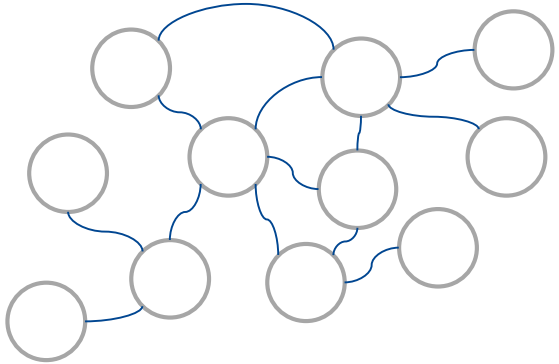
Challenges of agile methods in the system development and unexpectedly finding answers from the V-model

Miia Onkalo

System Under Development

- Part of a complex system of systems
- Use cases are complex and unpredictable
- End users are specialized to do their work
- New system replaces the old system



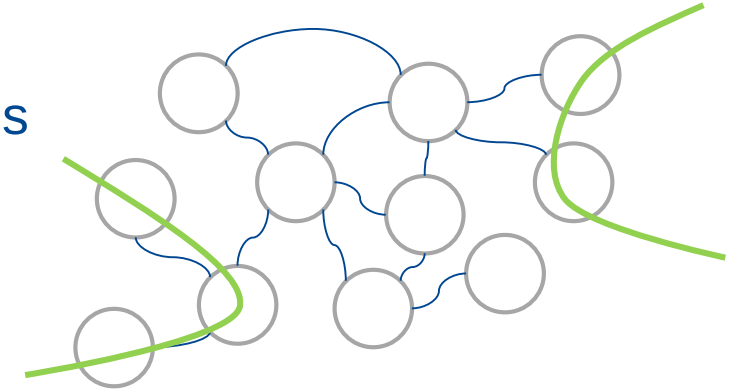
Changing Operating Model

- Currently different ways of working based on geographical location
- With new system operating model is evolving and harmonizing ways of working
- Both the system and the operating model should be validated against each other
- How to distinguish the current operating model and crucial parts of the daily work?

Learnings from User Stories

- Attention was in those parts that were easy to see and understand
- Assumed that the end user scenarios are simple and ordered
- There is a space between and behind the user stories
- Small user stories directed focus to verification rather than validation

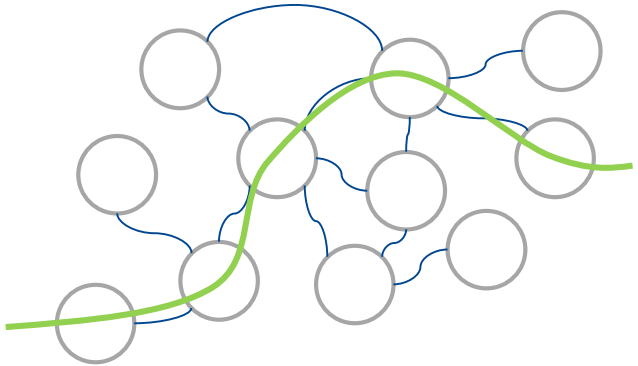
Adapt → we need to do more specifications



Learning from Specifications

- To fill in spaces between and behind the user stories, more extensive specifications were written
- Got good specifications but features didn’t fit for intended use
- Interactions between features were not always understood
- Still guides to verification instead of validation in the system level

Adapt → we need understand end users daily work better



Learnings from the Information Mining

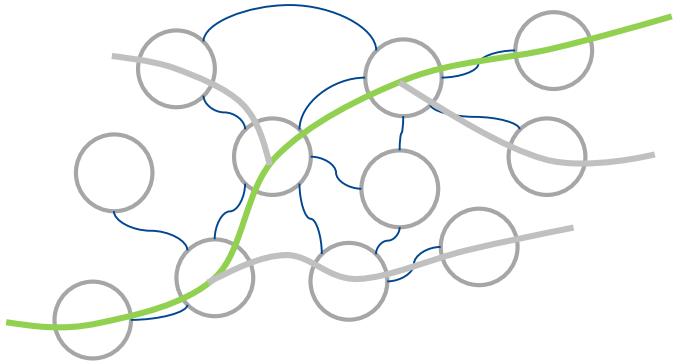
- Developers are not subject area experts of the customer’s domain
- Developers aren’t familiar with methods to collect information from customers and end users
- System of systems have a lot of information and information is unintentionally filtered out all the time

Adapt → observe end users in their daily work, engage user experience folks

Demos

- Developers are demoing, stakeholders are watching
- A large part of the implementation is quite technical and not interesting for most of the stakeholders
- Does the story or features fit for the purpose? We don’t know as demoed content don’t form a complete end-to-end scenario

Adapt → no good ideas



Situation We Found Ourselves

- Project went on, we inspect and adapt but it was never quite there
- We kept redoing already done functionality
- Root cause analysis said often:
  - “we didn’t know it is used like that”
  - “that was not specified at all”
- Capturing the intended use of the system in the context of evolved operating model was the key problem

In the meanwhile we need to do system testing…but based on what?

System Testing That Made a Difference

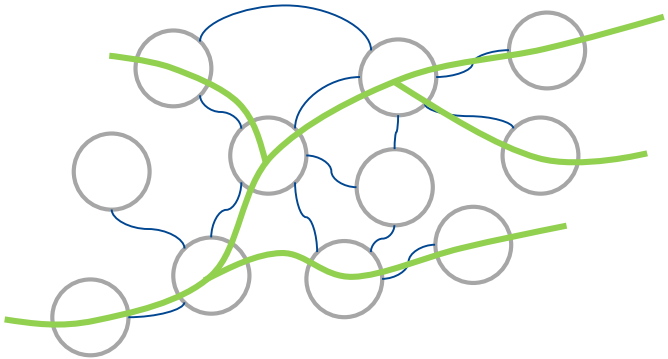
- System testing was considered as a must, however we didn’t have testable requirements
- System testing approached missing requirements from the user acceptance testing and validation point of view
- Use cases were defined based on the real operative end-to-end scenarios
- Testing was executed by real end users simulating their daily job – parallel, end-to-end, with surprises

Adapt → System testing as a **simulation**

Simulation - Where It All Comes Together

- Focus was in the end-to-end system testing
- Agile teams got first hand knowledge of the subject area
- UX folks got a lot of information for the future development
- Real opportunity to validate if the system fits for the purpose
- Testing of new operating models was enabled

Adapt → benefits of hands-on simulation exceed demos, include end users and agile teams



Key Learnings

- Knowledge will filter out and disappear in long projects
  → Right amount of documentation
- Systems are complex
  → Don’t downplay the complexity for the sake of Agile
- Subject area expertize can be difficult to capture right
  → Hands-on engagement and observe real end users
- Inspect & Adapt is the key
  → You can’t plan your optimal way of working in the beginning

Questions?