INTRODUCING THE SCRUM FRAMEWORK AS PART OF THE PRODUCT DEVELOPMENT STRATEGY FOR THE ALBA CONTROL SYSTEM

G. Cuni, F. Becheri, D. Fernandez-Carreiras, Z. Reszela, S. Rubio-Manrique, ALBA-CELLS Synchrotron, Cerdanyola del Vallès, Spain

Abstract

The Controls Section at the Synchrotron ALBA[1], produces and supports the software to operate the accelerators, the beamlines and the peripheral laboratories. It covers a wide range of disciplines like vacuum, motion, data acquisition and analysis, graphical interfaces, or archiving. Since the installation and commissioning phases, we have been producing the software solutions mostly in single-developer projects based on the personal criteria. This organization scheme allowed each control engineer to gain the expertise in particular areas by being the unit contact responsible to develop and deliver products. In order to enrich the designs and improve the quality of solutions we have grouped the engineers in teams. The hierarchy of the product backlogs represents the desired features and the known defects in a transparent way. Instead of planning the whole project upfront, we try to design the products incrementally and develop them in short iterations mitigating the risk of not satisfying the evolving user requirements. This paper describes the introduction of the Scrum framework as the product development strategy in a service oriented organization like the Computing Division at Alba.

INTRODUCTION

Alba is a 3rd generation Synchrotron Light facility located in Cerdanyola del Vallès, (Barcelona), commissioned in 2012 with seven operational soft and hard X-ray beamlines, devoted to experiments in biosciences, condensed matter (magnetic and electronic properties, nanoscience) and materials science among others. Nowadays, two new beamlines are in construction (infrared microspectroscopy and low-energy ultra-high-resolution angular photoemission for complex materials).

In addition to the Controls Section, the Computing and Controls Division has other 3 sections devoted to management systems software, electronics design and support or Information Technology systems and network administration. The mission of the division is to offer services and give support to the beamlines, laboratories and accelerators, as well as to the whole installation to produce high quality experiments. The range of services comprises hardware and software solutions for control systems, personal safety, equipment protection, data acquisition, data analysis, high performance computing, document management, and networks and communications among others.

ORGANIZATION OF THE CONTROL SOFTWARE DEVELOPMENT DURING THE INSTALLATION AND EARLY OPERATION

The Controls Section was setup in the early 2005. During the phases of design, construction and installation the size of the team was progressively increasing until the current 16 members. The section was responsible for the design, installation and commissioning of projects in a wide range of disciplines, for both accelerators and beamlines. The fact of being a single support group for the whole facility showed important competitive advantages during the installation, where peak loads, could be better managed balancing manpower among the different customers. The software design, development, and the protection systems were scheduled in parallel for the accelerators and the beamlines, although the installation and commissioning of the accelerators engendered peak loads, which often took precedence in order to match the milestones and manage the critical paths for the deliveries. The project and the group relied on common tools, technologies, international collaborations and internal-developed transversal libraries and frameworks, crucial for keeping the service support manageable and the service level agreements.

The different products and subsystems were progressively delivered, and run in production. At this stage, the section started to share efforts between the on-going projects and the service support with a number of bugs and defects to fix at the beginning, and later an increasing number of service requests and requests for change. The service desk relies on a ticketing system where every issue is registered with the relevant information to be classified by priority, unit (e.g. an accelerator's group, a specific beamline, etc.), and service (e.g. a subsystem like vacuum, motion, graphical interfaces, etc.). In this scenario, although all products were based on common tools, they were mostly developed by individuals, who ended up taking the responsibility for the related tickets (bugs or new feature requests), and therefore the particular knowledge of the product could not be spread.

Once the Control System has been delivered, during the operation, new features, bug fixes, and advanced functionalities need to be properly scheduled for deployment and commissioning in very restrictive time slots (mostly on shutdowns or "maintenance days"). Furthermore, during operation, the support and maintenance tasks for the running products gain priority

ISBN 978-3-95450-148-9

against new developments due to its critical effect on the stability, the performance, or new capabilities of the installation.

The combination of service management best practices and project management methodologies for the development of controls software has proven to be successful [2] allowing us to give support and rapidly respond to incidents as well as periodically provide updated versions of the software packages of the Alba Control System [3].

SPREADING THE KNOWLEDGE

Taking responsibility for a project is often a great motivation for a software developer. Making one single person responsible for the development, installation and support of a particular product is in many cases efficient. The developer feels immediately committed to the project and motivated for delivering a high quality product in time and within specifications. However, sometimes a project may be unmanageable by a single person, or in case of a deployed product in operation, the necessary knowledge needs to be spread and transferred to a bigger group to accomplish the required service level (i.e. 24/7). Having regular meetings among all the members of the group help to share information, functionalities, feature requests etc., so the team gets updated on the evolution of the products. However, although weekly meetings keep the group informed and the dedicated collaborative brainstorming sessions are also setup occasionally to inject fresh ideas into the project dynamics, the ownership of the products remains in the developer hands.

During the operation and support of the service, having the specific know-how of complex products centralized in one person has important risks. An urgent issue may be raised and require dedication of a person who had already committed 100% of his work time to another issue, a scheduled maintenance task or a delivery on a fixed deadline. Being the only developer able to do some tasks creates also internal stress when equally urgent high priority issues appear. Besides, the developers can get stuck inside their own knowledge, unable to reach beyond their projects' scopes, not being able to be open to other projects or perform other collaborative activities. Moreover, if a developer leaves the team, there is an impact on the pace of product delivery. The orphaned projects need to be distributed among the rest of the team, who although know the tools, need to learn about that particular project, and the newcomer will only be able to take over some projects after long learning period. The combination of the single-developer projects with the service support of the existing products given by individuals leads to the following difficulties:

- Single developer bottlenecks
- Single person of contact for services has to take decisions on priority when new issues arise
- The team knows the highlights of projects but not the insights
- Developers are not aware and therefore do not profit from other similar solutions already provided by the group
- Newcomers (50% of the team was renewed in the last three years) need to learn common tools, frameworks, and libraries. This requires a huge effort from senior members to setup trainings and give support to the newcomers

Once the Alba's operation started, for the reasons above stated, we decided to explore other alternatives for managing projects and services in parallel. Agile methodologies focus on the collaborative work and empower team culture especially needed when interests of the group take precedence over personal projects. Sometimes, projects require an extra effort due to deadlines or events that cannot be postponed, and a team has to work together for a while, leaving the individual projects in a frozen state.

THE SCRUM FRAMEWORK

The Scrum [4] process was conceived in the early 90's, based on a research stressing the importance of teams and indicated the excellent performance achieved when teams, as small and self-organized, are fed with objectives instead of tasks: *"the best teams are those that are given direction within which they have room to advice their own tactics on how to best head toward their joint objective. Teams require autonomy to achieve excellence"*.

Scrum is a framework for developing and sustaining complex software products based on an empirical process approach where more is unknown than known and predictions have little value given a high rate of change and uncertainty. In this environment, knowledge comes from experience and making decisions based on what is known. It is not a process or a technique for building products, but a framework that makes clear the relative efficacy of the product management and development practices enabling improvement.

The Scrum Team

There are three core roles that compose the Scrum Team: a **Product Owner**, the **Development Team**, and a **Scrum Master**. The Scrum Team is self-organized in order to accomplish the objectives, and has all the skills required to execute the work to be done. The Product Owner is responsible for maximizing the value of the product and the work done by the Development Team ensuring that **Product Backlog** items have clear definitions, they are ordered to best achieve goals, and confirming that the Development Team members understand them to the level they need. These people are the professionals that create the **Product Increment** functionalities based on the Product Backlog items, and the accountability belongs to the Development Team as a whole. The Scrum Master is the facilitator whose responsibility relays on ensuring that the Scrum Team as a group follows the rules to maximize the value to be created, and promoting the self-organization.

ISBN 978-3-95450-148-9

Scrum Events

Scrum suggests a set of events with a predefined maximum time length to fulfil all the coordination and management activities. The **Sprint** is the core event, it consists of the **Sprint Planning**, **Daily Scrums**, development work, the **Sprint Review** and the **Sprint Retrospective**. Sprints duration is often limited between a week and a month. Delivering products iteratively and incrementally maximizes opportunities for feedback and reduces the risk of changing requirements or increasing complexity. The Sprint Planning is a time-boxed event where next possible deliverable increments are selected, discussing how will be the work done. Daily Scrums are events of fifteen minutes fixed length, where the Development Team shares progress since last Daily Scrum, organizes the work paying attention on any blocking situation that may prevent reach the planned increments. At the end of the Sprint, the Sprint Review checks the work that has been done, and what has been not finished based on the original planning. The Sprint Retrospective analyses how the team has performed in terms of communication, coordination, and means to achieve the work, with the objective of identifying what can be improved for the next Sprint.

Scrum Artifacts

The Product Backlog is the main artifact of Scrum, it is a unique ordered list of items that describe what is still needed to be done for a particular product and it is responsibility of the Product Owner to expose which are the current known requirements. The Sprint Backlog contains the items that have been selected, and the tasks identified in the Sprint Planning which are needed to do the work. At the end of the Sprint, the Product Increment is a concept that combines all Product Backlog items that have been completed.

EMBRACING THE SCRUM FRAMEWORK

Evaluation Phase

By the end of 2013, the Controls Section had three individual projects that required an internal boost. There was a big opportunity to evaluate if Scrum would help in the organization, execution, and coordination endeavors. During the first half of 2014 this exercise was started by defining the roles and following the Scrum rules and events, planning sprints with well-defined user stories and focusing on small product increments. It was soon appreciated the benefits of the enforced communication and implicit collaborative activities in that controlled environment, and at the end of this first implementation, there were very satisfactory results. There are plenty of documented advantages of incremental agile software development, and the list below highlights the outcomes from that research experiment:

- Designing solutions by teams instead of individuals lead to more robust products and of better quality since each member contributes from his experience and point of view
- Incremental and iterative approach helped in validating important assumptions fast what mitigated the cost of change
- Constant focus on delivering potentially shippable product increments enabled achieving earlier return of investment
- Self-organized development team did not require constant supervision and worked in a stress-less environment

These preliminary tests had proven that applying Scrum for complex software developments helps in achieving better results, and based on the size of the projects these are some recommendations:

- The best team size is from 4 to 6 members
- Two weeks sprint length has proven to be a good choice
- One controls engineer should not be part of more than one development team at a time
- Due to the support nature of the section, dedication of 60% of the time for the Sprint Backlog, keeping the other 40% for incidents, and high-priority individual service requests (which cannot be foreseen when planning) seems reasonable

Creation of the Scrum Teams

After this first contact with Scrum, a plan to enhance the Controls Section's development strategy was started. It was focused in incrementally changing the group organization for new developments and critical problems where collaborative work was clear to be an advantage.

The Scrum Master and Product Owner roles have been assigned at the beginning, sharing a common vision on how each project would enter in each team backlog. Our customers were informed about the change in how work will be planned.

During mid-2014 the first group built was the Beamline Control Systems Scrum Team. The communication channels between beamline scientists and controls-contacts didn't require any change, and it was enough to take care identifying those requests about complex problems for which design and implementation activities would be worth sharing. Those requests were then merged in a general, serialized and prioritized top-level product backlog that provided a clear view of what needed to be addressed during each sprint iteration for all beamline developments.

After few sprints, a second group mostly dedicated to internal software frameworks did the same transition, starting by defining specific product backlogs for all the individual projects that had been managed by each team member, and created a unified product backlog view that combined all the requirements that the team would have to work together when implementing them. While learning from co-workers, the developers enjoyed to contribute with their own points of view. At the end of 2014, we created a third team, and in the course of the

ISBN 978-3-95450-148-9

first quarter of 2015 we have started building the fourth and last squad.

Tools for the Scrum Artifacts

In the adoption of Scrum, it was enough to enrich the same platform that was already used for project tracking: Redmine which has plugins for managing product backlogs. Having a tool that provides at any point in time the possibility to monitor the Sprint progress is essential. The team can report progress by simply interacting with a shared board, inputting the remaining work which are very valuable inputs for the Daily Scrum, and can be simply extracted from auto-generated charts at any time.

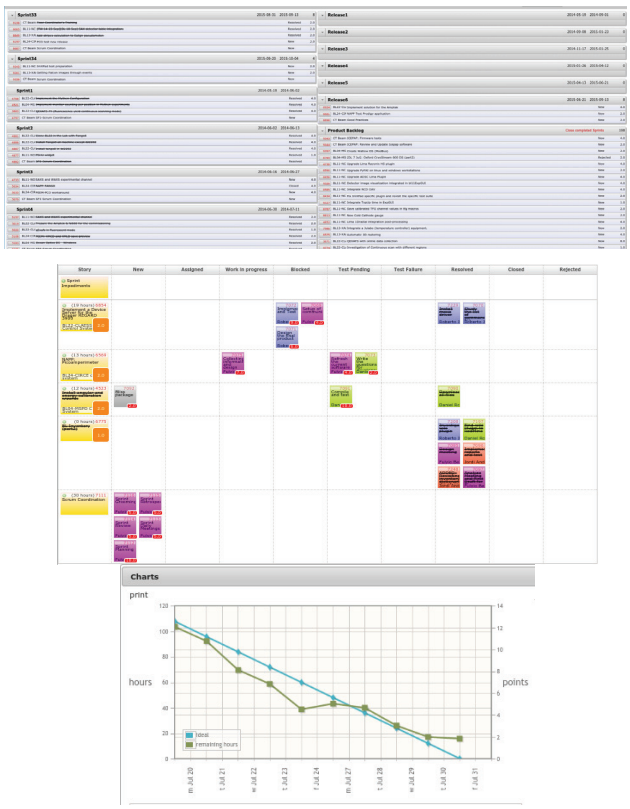

Figure 1: Team Backlog, TaskBoard and BurnDown chart.

Sprint Reviews and Retrospectives

Sprint reviews and retrospectives offer great opportunities that let information flow, and over the time, some topics evolve while others remain. Additionally, those meetings gave a chance to really keep track of the project evolution as a whole by reviewing the work from the given value perspective, and considering how it was reached. In nearly all retrospectives, one concept that repeats quite often is to bear in mind that during the sprint, one should spend some time in backlog refinement even that this task does not add any value to the current sprint product increment, it is fundamental for the next planning. Clear "definition of done" explanation in backlog items help when preparing the tasks, when developing to fulfil those requirements, because it reduces uncertainty forcing that everyone understands it. Our experience showed that it is really important to keep in mind some buffer of time reserved for support-oriented duties when calculating the sprint capabilities..

CONCLUSION

The introduction of the Scrum Framework as part of the product development strategy has proven to be very successful. It has been extremely important to start with a pilot program which facilitated the selection of the right tools, and gave hints on how the system could be tailored for our needs. Focusing on communication and the value given to our customers, as well as ensuring that there is a clear definition of what has to be done, helps during every stage of the development because all the team members are enforced to understand and agree on next steps. It is essential to have available a view of how the team is performing and what is still pending to be done within the Sprint. Also, having the Sprint Backlog defined upfront, indicates that the Product Owner admits that items not planned will be left for later phases, hence the team can concentrate the efforts in a limited set of projects. Actions like code reviews, pair programming and retrospectives about the process itself are activities that empower the team building process.

It is worth to mention that we still keep individual projects outside the Scrum Teams, mainly research activities, developments of proof of concepts and prototypes that are conceived as personal objectives and create personal and professional growth.

ACKNOWLEDGMENT

The authors would like to thank all Alba's Controls Section members for their openness in this adoption, and the collaborative mind-set when brainstorming in order to find the best matching for the Scrum theoretical framework and our specific facility environment. We would also like to thank the whole MIS section for their partnership in updating Redmine and installing the plugins needed for this purpose, as well as our customers for their cooperation and comprehension during this transition.

REFERENCES

[1] Alba website: http://www.albasynchrotron.es
[2] D. Fernández-Carreiras et al. "Using Prince2 and Itil practices for Computing Project and Service Management in a Scientific Installation", ICALEPCS2013, San Francisco, United States, TUMIB01
[3] D. Fernández-Carreiras et al. "Status of the ALBA Control System", ICALEPCS2009, Kobe, Japan, TUP090
[4] Scrum Guides website: http://www.scrumguides.org/

ISBN 978-3-95450-148-9