Agile Requirements Management
Why, When and How

Nils Pålsson – Senior Requirements Management Consultant
Introduction

- The advantages of applying lean and agile techniques to software design and development activities of are now well established and understood.

- But many organisations who have implemented agile techniques continue to struggle because they either do business analysis activities through a waterfall process or skip upfront requirements analysis completely as proposed by agile purists.

- How do we use requirements definition and management in an agile process?
Agenda

- Why?
- When?
- How?
The need for Requirements Management

I distinctly said monorails!

Did not!

B.C. by Johnny Hart
What is Requirements Management (RM)?

“The purpose of requirements management is to establish a common understanding between the customer and the ... project ... This agreement with the customer is the basis for planning and managing the ... project.”

The Capability Maturity Model for Software (CMM®) from the Software Engineering Institute at Carnegie Mellon University.

www.sei.cmu.edu/cmm
Informal terminology notes

- The term “Requirement” is loosely used to including all requirements-related information
  - Textual requirement
  - Use Case
  - Usage Scenario
  - Feature Descriptions
  - Diagrams and Sketches
  - User Story
  - Etc

- These are not all strictly requirements, the same points apply to all
Why RM is seen as a burden

- Requirements Management is often seen as completely against the principles of agile development
- Equated to “write loads of big documents that nobody will ever read”
- Nothing in place to use the requirements
- Requirements take a lot of time to create, and then just sits on a shelf
Why RM is (still) important in an Agile way of working

There are some items normally not found on the backlog. i.e.

- Feature descriptions
- Non-functional requirements
- Process diagrams
- Notes from Stakeholder meetings
- Architectural decisions
Why RM is (still) important in an Agile way of working

How do we know what we have built?

The development team won’t be there for ever. Someone has to maintain and extend the system. A list of user stories does not give a sufficient picture.

Requirements can be viewed as:

Team’s memory
Team members can find information about work done in previous sprints without having to dig through stacks of “done” user stories

Product owner/business analysts memory
To inform the creation of new user stories

User documentation and training material development
Why RM is (still) important in an Agile way of working

- Because solving the wrong problem well or fast does not make you successful
Why Requirements Management is (still) important in an Agile Way of Working

When?

Collaboration at the speed of Agile
A look in the rear view mirror
Issues with the Big Requirements Up Front Approach

- **Requirements Change**
  Months or years between finalizing requirements and deploying the system. The goal was usually to minimize or prevent requirements change to stay on budget and schedule.

- **The understanding of the requirements change**
  Often a changed requirement is really just an improved understanding of the actual requirement.
A look in the rear view mirror (cont.)

Issues with the Big Requirements Up Front Approach

- **Made up requirements**
  Historically stakeholders were taught that defining requirements early is painless, but painful later, resulting in a large percentage of functionality that actually wasn’t needed.

*Admit that those thick requirements specs were never complete or correct!*
When should you write Requirements?

- Requirements are only written *when needed* and *detailed enough* to know how to implement the system.
- Requirements are written *just in time*, to help understand and decompose items on the backlog.
- Capture decisions as they are made.

Documenting your system as it is being built enables you to better *reuse* work when developing the next feature/component/system, *saving both time and money.*

If you are fundamentally opposed to calling such decisions “requirements” then don’t. *But still capture them!*
How much Requirements Analysis?

- Agile purists who argue ‘do none or at the most don’t do much because the requirements will change’
  - “Rather than coming up with a bunch of features and planning a multi-month release, come up with new ideas continually and try them out individually on users.”

- Traditionalists who want to do as much as possible, because we need to know we are doing the right thing before investing
  - “For the second consecutive year, IAG found poor requirements definition and management consume over one-third of IT's application development budget.”

The Agile Requirements Lifecycle

Project Launch – Initial Requirements Envisioning
- Establish product vision and scope
- Elicit and analyze initial business requirements
- Get stakeholder agreement on initial business requirements

Before each iteration
- Get stakeholder agreement on acceptance criteria for high priority requirements. Suitable during Sprint Demos
- Get team estimates on high priority requirements
- Do further decomposition only if requirements are too large

During the iteration
- Identify and resolve requirements conflicts
- Identify and fill requirement gaps
- Analyze requirements for standards compliance
- Support the development team
The Agile Requirements Lifecycle

Scrum Alliance
Agile Requirements Definition and Management
Feb 2012
Use of the Requirement Model

- **Requirements Analyst**
  - Ensure consistency of new requirements
  - Identify existing areas that need change
  - Do impact assessment of needed changes

- **Product Owner**
  - Identify incomplete areas of the product
  - Report on completeness of the product

- **Team**
  - Get further information on what to implement
  - Recall details of how finished parts of the system work
Why Requirements Management is (still) important in an Agile Way of Working

The Agile Requirements Lifecycle

How?
Analyze the problem

- Users and customers don’t want systems – they want their problems solved
- Solving the wrong problem well or fast doesn’t help
- The problem as first stated is rarely the true problem
- Understand the problem
  - Determine the purpose
  - Look for root causes
- Gain agreement and document the problem as appropriate

Understand the problem your solution will solve
Eliciting Requirements

- Getting requirements requires people skills
- Often users don’t know what they want
- Sometime users know what they want but can’t express it
- You need skills and techniques for getting good requirements
Communication and Collaboration

- Organize the team(s) to encourage communication and collaboration
- Using an environment that encourages communication and collaboration.
- Whiteboard for small co-located teams and non-complex systems
- For complex projects and globally distributed teams, a requirements management tool with emphasis on
  - Collaboration using reviews and threaded discussions
  - Requirement change impact assessments, and gap analysis
  - Graphical requirements artifacts, a picture says more than...
  - Rich elaborations of stories and epics using diagrams, pictures and tables
  - Automatic metrics
A typical team structure

Requirements analyst team members can be:

- Embedded in a feature team, following the features all the way through development into test, answering questions and clarifying the requirements along the way.
- Focusing on the "big picture" and aren't embedded in any feature team. They look further into the future to define high level feature areas.
- Flexible and move between the above two roles depending on where the greatest need is at the moment.

Lean from the Trenches,
An example of Kanban in a Large software project
Henrik Kniberg, 2011
Shorter feedback loops using comments and reviews
Graphical Artifacts
Use Visual Scenarios to Uncover Customer Needs

- Defining requirement flows using scenarios to uncover missing critical details
- Text requirements link to diagrams to complete the development picture
- Visualise your development results through a variety of requirements forms
- Traceable elements help ensure complete coverage thinking
Traceability across the lifecycle
IBM Rational Requirements Composer 4.0.4
Requirements Management for the Development Lifecycle

**Definition**
- Rich-text documents
- Diagrams: Process, Use Case
- Storyboards, UI sketching & flow
- Project glossaries
- Templates (formal/agile)

**Visibility**
- Customizable dashboards
- Project dashboards
- Analysis views
- Collections
- Milestone tracking & status

**Collaboration**
- Review & Approval
- Discussions
- Email Notification

**Management**
- Structure, Attributes/Types
- Traceability, Suspect Link
- Filtering, Change History
- Tags, Reuse, Baselines,
- Reporting Metrics & Doc.

**Lifecycle**
- Central requirements, test, & development repository
- WAS Clustered Server
- Common admin and role-based user licensing
- Warehouse reporting

**Planning**
- Integrated planning
- Effort estimation
- Task management

Supports RequisitePro Data Migration
How to do Requirements Management in an agile world

Benefits must outweigh the costs.

- Benefits mostly come from
  - Structure. You know where to find information, and where to put it
  - Information transparency. Delivered through traceability

- Costs
  - You were going to create the information anyway, so put it in the right place, not just in a bucket
  - Cost of creating traceability at the time you create information is minimal.
    - You know what it relates to.
    - The information is all visible to you
“Agile is quickly becoming the most popular way of developing software because it ... more quickly deliver value to the end users. That value will be driven to a large extent by the quality and clarity of requirements that feed the software development process. An agile, lean, and timely approach to requirements as the starting point will help to ensure that the process is optimized.”

Scrum Alliance
Agile Requirements Definition and Management
Feb 2012
We’re Agile now, do we really need Requirements Management?

- Yes, if done correctly, at the right time, and to the right level!
- Yes, everything isn’t captured on the backlog!
- Yes, we need to know what we have built!
- Managing requirements does not need to be a burden
- Requirements management is quite possible and necessary in agile development
THANK YOU
Partners
Demo Solution Center

- Test
- DevOps
- FocalPoint
- Integrations
- Reporting
- Modeling
- Requirements
- Agile Planning
- JazzHub