Reconciling Requirements and Architectures with the CBSP Approach in an iPhone App Project

Harald Vogl        Klaus Lehner        Paul Grünbacher        Alexander Egyed

Catalysts GmbH
Ottensheimer Straße 27
4040 Linz, Austria
{harald.vogl | klaus.lehner}@catalysts.cc

Systems Engineering & Automation
Johannes Kepler University Linz
4040 Linz, Austria
{paul.gruenbacher | alexander.egyed}@jku.at
Challenge: Bridging Requirements & Architecture

• How can we refine the requirements into an architecture?
• How can we map new requirements to existing architectural elements?
• How can we deal with both functional and non-functional aspects?
• How can we explore and assess architectural options?
• How can we find additional requirements?

>> Used the Component-Bus-System-Property (CBSP) approach to support these tasks

**Project Background**

**taskmind – getting things done**

Existing desktop and web products  >> *extend with iPhone and iPad client*

- Create and share todo lists with team
- Organize tasks in projects
- Schedule tasks

- ~7,000 users in ~3,000 projects managing ~85,000 tasks and appointments
- Code base ~230 KLOC

www.taskmind.net

©CATALYSTS GMBH | HARALD VOGL
Project Background

taskmind iOS App Demo
Sample Requirements

- F1. Login with existing account
- F27. View news and daily journal in offline mode
- ...
- N7. Synchronization rate 20 seconds
- N8. App startup time maximum 2 seconds
- N15. Minimize amount of data transfer
- ...

Elicitation Methods

- Persona Profiles [1]
- Scenario-based RE [2]
- Volere Template [3]

29 (F)unctional and 25 (N)on-functional requirements


App Project
Adapt and extend Architecture

Adobe Flash
Web Application (Adobe Flex)
Desktop Application (Adobe AIR)

iOS App
Android App
Platform extensions for mobile devices

Blackberry OS App
Windows Phone App

Catalysts Platform
Internet
taskmind Server (Java)
Database

Outlook-Plugin (Microsoft .NET/C#)
CBSP: Component Bus System Property

- A lightweight process for refining requirements into architectures
- Key characteristics
  - Requirements explicitly or implicitly contain architecturally relevant information
  - Use simple taxonomy to bring forth this information
  - Classify and refine requirements according to taxonomy

CBSP: Component Bus System Property

C: Components ($C_d$: data components, $C_p$: processing components)
- F11: Editing Tasks, F... ➔ $C_d$: Data for tasks
- F12-13: Different task states ➔ $C_p$: State transition component

B: Buses (i.e., connectors)
- F1: Login with existing user, F... ➔ B: Communication protocol
- F26: Synchronize offline items when online again
  ➔ B: Connector between offline component and file system

S: (Sub-)systems
- F4-6: List tasks per contact, project, ...
  ➔ S: Strict separation of data and visualization
CBSP: Component Bus System Property

**CP: Component Property**
- N7: Synchronization rate 20 seconds
  - $C_p$: Data refresh component
  - $\rightarrow CP$: Maximum delay of 20 seconds

**BP: Bus Property**
- N15: Minimize amount of data transfer
  - B: communication protocol
  - $\rightarrow BP$: small amount of data

**SP: System Property**
- N25: Secure transfer and storage
  - $\rightarrow SP$: all data must be securely transferred and stored

N27: Offline task support
- Cd: Data for tasks
  - $\rightarrow CP$: persistent
App Project
taskmind
CBSP Model

CBSP model elements
• 12 C (6 Cd, 6 Cp)
• 25 CP
• 3 B
• 6 BP
• 3 S
• 2 SP

>> How to find adequate architectural styles?
Assessing Architectural Options: CBSP and the GQM approach

- Ad-hoc definition of pro’s and con’s of architectural options as suggested by CBSP too complex ➔ Derive goals and metrics from CBSP elements

Requirements: Minimize amount of data transfer, Startup within 2 seconds ➔ CBSP Elements: B: communication protocol,...

BP: small amount of data, fast processing ➔ Best architectural option??

<table>
<thead>
<tr>
<th>Goals</th>
<th>Questions</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Purpose:</em> Compare</td>
<td><em>Issue:</em> Amount of data</td>
<td>Data that needs to be transferred (in bytes).</td>
</tr>
<tr>
<td><em>Object:</em> Different data formats for persisting of data objects</td>
<td>Differences with respect to the amount of data?</td>
<td></td>
</tr>
<tr>
<td><em>Viewpoint:</em> Software architect</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>Purpose:</em> Compare</td>
<td><em>Issue:</em> Performance</td>
<td>Time for (de-) serializing objects (in ms).</td>
</tr>
<tr>
<td><em>Object:</em> Different data formats for persisting of data objects</td>
<td>Differences with respect to serialization and deserialization time?</td>
<td></td>
</tr>
<tr>
<td><em>Viewpoint:</em> Software architect</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Assessing Architectural Options: CBSP and the GQM approach

<table>
<thead>
<tr>
<th>Data format</th>
<th>Bytes transferred</th>
<th>Serialize (ms) (Run 1/2/3)</th>
<th>Deserialize (ms) (Run 1/2/3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom (binary)</td>
<td>360,504</td>
<td>114.8 116.4 135.9</td>
<td>209.5 209.5 184.2</td>
</tr>
<tr>
<td>Hessian (binary)</td>
<td>950,696</td>
<td>1171.6 1150.1 1158.4</td>
<td>4284.4 4280.2 4254.9</td>
</tr>
<tr>
<td>JSON (text)</td>
<td>766,001</td>
<td>2015.1 1959.7 1992.5</td>
<td>4619.5 4328.4 4797.9</td>
</tr>
<tr>
<td>XML (text)</td>
<td>1,250,081</td>
<td>3778.2 3819.1 3796.7</td>
<td>6128.8 6101.8 6098.6</td>
</tr>
</tbody>
</table>

**Feedback to requirements: feasibility?**

**Requirement:** Minimize amount of data transfer

**CBSP Elements:** B: communication protocol
BP: small amount of data

→ New architectural component: custom format

N8. App startup time maximum 2 seconds
## Relating the CBSP Model to the Architecture

<table>
<thead>
<tr>
<th>CBSP Elements</th>
<th>Architecture Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1_Cd: data for users, M4_Cd: data for tasks, M5_Cd: data for appointments,</td>
<td>Data Objects</td>
</tr>
<tr>
<td>M6_Cd: data for lists, M9_Cd: data for projects, M10_Cd: data for tags,</td>
<td></td>
</tr>
<tr>
<td>M12_Cd: data for change history and notes</td>
<td></td>
</tr>
<tr>
<td>M7_Cp: management component, M8_S: strict separation of data and visualization</td>
<td>Proxies</td>
</tr>
<tr>
<td>M11_Cp: task state transition component</td>
<td>State Machine</td>
</tr>
<tr>
<td>M2_B: query service</td>
<td>Services</td>
</tr>
<tr>
<td>M2_BP: lazy loading, M3_BP: small amount of data</td>
<td>Input Handler, Output Handler</td>
</tr>
<tr>
<td>M3_B: communication protocol</td>
<td>Communication Protocol</td>
</tr>
<tr>
<td>M13_Cp: Switch between offline and online mode, M14_Cp: Offline component,</td>
<td>Offline Management</td>
</tr>
<tr>
<td>M15_B: Connection between offline component and file system</td>
<td></td>
</tr>
<tr>
<td>M16_Cp: data refresh component</td>
<td>Auto-Update</td>
</tr>
<tr>
<td>M17_S: low power consumption, M18_S: attention on simple portability</td>
<td>System wide feature affecting all components</td>
</tr>
</tbody>
</table>
Relating the CBSP Model to the Architecture

<table>
<thead>
<tr>
<th>CBSP Elements</th>
<th>Architecture Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1_Cd: data for users, M4_Cd: data for tasks, M5_Cd: data for appointments, M6_Cd: data for lists, M9_Cd: data for projects, M10_Cd: data for tags, M12_Cd: data for change history and notes</td>
<td>Data Objects</td>
</tr>
<tr>
<td>M7_Cp: management component, M8_S: strict separation of data and visualization</td>
<td>Proxies</td>
</tr>
<tr>
<td>M11_Cp: task state transition component</td>
<td>State Machine</td>
</tr>
<tr>
<td>M2_B: query service</td>
<td>Services</td>
</tr>
<tr>
<td>M2_BP: lazy loading, M3_BP: small amount of data</td>
<td>Input Handler, Output Handler</td>
</tr>
<tr>
<td>M3_B: communication protocol</td>
<td>Communication Protocol</td>
</tr>
<tr>
<td>M13_Cp: Switch between offline and online mode, M14_Cp: Offline component, M15_B: Connection between offline component and file system</td>
<td>Offline Management</td>
</tr>
<tr>
<td>M16_Cp: data refresh component</td>
<td>Auto-Update</td>
</tr>
<tr>
<td>M17_S: low power consumption, M18_S: attention on simple portability</td>
<td>System wide feature affecting all components</td>
</tr>
</tbody>
</table>
Relating the CBSP Model to the Architecture

<table>
<thead>
<tr>
<th>CBSP Elements</th>
<th>Architecture Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1_Cd: data for users, M4_Cd: data for tasks, M5_Cd: data for appointments, M6_Cd: data for lists, M9_Cd: data for projects, M10_Cd: data for tags, M12_Cd: data for change history and notes</td>
<td>Data Objects</td>
</tr>
<tr>
<td>M7_Cp: management component, M8_S: strict separation of data and visualization</td>
<td>Proxies</td>
</tr>
<tr>
<td>M11_Cp: task state transition component</td>
<td>State Machine</td>
</tr>
<tr>
<td>M2_B: query service</td>
<td>Services</td>
</tr>
<tr>
<td>M2_BP: lazy loading, M3_BP: small amount of data</td>
<td>Input Handler, Output Handler</td>
</tr>
<tr>
<td>M3_B: communication protocol</td>
<td>Communication Protocol</td>
</tr>
<tr>
<td><strong>M13_Cp: Switch between offline and online mode, M14_Cp: Offline component, M15_B: Connection between offline component and file system</strong></td>
<td><strong>Offline Management</strong></td>
</tr>
<tr>
<td>M16_Cp: data refresh component</td>
<td>Auto-Update</td>
</tr>
<tr>
<td>M17_S: low power consumption, M18_S: attention on simple portability</td>
<td>System wide feature affecting all components</td>
</tr>
</tbody>
</table>
CBSP in the iPhone App Project

Lessons Learned

- Use of CBSP dimensions
  - Lightweight methodology
  - Helped identifying missing requirements
  - Traceability between requirements and architecture

- Using CBSP in the presence of an existing architecture
  - CBSP elements led to changes of the original architecture
  - Quite suitable for modifying an existing architecture

- Understanding architectural options
  - More CBSP properties than non-functional requirements
  - CBSP and GQM are a good fit
CBSP in the iPhone App Project

Lessons Learned

- Flexibility with respect the choice of requirements elicitation method
  - Use of personas and scenario walkthroughs provided a good input for CBSP
  - CBSP helps to complete the requirements

- CBSP for single architects
  - No problem, but not recommended

- CBSP tool support and visualization
  - Visualize CBSP elements and their many kinds of relationships turned out as a problem
  - Maintaining the CBSP model is also cumbersome
App Project

Status and Further Plans

- iOS App and platform extensions
  ~30,000 Lines of Code (Platform: ~18,000, iOS/UI: ~12,000)

- ~300 App Store downloads per week

- Develop taskmind App for Android and Windows Phone
  Use and adapt CBSP model

Thank you for your attention!