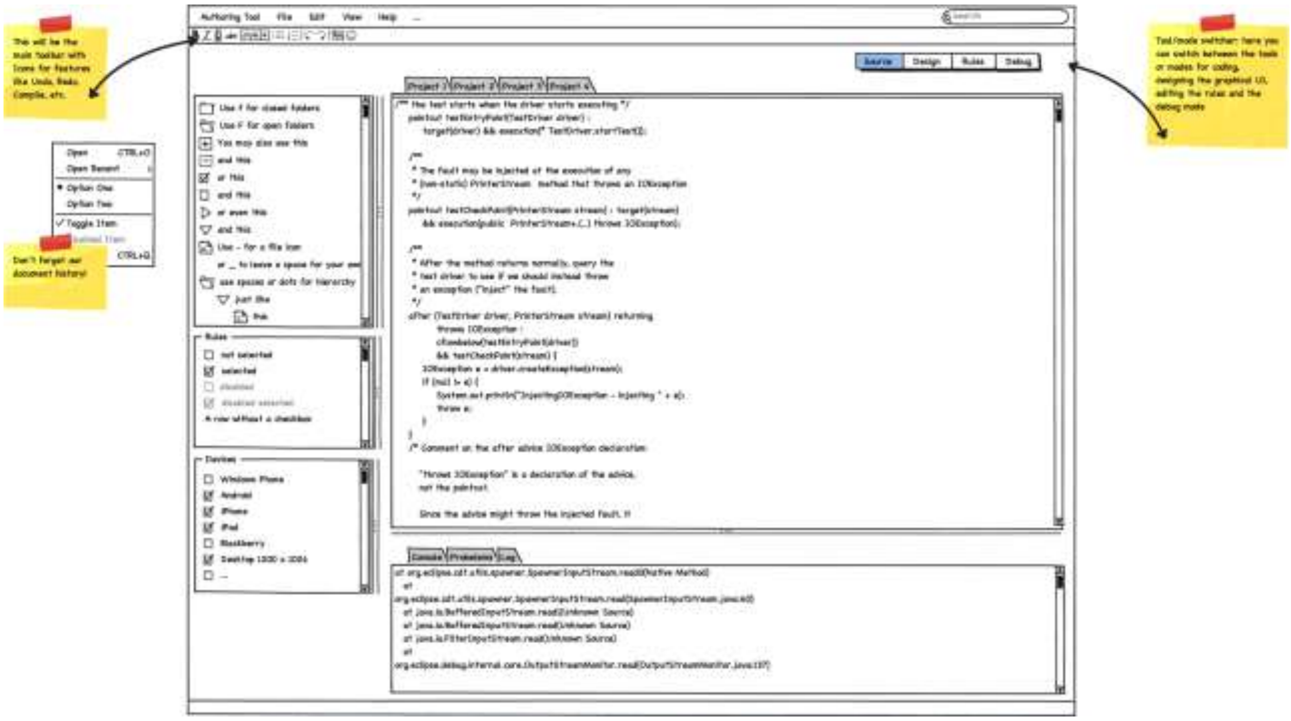
The JAVA classes/libraries from the already existing tools, i.e. MARIAE, LEONARDI will be used during the development of both types of authoring tools. In addition, online web tools will also be evaluated in order to gain experience of the already existing tools. The requirement specifications derived from Task 1.1 would then need to be revisited and the classification of functional and non-functional requirements would need to be verified.

4.2 Design of a Mock-Up

In the initial design phase, the architecture and the user interface needed to be defined. In this sense, Balsamiq was used as a tool for designing the first mock-ups of the plug-in based authoring tool. These mock-ups were presented to the users to get their quick feedback. At the same time these mock-ups will also be provided to the developers, who will start implementing the designs. Balsamiq was chosen because it is a an efficient tool with which we can rapidly design first ideas and user interfaces without going too much into detail with respect to colours, sizes and shapes. The mock-ups look like paper prototypes, which make it easier to get true user feedback. Often users are intimidated when polished and finished UIs are presented to them. Then they don´t have the heart to tell the truth because they think that they will 'destroy' the whole work of the designer or even developer. For the very first user feedback such paper prototypes are enough and don't need to be polished.

The following figures show the first designed mock-ups for the plug-in based SAP's authoring tool. Figure 11 shows a source code editor mock-up, which contains a menu bar and a tool bar, as well as several libraries, project and file trees, a console, log pane, error pane and a pane for editing the source. The yellow notes on the left and right side of the mock-up are notes from the UI designer to the developer. These notes can be questions or comments. Figure 12 shows a design editor mock-up. The composition is similar to the source code editor, but it includes an additional toolbar, which contains templates, styles, UI elements, etc.

After having defined the user interface (and gained the user feedback), the UI designer worked closely together with the developers and discussed the interfaces. Consistently, the results were discussed with the users, and eventually the developers started developing the first prototype of the authoring tool with the collaboration of designers and users.



This will be the main toolbar with icons for features like Undo, Redo, Compile, etc.

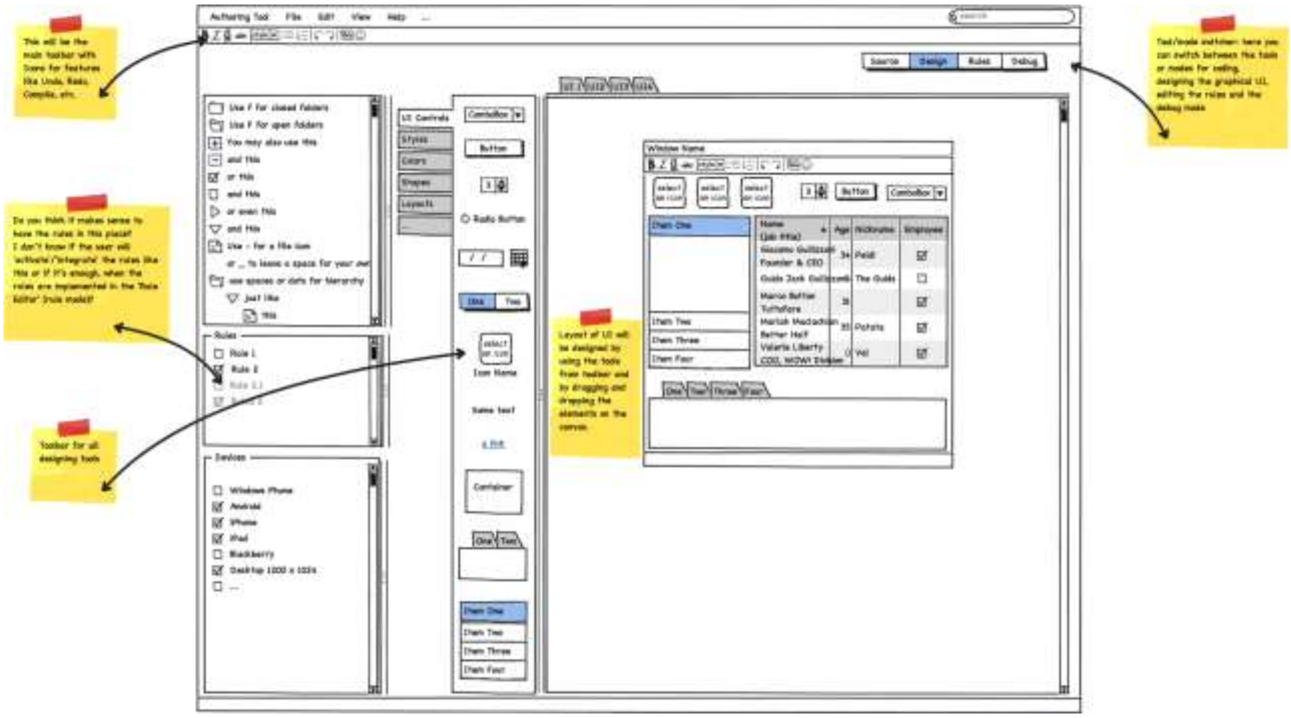
Don't forget our document history!

Tool/mode switcher: here you can switch between the tools or modes for coding, designing the graphical UI, editing the rules and the debug mode

```
/** The test starts when the driver starts executing */
  pointcut testEntryPoint(TestDriver driver) :
     target(driver) && execution(* TestDriver.startTest());

  /**
   * The fault may be injected at the execution of any
   * (non-static) PrinterStream method that throws an IOException
   */
  pointcut testCheckPoint(PrinterStream stream) : target(stream)
     && execution(public PrinterStream+.(..) throws IOException);

  /**
   * After the method returns normally, query the
   * test driver to see if we should instead throw
   * an exception ("inject" the fault).
   */
  after (TestDriver driver, PrinterStream stream) returning
        throws IOException :
        cflowbelow(testEntryPoint(driver))
        && testCheckPoint(stream) {
     IOException e = driver.createException(stream);
     if (null != e) {
        System.out.println("InjectingIOException - injecting " + e);
        throw e;
     }
  }
  /* Comment on the after advice IOException declaration:

     "throws IOException" is a declaration of the advice,
     not the pointcut.

     Since the advice might throw the injected fault, it
```

```
at org.eclipse.cdt.utils.spawner.SpawnerInputStream.read0(Native Method)
   at
org.eclipse.cdt.utils.spawner.SpawnerInputStream.read(SpawnerInputStream.java:60)
   at java.io.BufferedInputStream.read1(Unknown Source)
   at java.io.BufferedInputStream.read(Unknown Source)
   at java.io.FilterInputStream.read(Unknown Source)
   at
org.eclipse.debug.internal.core.OutputStreamMonitor.read(OutputStreamMonitor.java:137)
```

Figure 11: Mock-up for the Authoring Tool's Source Editor – Designed with Balsamiq



Name (job title)	Age	Nickname	Employee
Giacomo Guilizzoni Founder & CEO	34	Peldi	☑
Guido Jack Guilizzoni		The Guids	☐
Marco Botton Tuttofare	31		☑
Mariah Maclachlan Better Half	35	Patata	☑
Valerie Liberty CDO, WOW! Entity	0	Val	☑

This will be the main toolbar with icons for features like Undo, Redo, Compile, etc.

Do you think it makes sense to have the rules in this place? I don't know if the user will 'activate'/'integrate' the rules like this or if it's enough, when the rules are implemented in the 'Rule Editor' (rule mode)?

Toolbar for all designing tools

Layout of UI will be designed by using the tools from toolbar and by dragging and dropping the elements on the canvas

Tool/mode switcher: here you can switch between the tools or modes for coding, designing the graphical UI, editing the rules and the debug mode

Figure 12: Mock-up for the Authoring Tool´s Design Editor – Designed with Balsamiq

5 Agile processes and tools used for the development

This chapter describes which roles have been assigned to the members of the T4.5 team, which Sprint patterns have been applied and how they are related to the SAP team Sprint patterns and which kind of tools have been used.

5.1 Roles

In order to investigate and review the role of the Scrum Master in the T4.5 team it was decided to run the team for the first six months without and the following six months with a Scrum Master. At the time of finishing this deliverable the period without Scrum Master was just about to end.

The role of the Product Owner was assigned to a member from W4 as this person had experience in both agile methodologies and design time tools. Besides his role as a Product Owner, the person also acted with a certain capacity as team members (see Table 2). The team consisted of four additional persons with different degrees of capacity and domain expertise like development, UI design, user experience, communication, business development and analysis.

Table 2. Roles Distribution

Role	Partner	Domain Expertise
Product Owner	W4	Development
Team Member	W4	Development
Team Member	ISTI	Development
Team Member	SAP	User Interface Designer
Team Member	SAP	Development
Team Member	SAP	Development

5.2 Sprint Patterns

In order to fit the Sprint pattern of the T4.5 team with the already existing Sprint pattern of the SAP team, a Sprint duration of two weeks was chosen.

Wednesday was chosen as the meeting block day of the T4.5 team aiming not to collide with the existing one of the SAP team. Sprint and meeting block patterns are:

- Wednesday (W), 14:15-15:15h Sprint Review and Planning (meeting block)

Initially, no Daily Scrum for the T4.5 team was scheduled

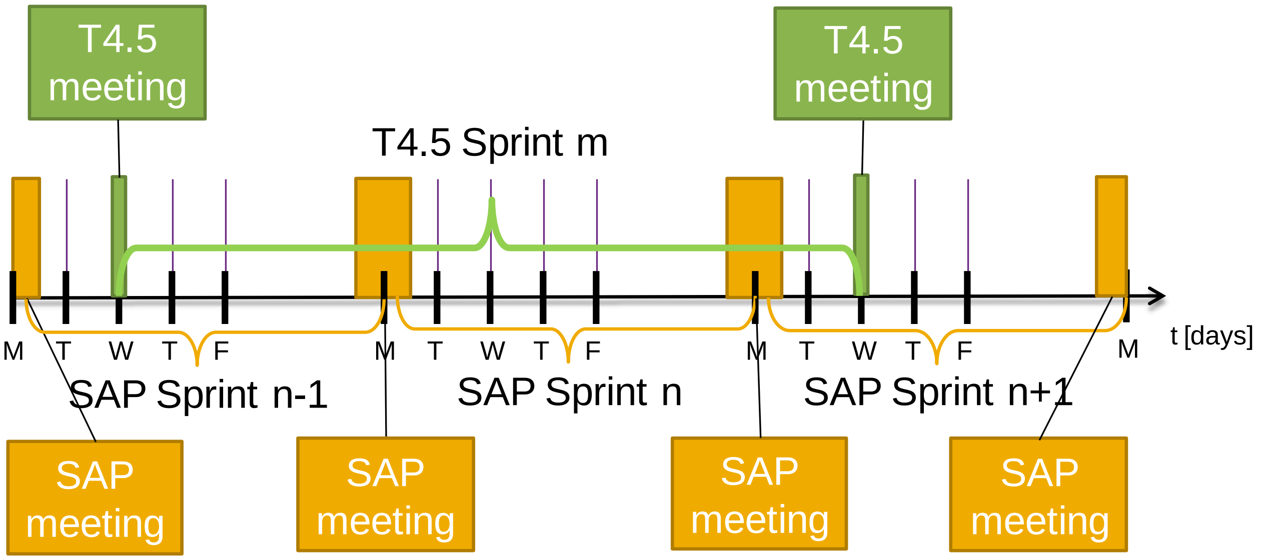


Figure 13: Weekly Sprint pattern of the SAP team together with the Bi-weekly Sprint pattern of the T4.5 team showing Sprints and meeting blocks.

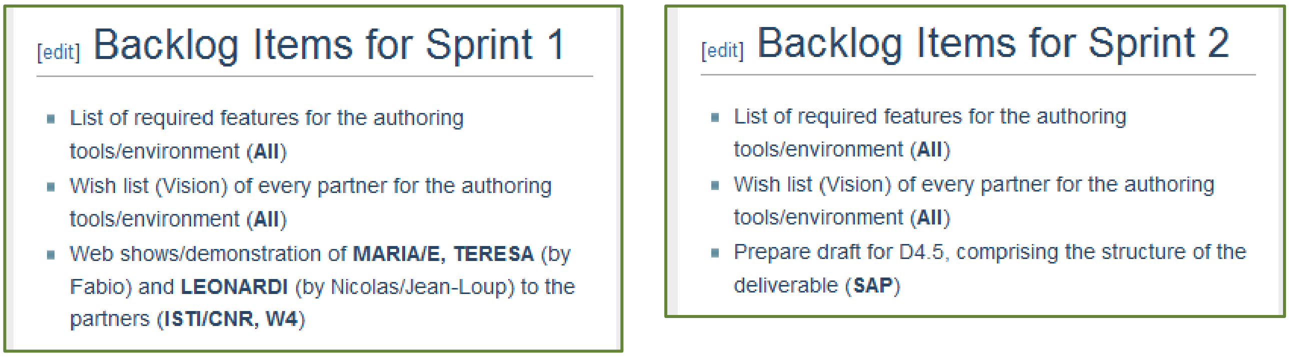
Figure 13 illustrates the weekly sprint pattern: inside the T4.5 meeting block, the usual three types of Scrum meetings can be found. In the Review meeting and to track progress, each member of the T4.5 team shares recent results they have achieved in the form of completed Backlog Items in the last sprint. The Product Owner signs-off respective Backlog Items according to his acceptance-criteria as Done. The Retrospective was not schedule but inserted on demand.

During the Sprint Planning, the Product Owner presents the Backlog Items to the T4.5 team. The team member(s) then commit to one or more Backlog Items. The content of the Backlog Items and the commitment are recorded in the wiki for the T4.5 team.

5.3 Tools

A wiki, which is part of the Serenoa wiki[4], serves as the main tool to support the agile process.

The Backlog Items and the issues spotted during the Retrospective were recorded some hours after the meeting block in the wiki for the T4.5 team[5] as shown in Figure 14.



[edit] Backlog Items for Sprint 1

- List of required features for the authoring tools/environment (**All**)
- Wish list (Vision) of every partner for the authoring tools/environment (**All**)
- Web shows/demonstration of **MARIA/E, TERESA** (by Fabio) and **LEONARDI** (by Nicolas/Jean-Loup) to the partners (**ISTI/CNR, W4**)

[edit] Backlog Items for Sprint 2

- List of required features for the authoring tools/environment (**All**)
- Wish list (Vision) of every partner for the authoring tools/environment (**All**)
- Prepare draft for D4.5, comprising the structure of the deliverable (**SAP**)

Figure 14: Backlog Items of the T4.5 team for Sprint 1 (Planning Meeting on 2011-04-06) and Sprint 2 (Planning Meeting on 2011-04-20)

In the main page of the wiki the patterns of the meeting block, the connection details and the links to each Sprint page can be found as shown in Figure 15 a).

[4] http://serenoa.morfeo-project.org/wiki/index.php/Main_Page (access only for Serenoa partners)

[5] http://serenoa.morfeo-project.org/wiki/index.php/WP4BWCall (access only for Serenoa partners)

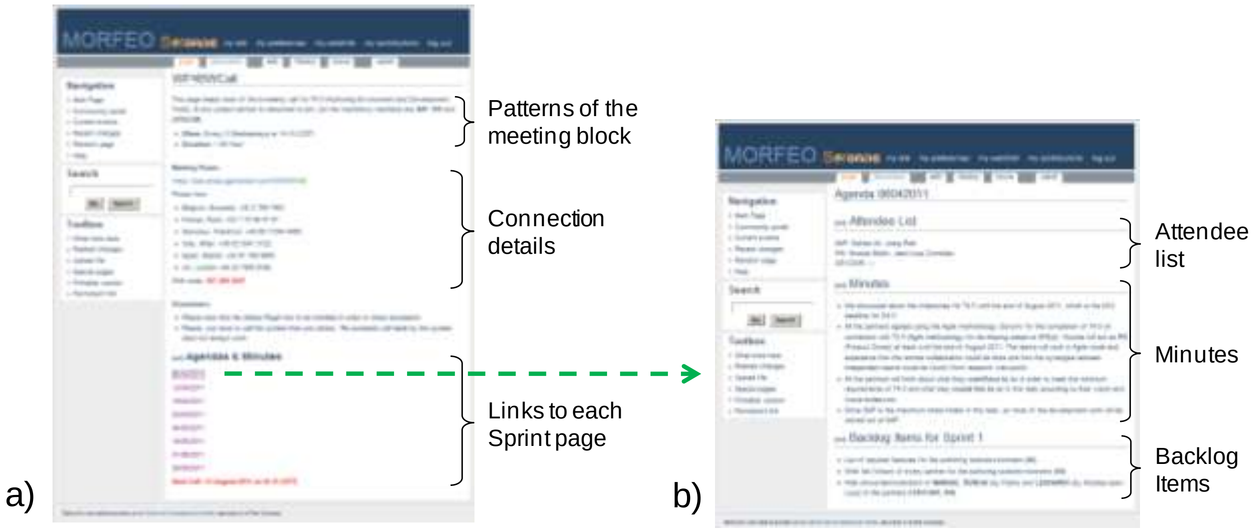


Figure 15: Main page of the wiki for the T4.5 team (WP4BWCall) with patterns of the meeting blocks, connection details and links to the Sprint page

Following a link to the Sprint page information about the list of attendees the outcome of the Sprint Retrospective (minutes) and the list of Backlog Items for the next Sprint can be found (see Figure 15 b)).

6 Results on the agile development process

The results presented in this chapter are mainly oriented in two directions. The first one concerns the integration of the agile process, as shown on the example of processing some Backlog Items. The respective section starts with answering some questions which are arising from matching the Sprint Patterns of the SAP team and the T4.5. The second direction is the result of a survey presenting a first retrospective of the team members on the implementation of the agile process.

6.1 Matching of Sprint Patterns

As the SAP team was already performing in Scrum mode since the fourth quarter of 2010 and the T4.5 only started its work in the second quarter of 2011, three questions arose concerning the alignment of the two teams.

How are the Backlog Items processed between the two teams?

As it can be seen in Figure 13, the Product Owner of the SAP team has two days (Thursday and Friday) in Sprint n-1, to include the Backlog Items from the Product Owner of the T4.5 team (given on Wednesday) as new Backlog Items in Sprint n. At the end of Sprint n (on Monday), the Product Owner of the SAP team can accept the Backlog Items as Done. The SAP team can then present the Backlog items two days later (on Wednesday) to the Product Owner of the T4.5 team.

Who decides if a Backlog Item is Done?

In this process both Product Owners need to accept the Backlog Item as Done. The Product Owner of the SAP team acts as a Quality Gate. If he does not accept the BIs on Monday they will not be presented to the PO of the T4.5 team.

What happens if the Backlog Items are not accepted as Done by the PO of the SAP team?

In detail the SAP team would inform the PO of the T4.5 team that they de-commit the BIs. The PO of T4.5 team has then the possibility to present the BIs again on Wednesday for the next Sprint. Thus, the non-acceptance of the PO of the SAP team leads to the same result as a non-acceptance of the PO of the T4.5 team.

6.2 Agile process integration

The following section shows the integration of the agile process through the processing of Backlog Items in the context of the above presented matching of Sprint Patterns. A simple example was chosen as the storyline but the general process and the modes of interactions hold true also for more complex tasks.

The goal of the presented task was the demonstration of two tools (LEONARDI and MARIA) by two partners (W4 and ISTI). As shown in Figure 16 the storyline starts with the presentation of a Backlog Item named “Attend and post process bi-weekly T4.5 call” to the SAP team at beginning of SAP Sprint 18 (1).

During this meeting the Backlog Item “Web shows/demonstration of MARIA/E, TERESA (by Fabio) and LEONARDI (by Nicolas/Jean-Loup) to the partners (ISTI/CNR, W4)” is presented to the T4.5 team (2). The PO of the SAP team took this BI from the PO of the T4.5 and included it his Backlog Item Planning. Thus, in the beginning of SAP Sprint 19 the BI named “Attend Webinar for LEONARDI tool from W4” was presented to the SAP team (3). The W4 team presented the LEONARDI tool during SAP Sprint 19 (4). The PO of the SAP team accepted this Backlog Item as Done at the end of SAP Sprint 19 and presented two Backlog Items to the SAP team at beginning of SAP Sprint 20. One was the missing Backlog Item “Attend Webinar for MARIA tool from ISTI” (5) and the second was the re-occurring “Attend and post process bi-weekly T4.5 call” (6). The ISTI team presented the MARIA tool during SAP Sprint 20 (7). During following bi-weekly T4.5 call the Backlog Item “Web shows/demonstration of MARIA/E, TERESA (by Fabio) and LEONARDI (by Nicolas/Jean-Loup) to the partners (ISTI/CNR, W4)” was accepted as Done from the PO of the T4.5 team an removed the list of Backlog Items for T4.5 Sprint 2 (8).

A complete list of Backlog Items of the T4.5 team so far and the related T4.5 Sprints can be found in the annex.

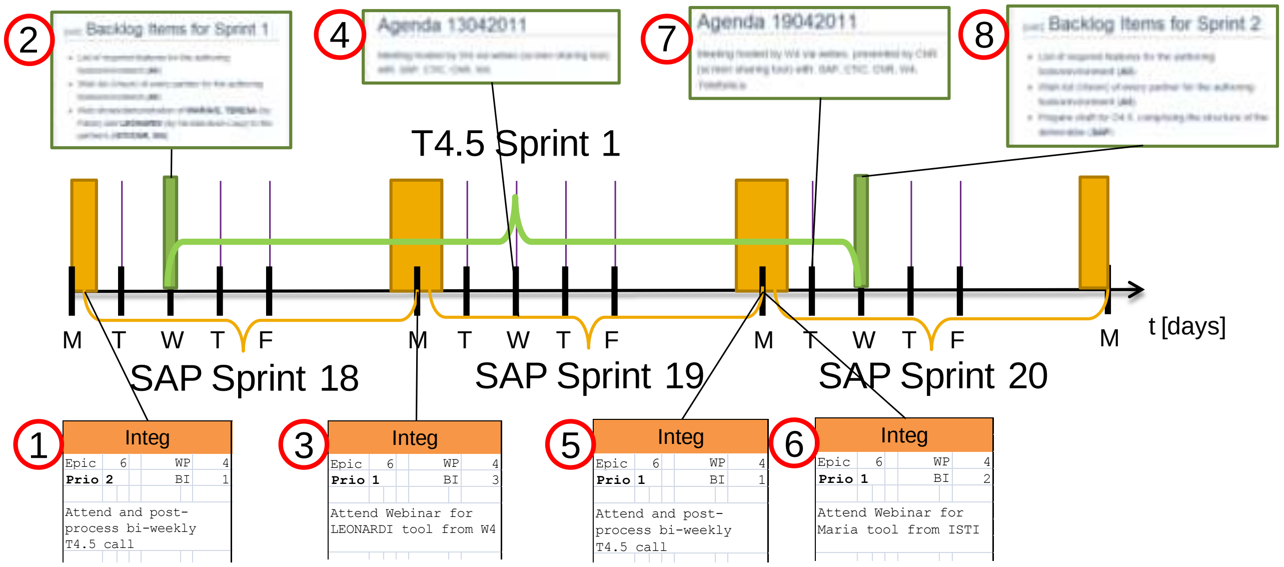

2
Backlog Items for Sprint 1
4
Agenda 13042011
7
Agenda 19042011
8
Backlog Items for Sprint 2
T4.5 Sprint 1
M T W T F M T W T F M T W T F M
t [days]
SAP Sprint 18
SAP Sprint 19
SAP Sprint 20
1
Integ
Epic 6 WP 4
Prio 2 BI 1
Attend and post-process bi-weekly T4.5 call
3
Integ
Epic 6 WP 4
Prio 1 BI 3
Attend Webinar for LEONARDI tool from W4
5
Integ
Epic 6 WP 4
Prio 1 BI 1
Attend and post-process bi-weekly T4.5 call
6
Integ
Epic 6 WP 4
Prio 1 BI 2
Attend Webinar for Maria tool from ISTI

Figure 16: Backlog Item processing as a storyline on the example of a task for presenting demonstrators on the web.

6.3 Prototype increments delivered

To reach the final goal of the first prototype release, several prototype increments need to be delivered by the T4.5 team. Some of the Backlog Items represent which have been accepted as Done by the Product Owner of the T4.5 team represent these small increments.

1. (Non)Functional Requirements for Authoring Tools: http://serenoa.morfeo-project.org/wiki/index.php/Reqs_authoring_tools
2. Deliverable 4.5.1 Authoring Environment V0.1 https://colabora.tid.es/serenoa/Shared%20Documents/WorkingDocuments/WP4/Authoring%20Tools/Deliverable4.5.1_v0.1_SAP.docx
3. Proposed Technologies for the Development of Authoring Tools http://serenoa.morfeo-project.org/wiki/index.php/Proposed_Technologies_for_the_Development_of_Authoring_Tools

6.4 Evaluating the Cross-location aspects of the agile methodology

In order to know the developer's perspectives on the cross-location aspects of the agile methodology, a questionnaire was created, tested and applied. The main goal was to identify the experience about SDLC (software development life cycle) methodologies of the members of the development team and gather their opinions about the adoption of an agile methodology with cross-located teams.

The questionnaire is organized in two parts: Profiles and Perspectives. The first one aims at gathering information about the profile and the background experience of the team members. The second part of the questionnaire concerns information about perspectives and opinions of the team member regarding the application of the Scrum methodology in a cross-located environment (as positive and negative aspects, impact, and suggestions).

Table 3 illustrates the questionnaire applied. SurveyMonkey[6], an online tool for creating, applying and analysing the results of surveys, was used to support this process.

[6] www.surveymonkey.com

Table 3. Questionnaire

Profile					
Name					
Main function	development	design	model	test	others
Duration	less than 1 year	1 to 5 years		more than 5 years	
Previous experience with agile methods	no	yes	Describe: ______________________		
Known methods	Waterfall	V	Incremental	Evolutionary	
	RUP	Spiral	Scrum	None	Others
Perspectives					
Do you believe Scrum affects the system development? In which aspects? And how?					
Which were the aspects of Scrum that you consider positive for the project development? Which were the negative points?					
Which is your opinion about Cross-located Scrum? How does the remote location of the partners affect the methodology? Was it feasible? Which were the main problems faced, if any?					
Which tools have you used and recommend for Scrum with cross-located partners? Mainly regarding: communication support, control versioning, and project management.					
Do you have any suggestions, recommendations or comments that could contribute with the application of Scrum for a cross-located context?					

All project members involved in the T4.5 were invited to answer the questionnaire. 8 valid replies were obtained. All the responders are male and work for European companies (W4, and SAP) and Research Institutes (ISTI-CNR). The analytics and commentaries about the results obtained are presented and discussed below.

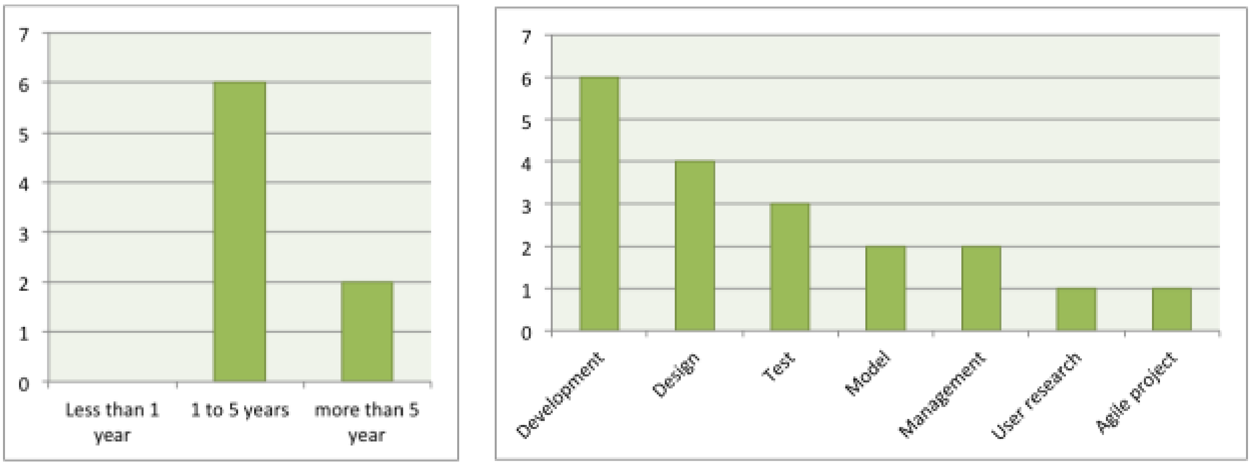


Figure 17. Team members regarding the years of experience and the main functions performed.

Most of the stakeholders (75%) reported to have from 1 to 5 years of experience in their main functions; only 25% (2 out of 8) declared to have more than 5 years of experience, and none had less than 1 year of experience (see Figure 17). This reflects a team that has already achieved a certain level of maturity in this domain. Concerning the main functions performed, 75% (6 out of 8) declared themselves as developers. 50% of them (4 persons) also work with design functions, 3 with tests, 2 with model, 2 with project management, and 1 mentioned as main functions also user research and agile project (see Figure 17). These numbers reflect a team composed by multiple expertise domains but mainly with experience in development tasks.

As the graphic illustrated by Figure 18 shows, 50% of the participants (4 out of 8) declared to not have any experience with SDLC methodologies, the other half of the participants declared to have worked with Waterfall and V-Model (3), Incremental and Scrum (2), and 1 person had already worked with Spiral, RUP and Evolutionary methods. ASAP was mentioned by one person as another SDLC methodology also used. Although the team declared to have experience with multiple SDLC methodologies (8 in total), the fact that only 2 persons (25% of the participants) had previous experience with Scrum can be seen as a critical issue for applying cross-located Scrum, once efforts to train the majority of the team would be required.

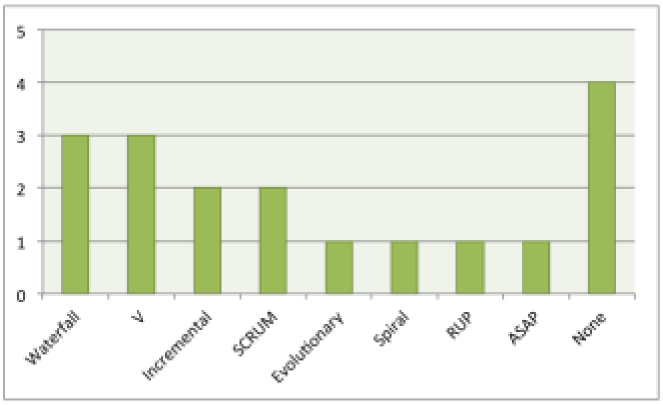


Figure 18. SDLC Methodologies according to the background of the members

The second part of the questionnaire, Perspectives, was composed by 10 open questions aiming to gather the perceptions of the participants about the application of Scrum in a cross-located context. The results obtained were analysed and they are summarized and discussed below.

Regarding the impact of Scrum in the SDLC, it is clear for all participants that Scrum affects the development of system, mainly regarding the way in which its management aspects are organized. The impact factors mentioned by the participants include: a faster response to changes, a better communication between stakeholders, more focus in the team work, earlier detection of potential issues, more transparency in issues, easier detection of problems, faster addressing of problems, and the development efforts more optimized regarding the end user expectations. This reflects the fact of the Sprints having a short duration, which concentrates the work in limited actions, more focused and allowing future discussion about the next actions. This structure contrasts with V cycles, that are long and start by the development of the infrastructure of the system (which can produce results different than those that the end user actually expects).

The positive aspects that the participants highlighted were: the transparency of the methodology; the quick response to changes; the achievement of better results, once they tend to be closer to the expectations; more commitment of the team members; the removal of additional politics; clear goals and expectations; a strong participation of the project owner; the avoidance of heavy delays during the development phase and early detection of the shift capacity to postpone or cancel complex functionalities that are of limited interest for the project owner. Most of the positive aspects remarked seem to be a direct consequence of the short sprints adopted, once they force a frequent and systematic communication between stakeholders. Although the short sprints demand extra efforts for the team members, they provide many benefits for the SDLC, as the participants stated.

The negative aspects that the participants remarked were: the higher costs for development, once many iterations may cause many changes; the strong commitment of all the members of the team (including not only the developer but also the project owner) as a strict requirement to avoid delays; the cross-location aspect that may cause difficult synchronization; the constant focus on project owner needs; the short sprints that may strongly postpone some tasks of limited values for the project owner (in case he prefers features as code clean up, upgrade of libraries, documentation), this risk depends, however, on the product owner experience (if he asks for new features all the time and skip the consolidation of existing code...); the periodic meetings for sprint have a time cost, once they require preparation, frequent demos, requests for new features. Most of the negative aspects remarked by the participants are also a direct consequence of the short sprints adopted, once they demand time, efforts and constant synchronization. Although the short sprints demand additional efforts for the team, they provide many benefits for the SDLC, as the participants stated. Therefore, it is a challenge for the Scrum team to balance the trade-off caused by having frequent meetings, which on one hand improves the communication and allows quick responses to changes, and on the other hand requires additional efforts of the team members.

Regarding the cross-location aspect in applying Scrum, the participants highlighted: the higher possibility of no attendance to the meetings; a potential negative impact in the team productivity caused by non-collocation; the need of longer sprints; and a higher risk caused by managing parallel activities.

Regarding the tools used to support the Scrum methodology, they were considered as classical tools (not dedicated to a specific methodology) as: audio conferencing, screen sharing and phone. The Wiki was pointed as the most important tool used, followed by: Atlassian-JIRA, MS TFS, hansoft.se and google-wave. Besides, it was mentioned that no collaborative development tool was used (such as common code versioning systems or coding rules).

Regarding the comments, suggestions and contributions provided by the participants, they believe that cross-located Scrum can work appropriately for identifying requirements and design, but for development tasks, it was recommended, for the different modules of the project, to be locally developed. They recommended also tight synch-up, some face-to-face sessions at the initial phase and in regular intervals, stick to commitments (including time-boxes for deliveries). One of the participants believes that the SERENOA experimentation is not representative of a traditional Scrum development mainly because the team members are not working on the same development and standup meetings with post-its are not occurring; however he also believes that the actors can play their roles without problems in a cross-located environment, even if some aspects require some particular attention.

7 Conclusions

7.1 Summary

The results have shown that the Serenoa project has successfully implemented a process following an agile methodology. At this stage the agile methodology is limited to a single task within work package 4 “Algorithms, Runtime and Tools”. The task T4.5 “Authoring environment and development tools” was chosen for several reasons.

- The number of participants in terms of partners according to the DoW is limited to three: by limiting the number of partners to three it takes less time to reach a stable state in the process of implementing the agile methodology;
- The participants are internationally cross-located: to reach an internationally cross-located distribution of teams is the major goal of our research on agile methodologies within Serenoa. With W4 from France, ISTI-CNR from Italy and SAP from Germany all three partners are internationally cross-located;
- The participants had different levels of expertise in agile methodology: a secondary goal is to research the effect of teams with different levels of expertise in agile methodologies. W4 has a strong background in applying agile methodologies in projects and produce products, which support agile methodologies. The W4 team within Serenoa is currently not working in an agile mode. ISTI-CNR had until now no experiences with agile methodologies. The SAP team within Serenoa is running since the beginning of the project in Scrum mode. Thus, the partners can be seen as diverse in terms of levels of expertise in agile methodology.
- The deliverable of the task is a prototype (P): the state of the art on agile methodologies comes to a large extend from projects in which software is produced. In order to better refer our results to the existing knowledge a task with a deliverable of nature prototype was chosen rather than a deliverable of nature report.

The example of giving demonstration of two tools by two partners was chosen to show the successful implantation of the agile process. The member of the T4.5 team have understood and accepted agile concepts like Sprint, Planning Meeting, Review Meeting, Backlog Item, Product Owner and Done criteria.

With the commitment to Sprints, the team members accepted that there are static cycles of two weeks that are flanked by Planning and Review Meetings. With the commitment to Backlog Items the team members accepted that there are certain tasks that need to be done during the Sprints, that those tasks are given by the Product Owner, and that the team members have the choice to commit to a Backlog Item during the Planning Meeting. The team members accepted that the Product Owner prioritizes the Backlog Items according to his long term view on the prototype. The team member also accepted that the Product Owner decides during the Review Meeting if a Backlog Item fulfils his Done criteria.

The results have also shown that the agile process of the T4.5 team was seamlessly integrated into the already running agile process of the SAP team. The Product Owner of the T4.5 team took from the viewpoint of the Product Owner of the SAP team the role of an outside event, like a costumer’s or an executive’s request. By having a Sprint duration ratio of 1:2 between the T4.5 team and the SAP team and an offset between the meeting-blocks the integration worked out fine.

The results of the survey tend to prove that Scrum is perceived as feasible in a cross located manner, however it is necessary to enforce the strong commitment of all the team members and decide a Sprint duration that is beneficial for the project, without demanding too much efforts of the stakeholders regarding the accomplishment of additional tasks.

7.2 Future Work

From the current state of the team, there are several lines that can be followed as future works.

One possibility is to apply further concepts from the agile methodology. Currently it is not possible to track the performance of the T4.5 team along the Sprints. In order to achieve a performance tracking some kind of effort measure needs to be introduced and attached to the Backlog Items. The effort unit could be in hours, days or more abstract in story points. This would then automatically trigger the introduction of effort estimation, like Planning Poker. The performance could then be visualized in Product Burndown Charts.

Based on an on-going discussion there might be different versions of the prototype that involve also additional partners. This might lead to increasing the number of partners participating in the T4.5 team. It will be interesting to investigate the influence of these changes to the T4.5 from an agile perspective.

Finally the Serenoa consortium might also decide to apply the agile methodology to an additional task in which prototypes are developed. It will be interesting to verify if the already gained knowledge from the T4.5 team can be applied for a fast adoption in another team.

8 References

James Grenning, "Planning Poker", 2002, Renaissance Software Consulting

Mike Cohn, "User Stories Applied", 2004, Addison Wesley, ISBN 0-321-20568-5

Mike Cohn: Agile Estimating and Planning, 2006, Prentice Hall, ISBN 0-13-147941-5

Jeff Sutherland, et al, “Distributed Scrum: Agile Project Management with Outsourced Development Teams”, 2007, Proceedings of the 40th Hawaii International Conference on System Sciences

Joe Krebs, “RUP in the dialogue with Scrum” 2005, IBM developer Works, Available at: http://www.ibm.com/developerworks/rational/library/feb05/krebs/

Ken Schwaber, “Agile Project Management with Scrum”, 2004, Microsoft Press, 1, ISBN 0-735-61993-X

Ken Schwaber, “The Enterprise and Scrum”, 2007, Microsoft Press, 1, ISBN 0-735-62337-6

Ken Schwaber and Jeff Sutherland, “The scrum guide”, 2011, Retrieved from http://www.scrum.org/scrumguideenglish/

Acknowledgements

- TELEFÓNICA INVESTIGACIÓN Y DESARROLLO, http://www.tid.es
- UNIVERSITE CATHOLIQUE DE LOUVAIN, http://www.uclouvain.be
- ISTI, http://giove.isti.cnr.it
- SAP AG, http://www.sap.com
- GEIE ERCIM, http://www.ercim.eu
- W4, http://w4global.com
- FUNDACION CTIC http://www.fundacionctic.org

Glossary

- http://serenoa.morfeo-project.org/wiki/index.php/CommonGlossary
- http://www.scrumalliance.org/articles/39-glossary-of-scrum-terms
- Agile Development: iterative software engineering with open scope, focusing on working software over comprehensive documentation
- Agilia: W4's application for managing and reporting purposes in agile (scrum-like) projects.
- AMDD: Agile Model Driven Development. AMDD is the agile version of Model Driven Development (MDD). MDD is an approach to software development where extensive models are created before source code is written.
- B2B: Business-to-business. B2B describes commerce transactions between businesses, such as between a manufacturer and a wholesaler, or between a wholesaler and a retailer.
- BPM: Business process management. BPM is a holistic management approach focused on aligning all aspects of an organization with the wants and needs of clients.
- BUSINESS FIRST: W4's software suite for designing and executing composite, collaborative business applications.
- Iterative Development: Software engineering activities are performed frequently in short and repetitive cycles, with feedback from stakeholders after each iteration
- LEONARDI: W4's software suite for designing and executing composite, collaborative business applications
- MDA: Model Driven Architecture. MDA is a software design approach for the development of software systems.
- PIM: Platform Independent Model. PIM in software engineering is a model of a software system or business system, that is independent of the specific technological platform used to implement it.
- PSM: Platform Specific Model. A PSM is a model of a software or business system that is linked to a specific technological platform (e.g. a specific programming language, operating system or database
- RUP: Rational Unified Process. The RUP is an iterative software development process framework created by the Rational Software Corporation, a division of IBM since 2003.
- Waterfall model: The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall).
- XP: Extreme Programming. XP is a software development methodology which is intended to improve software quality and responsiveness to changing customer requirements.

Annex

Backlog Items of the T4.5 team and related T4.5 Sprints

Sprint 1 06.04 – 20.04.2011	List of required features for the authoring tools /environment (All)	Wish list (Vision) of every partner for the authoring tools /environment (All)	Web shows /demonstration of MARIA/E, TERESA and LEONARDI to the partners (ISTI, W4)
Sprint 1 Review 20.04.2011	Not DONE: Planned for next Sprint	Not DONE: Planned for next Sprint	Done
Sprint 2 20.04. – 04.05.2011	List of required features for the authoring tools/environment (All)	Wish list (Vision) of every partner for the authoring tools /environment (All)	Prepare draft for D4.5, comprising the structure of the deliverable (SAP)
Sprint 2 Review 04.05.2011	Not DONE: Planned for next Sprint	Not DONE: Planned for next Sprint	Not DONE: Planned for next Sprint
Sprint 3 04.05. – 18.05.2011	List of functional and non functional requirements for the authoring tools /environment (All Serenoa)	Wish list (Vision) of every partner for the authoring tools /environment (All Serenoa)	Prepare draft for D4.5, comprising the structure of the deliverable (SAP)
Sprint 3 Review 18.05.2011	Done: Prod 1	Done	Done: Prod 2
Sprint 4 18.05. – 01.06.2011	Careful Review /Go through of functional /non functional requirements for the Authoring tools /environment and compare them with the requirements defined in D1.1.1 in order to identify any conflicts (All)	Explore the (Web) technologies which can be helpful in implementing the requirements of Authoring tools/environment (All)	
Sprint 4 Review 01.06.2011	Done	Done: Prod 3	
Sprint 5 01.06. – 29.06.2011	First version of mockup for authoring environment (SAP)	Detailed description of list of modules (W4, ISTI)	