Essential Scrum Patterns

Mike Beedle[1], James O. Coplien, Jeff Sutherland,
Jens Christian Østergaard, Ademar Aguiar[2], Ken Schwaber
[1] New Governance, [2] Universidade do Porto
mike.beedle@newgovernance.com, ademar.aguiar@fe.up.pt

Abstract. Teams willing to start adopting Scrum usually face many problems that must be carefully addressed, one by one, in a disciplined and informed way, especially when they are not aware of the key problems and respective best ways of solving them. Based on the existing body of literature, case studies and lessons learned, this paper compiles the most essentials best practices of Scrum in pattern form, as patterns are a very effective way of communicating expertise and best practices. In addition, this paper proposes a way of organizing the overall body of literature of Scrum, where Essential Patterns act as key entry points to beginners and experts looking for more knowledge and expertise about Scrum and related disciplines.

Introduction

While some of the recurrent problems that teams typically face when doing Scrum are elementary and general, many others may be very complex and specific, although all of them require careful understanding, decomposition and analysis prior to be effectively solved.

In fact, to be able to perceive the many details of the problems and solutions you may face when doing Scrum, it is very important to know, understand and get familiarity with its most essential problems and solutions.

Essential Scrum Patterns aims not only at compiling the most essential best practices of Scrum, in the vein of the original Scrum patterns paper [Beedle et al], but also to provide more experienced users with entry points into a whole body of literature (ScrumBoL), containing other typical and advanced patterns (strategic, tactical, technical, organizational, etc), as well as relations to other sources of knowledge, in a layered and navigable way, where the first layer is their familiar Essential Scrum patterns (Figure 1).

Among all the patterns mined from Scrum practices, the Essential Scrum Patterns were considered as those conveying the most agreed upon practices, without all of which you are not doing Scrum as a whole, but only something related with Scrum. For example, if you are not doing Sprints, Product Backlog, or Sprint Backlogs you are probably not doing Scrum, even if you are holding Daily Scrums.

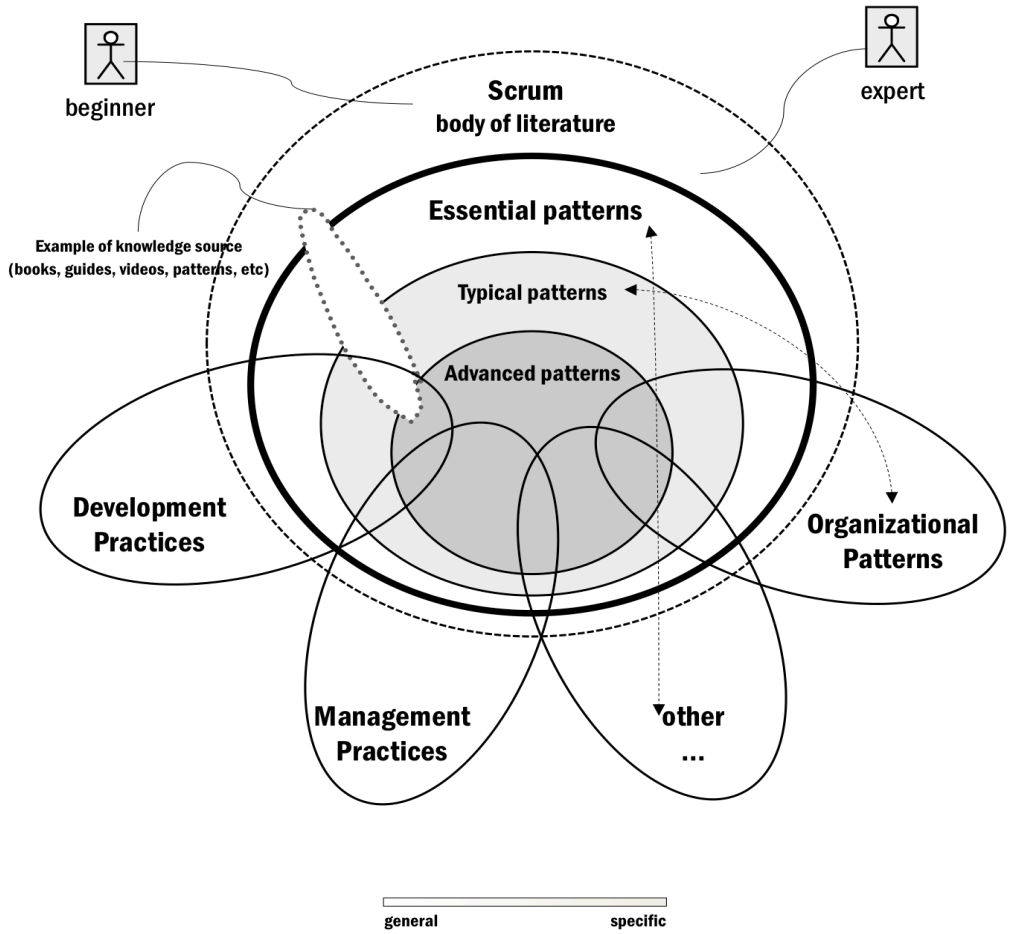

Figure 1. Essential patterns as part of the Scrum body of literature.

Pattern Form

We believe that the Essential Scrum Patterns should be written to be pedagogical, aiming at first to teach the basics, but, on second or third passes, to allow them to link to more advanced topics and get a deeper understanding. In this way, we will mimic the style of writing of Christopher Alexander, and provide a layered approach to Scrum education, until the interested learners pass completely through the gate, which would mean they understand and have experience with the pattern language as a whole.

While it is true that applying the Scrum basic framework to software projects always brings some benefit, we strongly believe that Scrum implementations would benefit considerably if they understood and incorporated other patterns that represent the community's experience contextually. For collectively intelligent teams, where the layered inspection-adaptation Scrum framework points them to problems but the team can come up with solutions themselves, this may have less value, but for teams that are stuck with a hard issue, exploiting the power of other patterns as they encounter contexts may mean either overcoming a severe obstacle, becoming the hyperproductive teams they really want to be, and in some cases, the salvation of their overall Scrum implementation. Simply said, for teams with problems they can't fix, this collection of

essential patterns may make up for their limited collective intelligence, knowledge or experience, and get them "out of the tar pit".

It is our goal that the supporting patterns represent the shared knowledge of the Scrum community, capturing the socialized experience of successful teams, and that the pattern language evolves as needed to support Scrum teams in whatever evolutionary step they might be on. This is of course very important to the Scrum community as we want and need the highest number of successful Scrum implementations as possible.

Speeding Up the Inspect-Adapt cycle

We strongly believe the patterns paradigm is very well suited to Scrum, because Scrum provides a more descriptive and adaptive process paradigm than the defined process paradigm. In Scrum, there are a layered number of "state evaluators", for example, the **Sprint Review Meeting**, or the **Daily Scrum** that allow the team to quickly react and make quick decisions to the inspected state of the team. It is in these inspections that Scrum provides a context where patterns can be applied to adapt to a new situation.

Traditionally the teams just thought of solutions to their problems, either using the collective intelligence of the team, or drawing into their collective experience, but providing patterns that are readily available to a certain context, allows Scrum teams to tap into the experience of other successful teams. This is in contrast to a team that tries to execute a defined process, and that cannot react or reinvent the process while executing it -- it has very limited ability to adapt.

Scrum adoption stories and paths

To better illustrate the application of these patterns, we want to provide some examples of how this process may actually develop in practice. Of course, the sequence of the application of these patterns would be largely contextual, as it would inspect the state of the team, find problems in context, and adapt either intelligence or patterns to the situations. Anyway, there might be some typical adoption paths and pattern sequences that are more predominant and it would be good to have them added as exemplars, in situations such as:

- new teams adopting Scrum
- existing teams starting to adopt Scrum
- existing Scrum teams starting a new product or release
- existing Scrum teams "not really doing" Scrum.

Patterns overview

The Scrum patterns considered essential are the following: PRODUCT BACKLOG, PRODUCT OWNER, SCRUM TEAM, SCRUMMASTER, SPRINT BACKLOG, SPRINT PLANNING MEETING, SPRINT, DAILY SCRUM, SPRINT BURNDOWN, SPRINT REVIEW, and SPRINT RETROSPECTIVE (Figure 2).

<to be done>

Figure 2. Essential patterns map.

In a less essential way, the following patterns were also considered, since they are very important, but not considered necessary to Scrum: RELEASE PLANNING MEETING, RELEASE BURNDOWN and VISIBLE STATUS.

Before starting to detail each pattern, we will overview all of them by summarizing their intents.

- PRODUCT BACKLOG helps on…
- PRODUCT OWNER defines a role that…
- SCRUM TEAM helps on…
- SCRUMMASTER defines a role that…
- SPRINT BACKLOG helps on…
- SPRINT PLANNING MEETING aims at…
- SPRINT organizes the…
- DAILY SCRUM helps on…
- SPRINT BURNDOWN enables…
- SPRINT REVIEW allows…
- SPRINT RETROSPECTIVE helps on…

Pattern **PRODUCT BACKLOG**

... you have defined a business segment that will be supported by one or more teams that need strategic technical direction.

* * *

It is important that everyone agrees to what needs to be done next at any given time, and that the agreement be made visible. You can't do everything at once — in fact, you can't even do two things well at the same time. It's important to have focus.

* * *

Therefore, **create a single ordered list of deliverables, called PRODUCT BACKLOG ITEMS (PBIs), that are ordered by delivery date.** The list is called a PRODUCT BACKLOG. Development staff will be driven by the work at the top of the PRODUCT BACKLOG, and each unit of development staff works from only a single PRODUCT BACKLOG.

This list typically corresponds to a single product or product line; however, it can be any list of deliverables.

Annotating each PBI with a cost, and a positional sensitive market value, can help the PRODUCT OWNER optimize Return of Investment (ROI). The PRODUCT BACKLOG can also be used to do a RELEASE PLAN and a schedule.

Pattern **PRODUCT OWNER**

... the Scrum team needs a single, well formed, sequenced list of product backlog items that allows them to focus on maximizing value delivery. Without it they can generate features that are never or rarely used, deliver features at the wrong time, which suboptimizes revenue, or become confused and disoriented and unable to deliver anything.

***

One person needs to be responsible for this backlog. This person needs deep domain knowledge, business insight to determine revenue impact of feature delivery, understanding of technical dependencies, and the authority to force rank the backlog to maximize business value.

Too many organizations try to put several bosses in charge of charting product rollout. Sometimes they can't even agree what the product is. While it takes up a staff position, and therefore a cost, to have someone explicitly lead the product development, it's almost impossible to make this work without a single focal point.

You can maximize revenue by continuously using your users as beta sites (where the deliverables are the beta deliveries) but customers can be made much more comfortable if a person on the vendor side takes responsibility for some up-front planning and thinking. Lean says that it is important to line up the decisions early on, so that the decisions are easy to make once the time comes to make them.

***

Therefore, **get a PRODUCT OWNER to order the PRODUCT BACKLOG and to take responsibility for the vision of the product**.

The best PRODUCT OWNER is as close to value delivery as possible. There are multiple stakeholders - the end user, the organization deriving revenue from end user use of the product, persons concerned with regulatory authorities, legal issues, or standards bodies, etc. Often stakeholders cannot be Product Owners because they have limited knowledge, a conflict of interest, or lack of authority to make decisions. (Picture - "The One" from the matrix). One of the Product Owner's primary key performance indicators (KPI) is revenue generation from the product - the business plan.

Each team needs one PRODUCT OWNER to deliver the Scrum TEAM one backlog. If there are multiple teams, there should be one PRODUCT BACKLOG for multiple teams to pull from.

It is often difficult to find a PRODUCT OWNER. The domain knowledge may reside in a business expert (sometimes the CEO) who does not have enough time to fully form the PRODUCT BACKLOG. The domain expert may become a CHIEF PRODUCT OWNER or you may need to find a SURROGATE PRODUCT OWNER.

If the creation of the PRODUCT BACKLOG takes too much time or expertise for one person, a PRODUCT OWNER TEAM needs to be formed with a CHIEF PRODUCT OWNER that has final authority over the sequencing of the product backlog.

Anyone can put anything on the PRODUCT BACKLOG. They own it and must provide support for fully forming the PRODUCT BACKLOG ITEM. The PRODUCT OWNER TEAM sequences that item in the global PRODUCT BACKLOG.

The PRODUCT OWNER is also responsible for the PRODUCT ROADMAP and the RELEASE PLAN.

Pattern **SCRUM TEAM**

… you have a project/product you want or need to get done with a definite goal.

***

Progress in a project needs to be achieved as best possible, and knowledge and risks about the work to be done need to be addressed as soon as possible.

You need to balance the level of responsibility of the team elements, micro-management (local plan vs global plan), terrain knowledge, prediction vs adaptation.

***

Therefore, create a cross-functional, self-organizing team, that gets facilitated by someone experienced (the SCRUMMASTER), and that has enough competencies to accomplish the project.

Once you have such a team (SCRUM TEAM), you are ready for a SPRINT and your next step is a SPRINT PLANNING MEETING.

Pattern **SCRUMMASTER**

… a ScrumTeam has been formed and they are task to self-organize to produce results

***

Sometimes the SCRUM TEAM fails to follow the Scrum framework and its patterns, but at the same time you need to promote team's autonomy and self-discipline vs. forced alignment and imposed discipline.

***

Therefore, select a person (SCRUMMASTER) that helps the team to enforce the Scrum patterns and rules.

As a consequence, the SCRUM TEAM now has some insurance that the Scrum patterns will be followed.

Variants:

* is the Scrum Master considered part of the Scrum Team (a role) or separate from the team (an individual)?

* We think that can be both, but probably one of them is better in some cases...

* under which conditions?

* which are the forces behind one option or the other?

Pattern **SPRINT**

Context: A Product Backlog has to be concluded.

Problem: What is the optimal amount of time that a team can work on?

Solution: Start a 30 day interation that comprises a SprintPlanningMeeting, the development work, including DailyScrums, the SprintReviewMeeting and SprintRetrospective. The 30 day iterations are a compromised in the amount of time people can work without feeling disconnected from the overall goal, in the amount of risk an organization can take, in the client or end user on seeing real progress. Sprint occur one after another one wihtout interruptions between them.

Resulting Context: after finishing a Sprint, the ScrumTeam is readyi for another Sprint, which may or may not be coincident with a release to shipable product to clients or to production for mission-critical systems.

Pattern **SPRINT BACKLOG**

Context: A well defined Product Backlog and a Scrum Team and a Scrum Master.

Problem: A specific way to break down the Product Backlog assigned into the Sprint into tasks that can be estimated, assigned, tracked, and reported.

Forces: size of the task, complexity, priorities, value.

Solution: Translate the selected portion of the Product Backlog into a Sprint Backlog by letting the developers choose what tasks they want to be responsible for, while making sure they are collectively capable of completing the work. Only some percentage of the Product Backlog is translated into tasks at the SpringPlanningMeeting, because some of the tasks will have to be defined as the ScrumTeam learns or knows more. The Sprint Backlog is updated during the Sprint as new tasks are discovered, refactored, deleted, and if there are any tasks left at the end of the Sprint, it is preferable that they all belong to the same Product Backlog item. That way these tasks can be transferred to another Sprint.

Resulting Context:The ScrumTeam has an instrument to measure its progress (see problem).

Pattern **SPRINT PLANNING MEETING**

Context: A ProductBacklog, a ProductOwner, a ScrumTeam, and a ScrumMaster have been selected

Problem:What work should be selected by a ScrumTeam to work on a Sprint, and how is this work to be accompliashed or... How does the team create and maintain the SprintBacklog? or

Forces:

Solution:For a 30 day Sprint, have an 8 hr meeting where the ProductOwner presents the prioritized ProductBacklog to the team and discusss the ProductBacklog for about 1 hrs. In the next 2 hrs the ScrumTeam then selects the prioritizede the work to be done as it negotiates the work to be done with the ProductOwner. The ProductOwner articulates the SprintGoal, which is the theme or the summary of the purpose of the Sprint. Members of the ScrumTeam then volunteer for certain ProductBacklog items, and then they proceed to translate the ProductBacklog items into specific tasks that they own now in the SprintBacklog. It is not expected that they translate all the ProductBacklog items or that these estimates are always correct. It is just a starting point that can be evolved at the DailyScrum meetings.

Resulting Context: An initial SprintBacklog.

Pattern **SPRINT BURNDOWN**

Context: the ScrumTeam has reported its status through the DailyScrum

Problem:What is the best way to provide the overall status for the team?

Forces: activity vs backlog

Solution:Provide a SprintBurndown as overall status of the ScrumTeam in the Sprint. A SprintBurndown is a graph of remaining estimated time to completion for the whole team vs time. This graph provides the status by saying "this is the time we need as a whole to complete what we agreed upon in this Sprint". Other units other than time can be used such as story points, chocolate points, or whatever else the ScrumTeam has agreed upon.

Resulting Context:The ScrumTeam knows where it stands in regards to accomplishing the Sprint

Pattern **SPRINT REVIEW**

Context:The ScrumTeam has used all the time for the Sprint

Problem:What is the best way to evaluate the status of the Sprint and move on to another Sprint

Forces:

Solution:Schedule a SprintReviewMeeting at the end of each Sprint. The SprintReviewMeeting is a 4 hr meeting for 30 day Sprints that brings the ScrumTeam, the ProductOwner, the stakeholders. First the ProductOwner identifies what was done and what wasn't. The ScrumTeam then discusses what went well during the Sprint and how they solved problems they encountered. The team then proceeds to give a demo of the functionality developed, and then asnswers questions about it. The ProductOwner then discusses where the ProductBacklog stands and makes some projections of where the team might be in the future making some velocity assumptions. Finally the team collaborates as it reflects on their experience and what the ScrumTeam might do next.

Resulting Context: the ScrumTeam is ready for another Sprint

Pattern **SPRINT RETROSPECTIVE**

Context:The ScrumTeam has just finished the SprintReviewMeeting, but has not started the next SprintPlanningMeeting.

Problem:What is the best way for the team to reflect on how they work and improve?

Forces:While the team is always at liberty to self-correct, it may not have the time to concentrate on how to improve on a daily basis, because it should be focused on completing the work for the Sprint. on the other hand, it may prove to be wasteful if the team doesn't take the time to improve after many Sprints, because there might be issues that could drag its performance or the comfort of the team members down.

Solution:After the SprintReviewMeeting but before the next SprintPlanningMeeting, schedule a 3 hr meeting call the SprintRetrospective for the team to discuss its development process and improve it, both to make it more effective and enjoyable. The Scrum team analyzes at least 4 areas: people, process, realtionships and tools. The inspection should both list the things that worked well, and the things that could be improved. So for example, team composition, meeting arrangements, communications internal and external to the team, ScrumMaster efficiency, tools, definition of "done", etc.

Resulting Context: the ScrumTeam has a specific list of improvements for the next Sprint

Pattern **DAILY SCRUM**

Context:The ScrumTeam is either starting or working withing a Sprint.

Problem:What is the best way to keep track of the status of the outstanding work within a Sprint?

Forces:

- too much/too little time between,
- too much/too little detail about the status.
- too much/too little horizon for planned tasks

Solution:Have a short inspect-adapt meeting, that lasts about 15 min, every day, to answer the following 3 questions for each of the participants:

1) what did I accomplish since the last Daily Scrum?

2) what are the impiediments to make progress

3) what will I be doing until the next DailyScrum?

Only team members can talk during these meetings, and they can only answer the questions above, or make plans for follow-up meetings to collaborate with one another. The meeting should take place at the same plance and at the same time. The meeting improves communication, provides project knowledge to everyone, promotes collaborations among team members, and supports quick decision making. It also replaces many other longer meetings and therefore avoids waste.

Resulting Context: the team members are ready to realize the work until the next DailyScrum

Related Patterns

As previously mentioned there are many patterns very important for Scrum that must be linked to the Essential Scrum Patterns. Here is a list of some of these related patterns to be linked to:

- possibly other Essential Scrum patterns such as **Product Backlog** with **Sprint Planning Meeting** and **Sprint Review Meeting**;
- detailed strategies patterns within a Scrum area, such as **Product Backlog Pattern Language**;
- other typical Scrum patterns, such as **Visible Status**;
- several patterns from the Organizational Patterns book, such as **EngageQA**;
- advanced Scrum patterns that allow a much faster Scrum implementation, such as **Bull Pen**;
- enterprise Scrum patterns that allow Scrum to be implemented in the enterprise, such as **Scrum of Scrums**, **Global Backlog;**
- other available organizational patterns that make sense within the Scrum framework, such as **Stable Teams**;
- engineering patterns, such as **Continuous Integration**;
- patterns for distributed **Scrum Teams**, such as **Constant Communication**.

Acknowledgements

The authors would like to thank all participants of ScrumPLoP 2010, from where the idea of documenting the **Essential Scrum Patterns** started, and to thank them also for their valuable contributions during the respective writers' workshop: Gertrud Bjornvig, Mette Jaquet, Espen Suenson, Gabrielle Benefield, Lachlan Heasman, Jeff Sutherland, Neil Harrison, Dina Friis, Jens Østergaard, and James Coplien.

References

[Beedle et al] Mike Beedle, Martine Devos, Yonat Sharon, Ken Schwaber, and Jeff Sutherland. "Scrum: An extension pattern language for hyperproductive software development", PLoP'1998.

[Coplien95] J. Coplien and D. Schmidt, Pattern Languages of Program Design (A Generative Development-Process Pattern Language), Addison and Wesley, Reading, 1995.

[Nonaka95] I. Nonaka and H. Takeuchi, The Knowledge Creating Company,Oxford University Press, New York, 1995.

Org Patterns – Borland Quattro Team, First Org Patterns paper

Scrum Guide – Ken Schwaber

Scrum Primer,

Nonaka and Takeuchi, The New, New Product Development Game

Etc.