

Agile Project Management in a Regulated Environment

Alistair Dootson, d-Wise, Manchester, UK

ABSTRACT

Scrum is an agile approach to project management for software development or implementation, which if implemented correctly is able to satisfy regulatory requirements effectively and efficiently with high productivity and high quality.

The classical waterfall approach has dominated the regulated industry as it meets the regulation requirement on Quality Assurance, Safety, Security, Effectiveness, Traceability, Verification and Validation, but this has resulted in challenges, such as vast amounts of unnecessary paperwork, error prone manual processes, lower productivity, slow response time to market, slow to implement change and adopt industry standards.

In this paper we outline how “Scrum as an agile” approach can be used to address some of the key challenges we come across in the regulated environment. We will also discuss the key components of Scrum – values, roles, activities and artefacts – and how they are used during the process.

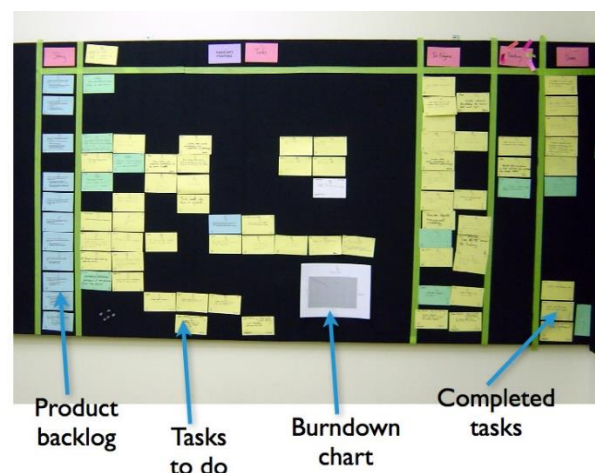
INTRODUCTION

This paper examines the use of an agile approach in a regulated environment and in particular a project that would not normally conform to the classical approach of a scrum based project. We will look at Scrum and its components, then take a look at the project in question before drawing our conclusions.

INTRODUCTION TO SCRUM



This is not the one.



This is the concept we are looking at

As with any project management methodology, there are components specific to Scrum in name, but similar to other methodologies in type. We will now take a look at some of the features of scrum.

Feature requests for the project are defined and collected from the perspective of the user and are called **User stories**. These are laid out in an easy to follow way – As a (role), I want (feature), so that (benefit).

The User Stories are collected in a **Product Backlog**, which is essentially a wish list of all stories (requirements) that are wanted for the system.

There are several roles that are required in a scrum project. **Product Owner** – this person has overall say of which features are implemented for the system and gives direction to the project. The **Scrum master** is what is classically known as the project manager. They ensure the project is running smoothly and clears any obstacles in the team's way. Then, there is the rest of the team members made up of developers and testers.

Releases are a subset of user stories from the product backlog. A number of stories are selected, broken down if necessary, prioritised and the time to complete each part is estimated. Estimating the time required to complete each story is never an easy task, but it should be broken down in a way to keep tasks small. It is preferential to have only a few standard estimate times. 1, 2, 4, or 8 hours if a task is less than 1 day. With days of effort; 1, 2, 5, or 10 days. Anything larger should be broken down to a much smaller level.

Once the estimates have been completed, several **Sprints** will be defined. Each sprint is of short duration to complete a milestone, for example 2 weeks. This allows the team to tackle a smaller, more manageable piece of the project and allow it to be completed modularly.

Sprints can be managed using a **burndown chart** – a day by day measure of how much work is remaining for each sprint. This is reliant on each of the team members being responsible for updating the time remaining on their tasks and the scrum master can manage the overall burndown chart.

A **daily scrum meeting** is held every day to discuss the achievements of the prior day, if any obstacles are preventing progress from being made and what the plans are for the next day.

At the end of each sprint, the team holds a **review meeting** to tie up any loose ends for the tasks in that sprint; and a **retrospective meeting** to discuss what went well, and what could have been done better.

THE PROJECT

Scrum projects usually focus on building something, whether it be a product or code. In this example we look to see how scrum can be used in a more typical industry problem, a software migration from an existing version of an off-the-shelf software product to the next.

The focus of the paper is not on the project itself, but on the project management around it. However it is important to understand the general make-up of the project.

We will be migrating SAS Drug Development version 3.5 to version 4.5. A change in which the configuration and data migration is substantial.

APPROACH

Initially, only a single high level user requirement was defined – the need to migrate to the new version of SDD. A draft list of tasks required to achieve this was defined and used as the basis of the functionality requirements. At the high level, it is possible to define releases – smaller chunks of the overall desired result – and to work on those in sprints. So, the project was broken down into the setup and configuration of the new development environment, the setup and configuration of the production environment, workflow design and implementation, integration of statistical computing environment, other third party systems, and data migration and configuration.

Each of those high level releases are broken down into manageable tasks and arranged into sprints.

	Category	User Story	Deliverable	Sprint	Priority Rank	Task Status
41	Environment Configuration - DEV	Develop Script using SDD API to build Global folder structure for DEV. This will be used later to be run in PROD	Global Hierachy Folder Structure building script (GFSB_SCRIPT)	3	0	
42	Environment Configuration - DEV	Run GFSB_SCRIPT to create a Global folder structure in the DEV environment	Screen Shot of the Global folder structure, Log and output from the Script	0	0	Not Started
44	Environment Configuration - DEV	Develop Script using SDD API to build the standard folder structure for new Trial. This will be used later to be run in PROD	Standard Folder Structure building script (SFSB_SCRIPT)	3	0	

The sprints last for a short period of time, only 2 or 3 weeks, so a sprint planning meeting takes place to define and prioritize the tasks to be done in that sprint. With scrum, it is important that there is clear and open discussion, so it is opened to the team to decide if the tasks in the sprint are achievable within the time frame allocated.

TOOLS

There are several scrum tools available for managing agile projects. They range from the simple EXCEL spreadsheet, to a MS Project plug-in, to specific Scrum tools.

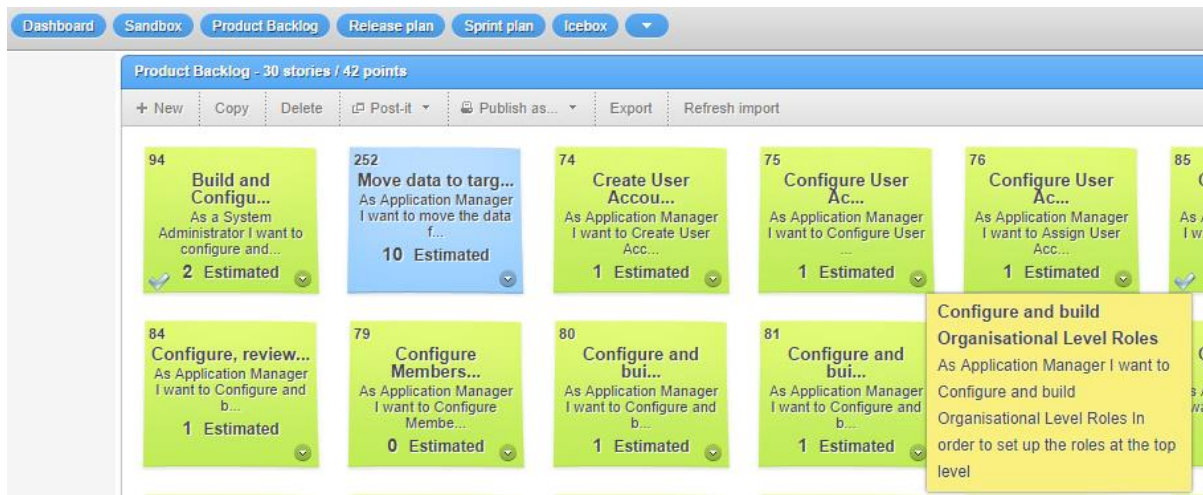
The simple EXCEL spreadsheet works well to track tasks, due dates, responsible parties etc., but does not really guide the user down the path of scrum.

No.	Date of agreement	Responsible organization	Topic	Sprint no.	Task / User story	Deliverable	Responsible person	Category
158	2014-Jul-17		Sprint	3	Develop Script using SDD API to build Global folder structure for DEV. This will be used later to be run in PROD	Global Hierachy Folder Structure building script (GFSB_SCRIPT)		Environment Configur
159.5	2014-Aug-11		Sprint	4	Move GFSB_SCRIPT to create a Global folder structure in the DEV environment	Confirmation of move		Environment Configur
159.5	2014-Jul-17		Sprint	3	Run GFSB_SCRIPT to create a Global folder structure in the DEV environment d-Wise DEV	Screen Shot of the Global folder structure, Log and output from the Script		Environment Configur

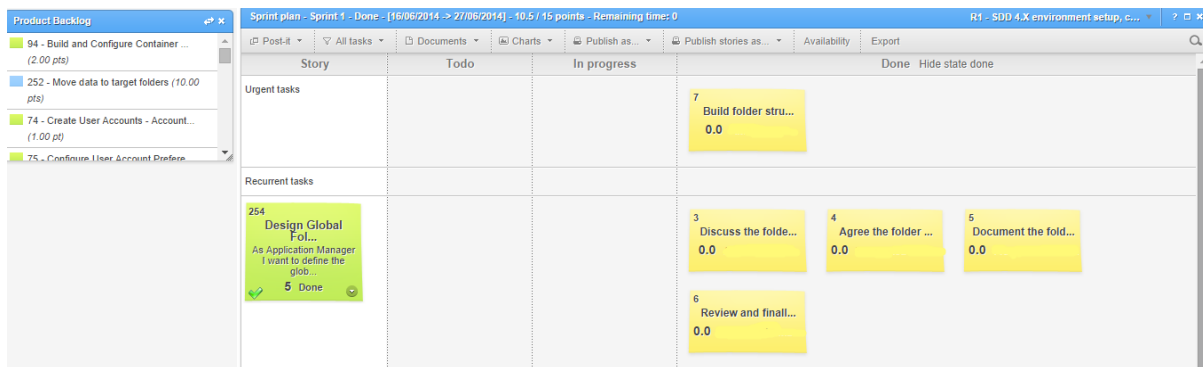
A specific scrum tool is designed so that the artefacts of scrum project management can be followed easily, but does require full participation of the entire team, plus a lot of effort to set it up and follow it completely.

The access to the system is restricted and the users are allocated roles such as product owner, scrum master or team members. They each have specific jobs to perform in the system.

The tool will prompt the user to enter user stories, promote them to the product backlog and encourage tasks to be created and estimated for the user stories. One thing we haven't mentioned so far is the definition of done. This is also defined here, to show what constitutes when a user story, a sprint or a release is considered complete. Acceptance criteria can also be defined.



Once all the user stories are entered into a sandbox and promoted to the product backlog, they are broken down into sprints and moved through the process by the users. This is where the team members must go through their tasks that are to-do, move them to the in progress area when they start to work on them and mark them complete when done.

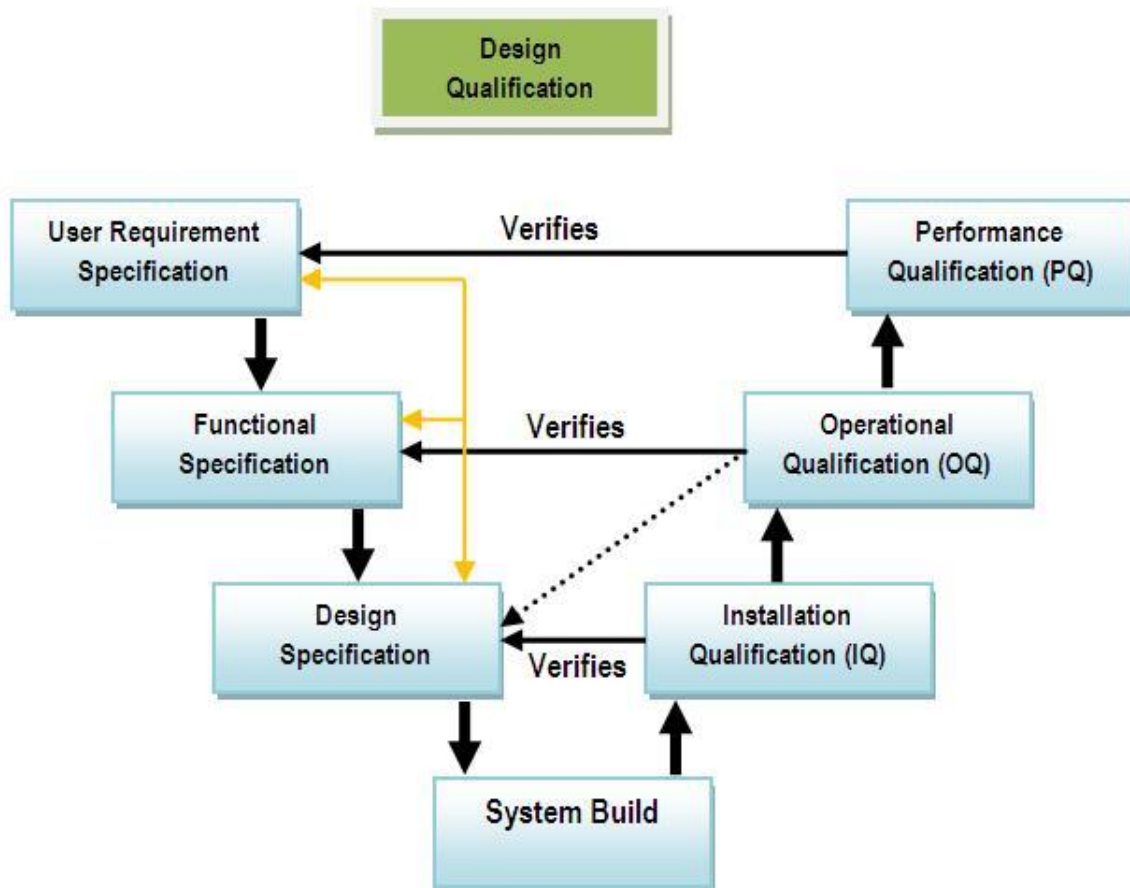


In the case of the specific scrum tools, they are excellent for following the scrum methodology from beginning to end. But does it work in a classic project in our industry?

VALIDATION

How does the requirement for companies to conform to the classical V model for validation work with an agile approach to project management?

It would normally be required that a full list of agreed requirements be agreed and signed off at the start of the project and be tested in the User Acceptance Testing – or Performance Qualification section.



With an agile approach, the requirements can be added throughout the project. The V-Model still applies to the validation of the project, but it may just be that there are several V-Model sections throughout the project for the releases as they happen, with an overall validation plan explaining the approach. So, for each of the deliverable releases, the reports will reference a subset of requirements (or user stories) and the relevant testing of those requirements.

The agile approach offers excellent flexibility to sign off each portion when it is complete, rather than running a complete validation package at the end of the project. As long as the validation plan explains the approach, and the validation report can tie up all the testing and requirements, the approach should be acceptable. This also spreads out the burden of validation across the whole project rather than all at the end.

CONCLUSION

Does the agile approach really work in our industry? I think the answer is partially. It is wise to use an agile-like approach and pick the best parts and work with them.

THE BEST AND WORST OF SCRUM

The best features of scrum include the flexibility of requirements, the broken-down, smaller chunks of work and daily check in meetings to keep on top of the work. Feel like you are achieving more along the journey of the project, rather than seeing only results at the end. Validation can be made simpler by explaining the approach in a validation plan and either keeping documents live until the end of the project or completing smaller pieces of validation along the way.

The burn-down chart is really useful to keep track on how the project is performing and if it is on target, but it is reliant on accurate estimating, constant tracking of time spent on tasks which adds overhead.

The worst features include the overhead of maintaining any tools and tasks, plus the initial set up. Defining acceptance criteria, definitions of done, tasks, roles and moving the user stories through the process can be time consuming. Team members that have to spend more time understanding how to enter and update their time and tasks in a specific tool than on their tasks is also a waste of valuable project time.

Agile and scrum success is dependent on a multi-functional team that is self-disciplined, honest and knowledgeable about activities related to the project. It requires a team effort and not just one team member with

responsibility to drive the project. The tools are good to push the team in a true agile direction, but there must first be full buy in from the team to learn and understand the tools. This must be included in the timing for the project.

There must also be an overall framework to the project. The releases must be thought out early on in the planning stage because as flexible as agile is, it still requires overall control and aims otherwise it could just keep going on sprint after sprint.

In conclusion, from a non-project manager position, I feel that an agile – or scrum – based methodology for running projects in a regulated industry offers a lot of flexibility around requirements, can help with validation lifecycles and is an excellent way of managing projects.

REFERENCES

V-Model diagram taken from http://pharmaceuticalvalidation.blogspot.co.uk/2011_01_01_archive.html, accessed 15 September 2014

ACKNOWLEDGMENTS

Thanks to Amol Waykar, d-Wise for his contribution to the paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Ali Dootson

d-Wise

Suite 2, 3rd Floor, 61 Mosley Street, Manchester, M2 3HZ

+44 161 236 0961

ali.dootson@d-wise.com

www.d-wise.com

Brand and product names are trademarks of their respective companies.