User Centered Design and Agile methods: case Nokia Siemens Networks

Harri Klemetti
Senior Specialist, SW Usability

“World Usability Day” -seminar at Tampere University of Technology
8.11.2007

Table of contents

• About Nokia Siemens Networks
• About Agile methods
  – Values
• Agile in NSN
  – The NSN Agile “variant”
• Case: NetAct Optimizer
• Usability lessons learnt
About Nokia Siemens Networks

- Nokia Siemens Networks is a leading global enabler of communications services
- For more information visit us at www.nokiasiemensnetworks.com

NSN products: e.g. Wireless Radio Access

- For a long time we have been a technology driven company, lately services and software have been getting more ground
NSN products with Usability issues

HW products – Industrial Design for: Base Stations, Base Station Controller, etc.

SW Products – UI Design for:
• Business Support Systems
• Service Management Systems
• Operations Support Systems
• Element Management Systems

Documentation
• On-line documentation
• Paper documentation
• Procedural
• Product manuals

Various End-Users during Network life-cycle
SW Development Models

Extreme “Ad hoc” Model

1) Write code
   “I just can’t wait to get started!”

2) Think about what to code
   “... but I’ll start coding the first idea that comes to my mind...”

3) Think about how to code it
   “... I’ll design while I’m coding...”

4) Verify the results ASAP
   “I just can’t wait to see something working.”

- Me making an excel solution for myself
Extreme Waterfall Model

- Specify up-front, carefully and thoroughly whatever you are going to code.
- You can do coding only if it has first been specified, documented, reviewed and accepted.
- Commit 100% to your specifications. Changes will be regarded as failures.

If your plans fail, do more planning?!

Waterfall – pro’s and con’s

✓ Understand and design the big picture before coding
✓ Agreed and documented long term target for coding activities
✓ Planning makes implementation work more straight-forward
✓ Early visibility of intentions to internal & external stakeholders
  • Customers, managers, other R&D teams, etc.
✓ Early evaluation of design ideas
✓ Externalization of ideas helps understand them

✗ Analysis-paralysis?
✗ Implementation might be faster but overall duration gets longer
✗ Overdoing risk management for simple, routine or small coding tasks
✗ Lots of deliverables that is not tangible, sellable or functional
✗ Heavy mental burden to be right
✗ Heavy review process needed to manage any changes arising after coding has started
Overdoing waterfall...

V-model

• The V-model is based on the waterfall model
  - Decomposition used as basis for Project Management
    - Divide organization to teams based on roles (specification, design, implementation, testing,...)
    - Divide sw system to subsystems and components
    - “Divide and conquer”: strategy where small power groups are prevented from linking up and becoming more powerful, since it is difficult to break up existing power structures. (www.wikipedia.org)
Waterfall vs. “Ad hoc” sw development models

**Waterfall**
- "Government-contract" model
- Plan-driven
- "Well planned is half-done"
- Expect perfect planning and prediction
- "No documentation \(\rightarrow\) no design"

**Ad hoc**
- "No discipline model"
- Code-driven
- "Well planned is none-done"
- Utilize trial-and-error, learn by doing
- "All you need is code", "code is the design"

- Of all different sw development situations, the extreme models work only in few cases \(\rightarrow\) one size does not fit all

Agile SW development in NSN
Software Development models – visibility of progress

**Waterfall**
- Each time-boxed iteration (1-4 weeks) produces some functionality.
- The working product grows by small steps.
- Each iteration contains planning, analysis, design, code, test, integration and verification, which are done in parallel.

**Agile**
- Each time-boxed iteration (1-4 weeks) produces some functionality.
- The working product grows by small steps.
- Each iteration contains planning, analysis, design, code, test, integration and verification, which are done in parallel.

- When is the project half-done? When is the product half-done?

Values, Principles and Practices for sw project and product management

- No one Agile. No Agile Methodology.
  - But there are many practices that express the values
- Be agile! (You can not say “do agile”)
Values: Agile Manifesto

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.”

Agile Principles

1. Our highest priority is to **satisfy the customer** through early and continuous delivery of **valuable software**
2. **Welcome changing requirements**, even late in development. Agile processes harness change for the customer’s competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to a shorter timescale.
4. Business people and developers must **work together daily** throughout the project.
5. Build project around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within development team is face-to-face conversation.
7. Working software is the primary measure for progress.
8. Agile processes promote sustainable development. The sponsors, developers, and **users should be able to maintain a constant pace indefinitely**.
9. Continuous attention to **technical excellence and good design** enhances agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.
11. The **best architectures, requirements, and designs emerge from self-organizing teams**.
12. At regular intervals, the team reflect on how to become more effective, then tunes and adjusts its behavior accordingly.

BTW: Have you defined your company’s usability values and principles?
Origins of Agile manifesto

“Waterfall model: heavily regulated, bureaucratic, slow, demeaning, and inconsistent with the ways that software engineers actually perform effective work”

Planning

<table>
<thead>
<tr>
<th>Planning level</th>
<th>Freq.</th>
<th>By whom</th>
<th>Focus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>2 times per year</td>
<td>Product owner and business managers</td>
<td>Product evolution, ie, product and release lifecycle</td>
</tr>
<tr>
<td>Release</td>
<td>3-4 times per year</td>
<td>Product owner and team</td>
<td>Tradeoffs between features, resources and schedules for the release</td>
</tr>
<tr>
<td>Sprint (i.e. iteration)</td>
<td>Every iteration (2-4 weeks)</td>
<td>Product owner and team</td>
<td>What features can be delivered within the iteration</td>
</tr>
<tr>
<td>Daily</td>
<td>Every day</td>
<td>Scrum Team</td>
<td>How to complete committed tasks and features</td>
</tr>
</tbody>
</table>

The planning “onion”
- Planning happens at all levels on a regular basis
- The inner levels get their scope from the outer level
- The outer levels are concerned with and receive feedback from the inner levels
- Agile teams or projects plan at the innermost 3 levels, outer levels are planned in the respective businesses or organizations
NSN Agile practices: key issues 1/2

• Scrum
  – Method for project management
  – Self-organizing cross-functional teams
    ▪ 5-9 persons
  – Daily 15min standing scrum meetings
    ▪ What have you done since yesterday? (accomplishments)
    ▪ What are you planning to do by tomorrow? (to be accomplished)
    ▪ Do you have any problems preventing you from accomplishing your goal? (issues/concerns/risks)

• Iterative
  – Time-boxed (2-4 weeks) iterations (=sprints)
  – Time fixed → content de-scoped

• Continuous integration
  – Tested and working software after every sprint

NSN Agile practices: key issues 2/2

• Feature Driven Development
  – Work is divided based on features that provide value to the customer
  – Split features to fit to releases and sprints

• Demos after every sprint
  – The working software is demonstrated to stakeholders at the end of each sprint

• Product Backlog
  – Prioritized list of features

• Sprint Backlog
  – List of R&D tasks

• Some 40 NSN projects have been applying Agile
  – Internal team established to support new agile projects
  – Craig Larman has been used as one of our consultants (www.craiglarman.com)
About NetAct Optimizer® product

• Used to help optimize the performance of the network

• Some features:
  • Adjacency optimization
  • Frequency optimization
  • Administration type of workspaces
  • Scrambling code allocation
  • Use case helper
  • Plan management and CM data exchange
  • Visibility
  • WCDMA measurements and WCDMA Interference matrix

GUI shown only in live presentation.
NetAct Optimizer background

- Visualisation of network data is a critical GUI feature
- Several releases already delivered to customers
  - Usability feedback has been collected and used as input for next releases
- Agile principles have been applied for a few years
  - Process improvement still active
- Nowadays several UI Design and Usability specialists are in the teams
- Relationships have been established with real users

Key roles

**Product Management**
- **Product Manager**
  - decides the release contents
- **Feature Owners**
  - GSM
  - WCDMA
  - Visualisation
  - Usability (also in matrix)
  - etc...

**Scrum (project management)**
- **Scrum Master**
  - Daily work management
- “Pigs”
  - Team committed to implement the features. User stories completed in priority order.
- “Chicken”
  - Supporting and following the team work. Not committed to implementation
NetAct Optimizer – Defining Product Contents

1. **Features**
   - Product Backlog items
   - Product properties that have some value to the various stakeholders
   - Scoped to fit into one release
   - Candidate/Approved/Rejected items
   - Features are split to user stories

2. **User Stories**
   - Release Backlog items
   - Format: "As a <Role> I can do <this-and-that>"
   - Typically fit into one sprint
   - DONE criteria agreed for the story (e.g. implemented, tested, usability evaluated, customer documentation done etc)
   - User stories are transformed to R&D tasks

3. **R&D Tasks**
   - Sprint backlog items
   - Action points that need to take place to implement the user stories (and related features)

Usability specialist’s responsibilities

- As a feature owner, proposes and prepares feature candidates to the Product Manager
- Evaluates other feature candidates from usability point of view
- Specifies features to a level of detail that the value to customer can be estimated
  → Does GUI visualisations and evaluations to critical and unclear features
- Participates in writing and evaluating user stories
- Looks for customer value when release backlog items are allocated to releases → defines operator workflows when needed
- Is involved in daily scrum meetings. If this is not possible, is available whenever needed
- Fast feedback and early defect findings
Optimizer drivers for User Centred Design

- Specification
  - Description of "big picture" and whole workflow
  - "Looking for simple solutions for complex problems"
  - Customer involvement by providing feedback for prototypes
- Implementation
  - UI following Look & Feel guidelines and icon "language"
- Testing
  - Usability evaluation
  - End to end solution testing
- Documentation
  - "Simple and clear instructions and examples"
  - Participation to UI design already in early phase

Challenges and lessons learnt
Key usability concerns for Agile SW development

**Guidance for User Centred Design?**

- What guidance does Agile provide for doing user centred design activities?
  - Value#2: “Individuals and interactions over processes and tools”,
  - Principle #11: “The best architectures, requirements, and designs emerge from self-organizing teams.”

- Get skilled people and make them work together? No need for competence development nor design guidelines?
- There is a wide set of agile practices, make your pick!?

---

Key usability concerns for Agile SW development

**How does Agile understand usability?**

- How is usability, as a quality attribute of valuable software (Agile principle #1), understood and prioritized in Agile?

- Agile does not talk about product quality. It is a set of techniques (values, principles and practices) for product and project management
- Agile does not make a difference between making UI or embedded software. Practises are written mostly having the latter in mind
Customer collaboration solves usability and user centred design concerns?

Yes, if the single end-user
- Represents all customer end-users (= is the only end-user)
  - Does not have any personality disorders ;-)  
- Will take the responsibility if the usability of the product fails
- The customer does not suffer from design-blindness (as the R&D team does)
- Is an expert of state-of-art UI design
- Is an expert of company Look & Feel
- Is available all the time
  - If not, we need to plan and prepare carefully what to do during the 2 weekly hours we have together

- Of course, one end-user is always a lot better than none

“Working software over comprehensive documentation (and specification)”  
Agile Manifesto and usability

Guys, I took some extra time to design this prototype...

No thanks!

Where were you?

We already made a working system.

Dude, why don’t you do something useful and give a push.

Usability Dude

Hard Core Agilists
“Welcome changing requirements”
Agile Principles and usability

- To err is human
  - The team must be able to admit that the first implemented designs might not be the best ones
  - There has to be a readiness to improve the design, even if it’s “only” a usability correction to an already “working” system
  - If the team applies zero tolerance to bugs → apply zero tolerance also for usability problems
  - Check that usability problems have a balanced priority with SW bugs

“I think it is not usable...
Sure it is, you just need 1 guy pulling and 2 pushing.
Besides, the end-users are expert pushers and pullers.

“Working software over documentation”
Agile Manifesto and usability

- The later your feedback comes...
  - ... the more feedback you will have to discuss (and maybe argue)
  - ... the more change resistance you will face
  - Doing usability evaluations to early working demos is good.
  - Participation in the daily design is better.
  - Still, sometimes it might be worth doing up-front design, prior any coding?

It was pretty obvious that it would fail...
You can’t be sure before you try it.
Besides, the good news is that we already have something to deliver.
“Work together daily”
Agile Principles and usability

Teamwork is crucial. Be committed or involved

- The team should have the usability specialist inside the team, not as an outsourced resource
- If not committed, be involved. Otherwise you are as good as a vegetable.
- You sometimes have to give the push and help e.g. in routine testing

Key usability concerns for Agile SW development
When to do User Centered Design? 3 answers

<table>
<thead>
<tr>
<th>Waterfall</th>
<th>Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Design</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hard Core Agile</th>
<th>Agile</th>
<th>Up-front Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td>The working software is the only thing that really exists and matters</td>
<td>When analysis and design is being done, be there!</td>
<td>Admit that there are 3 different design domains: context of use, UI and technical sw</td>
</tr>
<tr>
<td>User Centred Design focuses on evaluating the working code</td>
<td>Usability Specialist is inside the scrum team and participates in the everyday design decisions</td>
<td>Apply full scale agile for technical sw design</td>
</tr>
<tr>
<td>“The best architectures, requirements, and designs emerge from self-organizing teams.”</td>
<td>Customer collaboration</td>
<td>Do end-user research and UI prototyping up-front and parallel to coding. Use usability specialists for these disciplines</td>
</tr>
<tr>
<td>Who needs user centered design?</td>
<td>Customer must be part of the scrum team. This will automatically lead to high usability(?)</td>
<td>Product manager risk management: “fail fast”</td>
</tr>
<tr>
<td></td>
<td>“Business people and developers must work together daily throughout the project”</td>
<td>Usability recognized as a critical success factor, do up-front GUI prototyping to iterate design prior coding. Apply agile for the design work</td>
</tr>
<tr>
<td></td>
<td>Product manager represents the customer in the scrum meetings</td>
<td></td>
</tr>
</tbody>
</table>

By Clark & Westra © 2006 implementingagile.com
Thank you!
Questions and Answers...

Oh, you are still there.
While the guys are coding
I have a few more ideas to show... unless of course they are not too late...
Design “onion” – the three design layers

- Customer (impact)
  - External sw design
  - Internal sw design
  - Coding

- Context design
  - SW Architecture
  - UI technical design
  - Coding

- Business improvement
- Process improvement
- User Experience
- User Satisfaction

The outer the layer is, the less direct is the impact of coding.

Best designs emerge from self-organizing teams
Agile principles and usability

- Each layer requires a different set of skills
- There is a design discontinuity between layers. “What – how” is asked repeatedly
- The inner layers get requirements from the outer ones
- The outer layers are concerned with - and receive feedback from - the inner ones

- The agile team must ask itself: how much of the design work can be managed within the sprints? Is there a need to do design (or feasibility studies) up-front?

- Example representations
  - Process descriptions?
  - Job descriptions?
  - Use scenarios, Use cases, on-line helps?
  - GUI Visualizations, Interaction Design?
  - System Specifications?
  - The higher layer problem you solve – the more innovative your solution is!
Best designs emerge from self-organizing teams
Agile principles and usability

- Agile team work is democratic
- Still, can we expect everyone be a specialist, in all design layers?
- Do not expect the customer to guarantee high usability