Using OSGi for the Realization of Complex Building Management Systems

Peter Schramm & Dimitar Valtchev
Busch-Jaeger Elektro GmbH

- Company in Lüdenscheid (near Dortmund) with nearly 130 years of history
- A member of the ABB group
- Manufacturer of building control systems
Busch ComfortPanel

- Control/application screen
- Navigation bar
- Status bar
Panel evolution

Controlpanel

- KNX/Scenes
- Time programs
- Alarm control unit
- Graphics message
- ...

ComfortPanel

- KNX/Scenes
- Time programs
- Alarm control unit
- Graphics message
- Audio message
- Web/IP
- Audio/MP3
- Video
- ...

Panel evolution
Scenario

- PSTN (Internet)
  - APL
  - Splitter
  - NTBA
  - TK (optional)
- LAN
- WLAN
- KNX (TP / PN)
Target customers

- Actually two customer groups
  - End customer who actually buys panel
  - Electrician who installs and configures panel with regard to the customer’s demands

- Need for powerful easy-to-use configuration utility
  - Device addressing
  - Control screen configuration
  - Application configuration
  - …
Configuration process

Configuration tool (CT) → ComfortPanel (CP)

Configuration data

Preview
Main requirements

- Stability
- Modularity
- Reusability
- Hardware independence

Concept of modular blocks already in early project phase (even before knowing OSGi)!
ComfortPanel architecture
Registers the CP as a UPnP device. It provides the UPnP device properties (serial number, manufacturer, etc) and links the device presentation URL to the Web Access portal.

Enables establishment of a TCP/IP. Like IPP and updates.

Possible to send video over the net and much more,

The DataPoint Engine abstracts the access of CP Applications to external networks and devices, so that they work only with data-points.

- Firmware Handler
- System State Log
- CP Configuration Manager
- Local Settings & Locale Service
- Konnex Stack Config Manager
- Network Camera Control
- Atmel Handler

Legend
- Gray: ProSyst Product
- CP Specific module
- Yellow: 3rd party module
- Green: Future extension
Specific CP mGUI components, such as sliders, tables, and others.

The Application Model defines the general requirements how CP applications SHOULD be implemented.

<table>
<thead>
<tr>
<th>CP GUI Components</th>
<th>CP Application Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPnP Device</td>
<td>Remote Access</td>
</tr>
<tr>
<td></td>
<td>Audio/Video Streaming</td>
</tr>
<tr>
<td></td>
<td>CP Configuration</td>
</tr>
<tr>
<td></td>
<td>CP System Services</td>
</tr>
<tr>
<td></td>
<td>DataPoint Engine</td>
</tr>
<tr>
<td>UPnP</td>
<td>mGUI Library</td>
</tr>
<tr>
<td></td>
<td>Multimedia</td>
</tr>
<tr>
<td></td>
<td>Automation Engine</td>
</tr>
<tr>
<td></td>
<td>Web Frk/wGUI</td>
</tr>
<tr>
<td></td>
<td>KNX</td>
</tr>
</tbody>
</table>

mBS

<table>
<thead>
<tr>
<th></th>
<th>Java Virtual Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>mBSA</td>
<td></td>
</tr>
<tr>
<td>Native Applications &amp; Libraries</td>
<td></td>
</tr>
<tr>
<td>Operating System - Linux</td>
<td></td>
</tr>
</tbody>
</table>

**Comfort Panel**

**Legend**

- ProSyst Product
- 3rd party module
- CP Specific module
- Future extension
Configuration tool architecture
<table>
<thead>
<tr>
<th>IPP Editors</th>
<th>IPP Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPP Library</td>
<td>IPP Parameters Area</td>
</tr>
<tr>
<td>IPP Help</td>
<td>IPP Help</td>
</tr>
<tr>
<td>Preview</td>
<td></td>
</tr>
<tr>
<td>CP Config</td>
<td>UPnP</td>
</tr>
<tr>
<td>Help, Search</td>
<td>GEF, Draw2D</td>
</tr>
<tr>
<td>IPP Editors</td>
<td>IPP Objects</td>
</tr>
<tr>
<td>IPP Library</td>
<td>IPP Parameters Area</td>
</tr>
<tr>
<td>IPP Help</td>
<td>IPP Help</td>
</tr>
<tr>
<td>Preview</td>
<td></td>
</tr>
<tr>
<td>CP Config</td>
<td>UPnP</td>
</tr>
<tr>
<td>Help, Search</td>
<td>GEF, Draw2D</td>
</tr>
<tr>
<td>IPP Editors</td>
<td>IPP Objects</td>
</tr>
<tr>
<td>IPP Library</td>
<td>IPP Parameters Area</td>
</tr>
<tr>
<td>IPP Help</td>
<td>IPP Help</td>
</tr>
<tr>
<td>Preview</td>
<td></td>
</tr>
<tr>
<td>CP Config</td>
<td>UPnP</td>
</tr>
<tr>
<td>Help, Search</td>
<td>GEF, Draw2D</td>
</tr>
<tr>
<td>IPP Editors</td>
<td>IPP Objects</td>
</tr>
<tr>
<td>IPP Library</td>
<td>IPP Parameters Area</td>
</tr>
<tr>
<td>IPP Help</td>
<td>IPP Help</td>
</tr>
<tr>
<td>Preview</td>
<td></td>
</tr>
<tr>
<td>CP Config</td>
<td>UPnP</td>
</tr>
<tr>
<td>Help, Search</td>
<td>GEF, Draw2D</td>
</tr>
</tbody>
</table>

**Provides a highly scalable, open-ended, multi-window environment**

A UI framework, layered on top of SWT, for handling many common UI programming tasks.

The Standard Widget Toolkit is designed to provide efficient, scalable, and portable access to the user-interface facilities of the operating systems on which it is implemented.

Provides the foundational support for plug-ins, extension points and extensions. It is built on top of the Equinox OSGi framework.

**Runtime (OSGi)**

**Legend**

- ProSyst Product
- IPP Specific module
- 3rd party module
Provides web-application-based Help UI, with support for dynamic content. Search allows indexing the HTML content and quick finding of particular help topics.

Allows users to discover and install updated versions of products and extensions. The updates can be located either on the Web or on the local file system.

Framework for building graphical editors. Includes Draw2D, a vector graphics framework. Allows developers to create a rich graphical editor from an existing application model.

Legend

- **ProSyst Product**
- **IPP Specific module**
- **3rd party module**
Responsible for communicating with the separate process that runs on the same machine a copy of the Comfort Panel framework – starting, uploading, updating, returning back, etc.
Main technical challenges

• Complex integration work – especially on the Comfort Panel side (Hardware, Linux, KNX bus system, Multimedia)

• Realization of a rich and complex environment for panel configuration and control (CT)
Benefits of using OSGi (1)

- Straightforward design because of the availability of reference architectures and numerous ready components
- Shorter and easier implementation phase – a lot of available components, both Open Source and Commercial OTS
- Allows the same underlying technology to be used on CP and CT
Benefits of using OSGi (2)

• Well suitable for heterogeneous environments (the CP stack can be run from the CT without any modifications!)
• Convenient testing – the CP software can be tested on PC
• Significant benefits for the product deployment and QA work
Thank you for your attention!