Agile SPL-SCM:
Agile Software Product Line
Configuration and Release Management

APLE 2006 – Workshop
SPLC 2006, Baltimore, MD
Reto.Kurmann@phonak.com
Phonak Hearing Systems

Presentation Roadmap

- 1. Introduction Phonak – Hearing – Systems
- 2. Phonak’s SPL Adoption
- 3. SCM in Software Product Lines (background)
- 4. SCM reflects static SPL architecture
- 5. Main Path Centric Development
- 6. Release strategies
- 8. Automate SPL Production Process
- 9. Lean and Agile SCM Processes
- 9.6 Requirements Process (not so lean)
- Questions & Answers
1. Introduction Phonak

- Phonak is specialized in the design, development, manufacture and global distribution of hearing systems.
- Our technologically advanced hearing systems significantly improve quality of life of many people worldwide.
- 555 Million Euros
- 3500 employees
- Sales in 70 countries
- 3 R&D Locations

1. Introduction Hearing Aid
2. Why SPLs in the Hearing Instrument Industry?

- Fitting Application Software – Product family
- Hearing Instruments – Product family

2. Phonak’s Software Product Line Evolution

- Development time for new Hearing Instrument families cut in half
- Continuous improvement in software quality
- Reuse of core assets in 5 Fitting Applications and 55 HIs
3. Software Product Lines Principles

Next generation of reuse
- Products or services which share a common set of features
- These features are managed, to satisfy a specific need, mission or market segment
- The products are developed from a common set of core assets in a prescribed way

Some SPL Principles

- Domain & Application Engineering
- Identify Commonality & Variability
- Define SW Product Line Scope
- Create SW Product Line Architecture
- Create Core Assets and Production
- Integrate Products and Services

3. Software Configuration Management Patterns

by Stephen Berczuk

Intangible Product, but tangible release schedule
- Each product is 100% reproducible for
  - Release version of OTS components, libraries and tools
  - Baseline of core asset components
  - Baseline of production mechanism and environments
  - Version of product-specific components and variations
4. SCM reflects static SPL architecture

- Product Line architecture is reflected in folder structure
- Platform (Core assets) is shared among products as
  — Released version
  — Specific development path based on checkpoint
  — The main path head

5. Main Path Centric Development

- Main path consists of source code of all products and core assets
- New feature development on main path
- Release paths only for verification, validation and bug-fixing
- Bug fixes immediately merged back to main path
- Daily build and smoke tests enforce good quality
6. Release strategies

- Align platform and product releases (release together)
  - Platform validation impossible or too expensive
  - Remove unnecessary time-to-market delay
  - Cluster product release to benefit from verification

- First release of new feature with single product
  - Core asset changes during validation in product
  - May invalidate validation or throw off schedule
  - Disadvantage: Feature delayed for other products

- Releasing source code rather than binaries
  - No need to release binaries -> automated production
  - Allows pre-compile, compile and build variation points

8. Automate SPL Production Process

- Provide tools to configure variation points (8000 for HI) of components
  (216 boolean features \(= 10^{65}\) combinations = #atoms in universe)

- Fully-automated production Build-Server

- Nightly builds and smoke tests with morning reports

- Separate fully-automated Test-Server for unit, component, integration, regression, etc. tests
8. SPL-Production Build Server

Build Server for iPFG, iPDCS, iDACs and PFG

<table>
<thead>
<tr>
<th>Issue Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFG</td>
<td>Build iPFG</td>
</tr>
<tr>
<td>iPDCS</td>
<td>Build iPDCS</td>
</tr>
<tr>
<td>iDACs</td>
<td>Build IDACs</td>
</tr>
<tr>
<td>PFG</td>
<td>Build PFG</td>
</tr>
</tbody>
</table>

Reports:
- Build Status
- Build Log Viewer

Additional Issues:
- Check-in e-mails
- Check-in policy
  - A) Synchronize local sandbox
  - B) Make private system build and smoke test
  - C) Check-in
  - D) Send check-in e-mail
  - E) Kick production server (build & smoke test)
  - F) Have change reviewed
- Stabilization of main path is highest priority of team
- Daily user deployment test
  - Installation and 30’ exploratory testing
- Requirements engineering and change management
  - NO lean processes, but allow controlled late changes

9. Lean and Agile SCM Processes

- Check-in e-mails
- Check-in policy
  - A) Synchronize local sandbox
  - B) Make private system build and smoke test
  - C) Check-in
  - D) Send check-in e-mail
  - E) Kick production server (build & smoke test)
  - F) Have change reviewed
- Stabilization of main path is highest priority of team
- Daily user deployment test
  - Installation and 30’ exploratory testing
- Requirements engineering and change management
  - NO lean processes, but allow controlled late changes
9.6 Requirements Process (not so lean)

- Define overall system as a set of features
- Features are reviewed and approved by all Product Managers (PM) and released by steering board
- Feature requests describe commonality as well as variation points

Comments, Questions & Answers

- Don’t choose just one method
  — Know all of them
  — Taylor to your needs based on
    - Project (Budget, Size, Time)
    - Organization (Teams, People)
    - Process maturity
    - Taylor Disciplines

- Do proper RE and architecture:
  — Specify variation points in RE
  — Implement only needed variation

- Automation is AGILE

- ???