Requirements Definition Sprints:

An Agile Case Study

By Richard "Dick" Carlson

Introduction

This case study occurred in 2007 and is based on a situation that happens often on projects when project planning is inadequate. The situation is also based on a large enterprise-level program at a leading aircraft manufacturing company where the modification of a large and complex commercial-on-the-shelf (COTS) product was acquired that needed to include the company's engineering and product data management process. The purpose of the new COTS product was to replace all existing and obsolete engineering and product data management systems. The project was expected to continue for the next several years until all essential functionality was modified to meet company compliance and support needs.

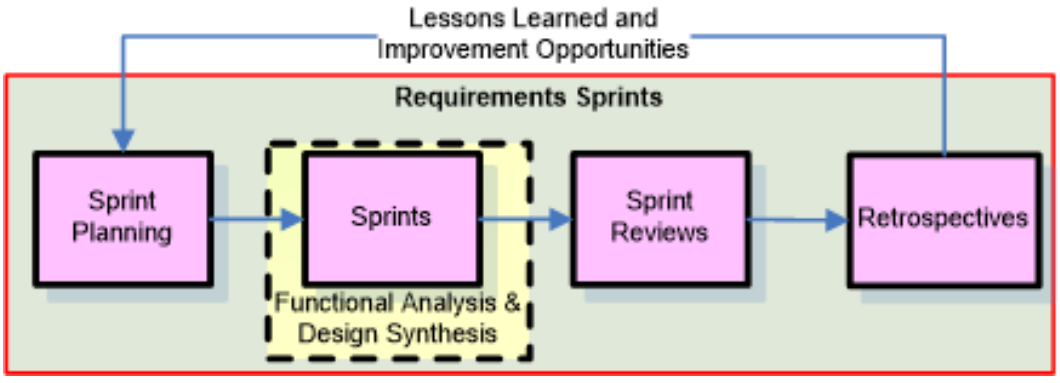
Background

As a leading Agile trainer and coach for the company, I was approached by program management to assess current program progress and recommend operational improvements in project execution. After listening to and assessing the dilemma explained by management, my recommendation was to apply an approach that could quickly improve project management techniques using the Agile project management approach, Scrum.

Scrum is the most popular and effective method used in Agile projects to manage difficult and large projects. Teams that use Scrum consist of motivated and self-organized members who are domain and technical experts skilled at requirements analysis (or the development of requirements), design, code, and test. Since Scrum was used to manage Agile software projects, program management demanded rapid feature modifications to the application.

The Process

Most Scrum teams were staffed with functional and domain analysts, one or two system engineers, an architect, one or more software developers, and a tester. Although the team members had specific skills, they were screened prior to team assignment to ensure that they were familiar with other requisite team skills so they could work the tasks of an absent team member. This provided the team with the required cross-functionality needed for their execution effectiveness. Each sprint was executed in the manner shown in the following diagram.



User stories were written by the team directly within the standard requirements management tool that included links to their source documents. User stories were written at the epic level, supporting epic themes contained related but smaller user stories. Every user story included one or more success and failure conditions to define acceptance test cases.

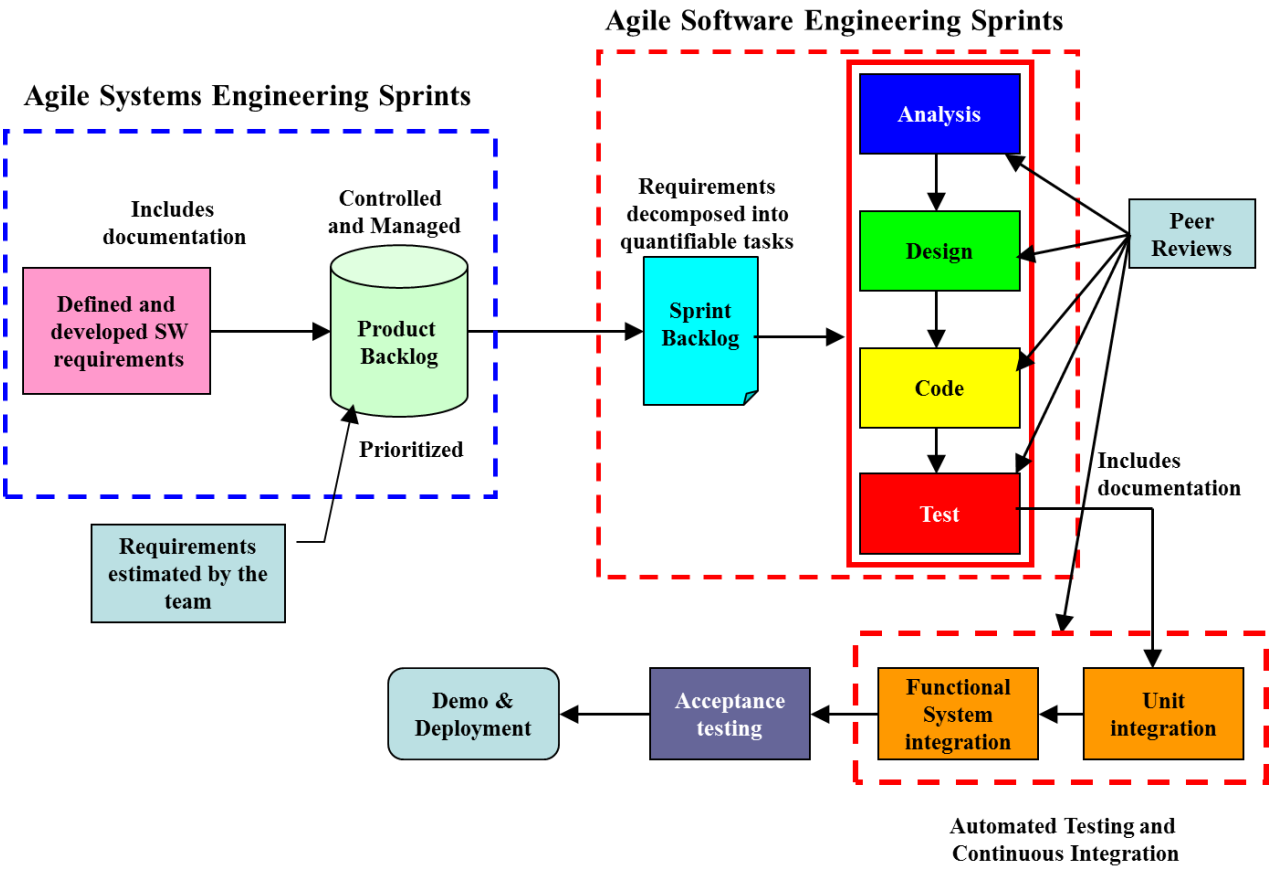
All sprints where 4 weeks long that began with a planning session and ended with a sprint review that included the team reviewing newly-written functionality with the Product Owner and relevant stakeholders. The results of each review included the following:

- Team members were able to define and develop functionality requirements in user story during each sprint format in a highly collaborative environment with domain experts
- Team members were able to write dozens and sometimes even hundreds of user stories based on business and user needs during each sprint
- Bi-lateral traceability was established in the requirements management tool during each sprint by linking each epic, theme, and smaller user story to design objects and test plans from original requirements sources (i.e. documents, tables, databases, etc.).
- Some refactoring was performed on user stories written previously during early discovery activities, and some user stories were either combined with similar stories, and some stories were deleted when it was determined they were no longer required or beyond user expectations

Requirements were still evolving and some requirements were not yet known, so a decision was made to establish initial sprint teams to develop high-priority functionality changes in user stories format and create product backlogs. The multiple product backlogs were necessary as the program would execute across multiple engineering domains, which also meant that each domain required unique Teams, Product Owners, and Scrum Masters to define and modify application functionality and to meet the company's operational standards. Initially, all requirements were written into Excel spreadsheets, but were later moved into the company's standard for managing requirements.

To further complicate things on this program, requirements were not stable. That is, many of the needs of the system had to be defined by its users, who were ill-defined. Typical system users include aircraft designers, engine designers, aircraft interior designers, electronic and electrical engineers, stress engineers, fluid mechanical engineers, mathematicians, control engineers, propulsion engineers, structure engineers, mechanical engineers, statistics and dynamics specialists, structural engineers, acoustic engineers, flight and test personnel, pilots, risk and reliability engineers, safety engineers, avionics, material science specialists, software engineers, systems engineers, and many others.

The instability of requirements at program start-up meant that software teams could not build software products, because the functionality modifications were yet unknown. So, I convinced management to delegate at least two Scrum teams to define and develop requirements first. Once the teams had developed a sufficient number of user stories and assemble requirements at an appropriate level of clarity, we could schedule software development teams to develop and make the software modifications to the COTS application. The experience model below shows how the two teams were structured to (a) define and develop requirements, and (b) modify functionality to the COTS software.



Since most system requirements were not well-defined or developed at all, the decision to conduct requirements sprints helped to establish accurate and robust product roadmaps and backlogs. Many of these requirements could have taken months to define and develop using a more traditional approach, but using Scrum's 30-day sprint structure proved to be most effective and productive solution.

Summary

The program was able to produce high quality software from undocumented and incomplete requirements. The velocity of the team was quite high, and vigorous refactoring resulted in the development of stable software. The project has been funded through 2016 and continues to use Scrum as its preferred project management approach. The level of system functionality modifications and team productivity has been maintained.

About the Author

Dick Carlson has been an active Agile transformational leader for many small and large projects, and has frequently shared his experiences of successful Agile, Lean, and Scrum implementations at conferences, workshops, and symposia, and regularly advises executives and organizational leaders on the cost, quality, and schedule benefits of using those initiatives and techniques. He has actively coached teams for more than 20 years on Agile and Lean Project Management fundamentals, and follows up with mentoring activities to ensure successful project execution. Dick has also provided concentrated Agile coaching support and led many organizations, programs, and projects that varied in size from six to more than 2,000 engineers, and that ranged in costs from under $50,000 to more than $1.4B.

Dick used Scrum practices and principles to manage and form the start-up of the Agile & Lean Education Associates (ALEA) Company that began July 2013. He continues to share his practical knowledge of Agile, Lean, and Scrum through more than two decades of experience by means of Agile and Lean training and the right amount of coaching to companies and other activities that want to increase their competitive advantage. The ALEA Company website (http://www.a2zalea.com) provides information about who we are and what we do.

