How to Use the OSGi Service Framework in Real-Life Projects?
ProSyst Facts & Figures

- Founded 1997 in Cologne, Germany
- Capital Stock € 4.3 M
- OSGi (software standard) member since 1999
- Shareholder with 16+ percent
- Leading technology partners like SAP, HP, etc.
- 180 employees with focus on OSGi, (Embedded) Java
  (150 Java engineers/developers/consultants)
ProSyst Products

**Embedded Software**
- mBedded Server

**Development IDE**
- mBedded Builder

**Remote Management Software**
- mPower Remote Manager

© copyright 2004 by OSGi Alliance. All rights reserved.
mBedded Server is:

- Service Platform, compliant with OSGi Service Platform Release 3 and enhanced with valuable extensions and additional features from ProSyst
- Scalable platform with small memory footprint and minimal system requirements, optimised for a broad range of embedded devices.
- Modular and enables the dynamic adding or removing of software modules (bundles) at any time, in order to have only the needed modules running
- Flexible and lets specify a lot of configuration properties for dynamically changing of the runtime behavior
- Hardware independent and optimized for different OS/RTOS and JVM’s
- Delivered with user-friendly tools for fast development of services and easy administration of the service platform

© copyright 2004 by OSGi Alliance. All rights reserved.
mBedded Server implementation

ProSyst implementation of the OSGi R3 Specifications:

The ProSyst Service Platform contains implementation of all current specifications defined by the OSGi Alliance, as well as valuable extensions to the standardized services:

- **Device Manager**
- **Http Service**
- **Log Service**
- **Preferences Service**
- **Configuration Admin**
- **Service Tracker**
- **User Admin**
- **Package Admin**
- **Permission Admin**
- **Wire Admin**
- **XML Parser Service**
- **Measurement**
- **Position**
- **Connector Service**
- **Jini Service**
- **UPnP Service**
- **Start Level**
- **URL Handler**
mBedded Server enhancements

In order to meet various deployment requirements, ProSyst has added multiple enhancements to the implementation of OSGi service platform specification:

• **Extending the System Bundle** - The framework provides a mechanism for extending the functionality of the System Bundle. This includes adding of new system services, exporting of packages, assigning configurations to the system bundle

• **Bootstrapping and Framework initialization** mechanism enables:
  • Developers to create customized and optimized initial configurations (list of bundles and configuration with which gateway to be started)
  • Remote staring / stopping of the JVM so if something goes wrong, the necessary actions can be taken

• **Boot Loader** extends the framework manageability by enabling local and remote restarts of the framework and of the JVM process.
mBedded Server enhancements

• **Flash Manager service** solves the common problem for most embedded devices using the fast RAM, and the slow Flash for persistent storing – the service transfers files from the Flash in temporary RAM files and synchronizes the RAM files with the persistent storage in a background process.

• **JVM Specific Optimizations** - optimized bundles formats, which speeds up the system starting and reduces the consumed memory. It works with different types for storing archives like
  • J9 JXE Storage for XIP (execute-in-place)
  • TAO native images.

• **Resource Management** - provides monitoring and managing of the allocated resources by each bundle running in the framework. The utility allows controlling:
  • Active threads and data spaces for any JVM by using conventional Java mechanisms.
  • Bundle memory spaces and sockets for J9.
mBedded Server Additional Features

ProSyst has developed valuable capabilities on top of the framework, which are necessary in real systems:

- **Power monitoring and management** - retrieves the power consumption and the battery status and manages bundles and groups of bundles (start or stop bundles) in order to maintain the necessary power supply to keep the framework functioning.

- **Security Model** - The ProSyst Framework enhances the basic security model defined by the OSGi specification and is fully capable to process the necessary permissions and trusted certificates in the bundle JAR archives.
mBedded Server Additional Features

- **Control Units and Automated Control**
  - The Control Unit abstraction allows to introduce various types of devices (such as UPnP, LonWorks, OneWire, EIB, EHS, X10) as Control Units with unified and uniform interface, which makes their control easier.
  - The automation and simultaneous execution of operations is facilitated by the unified representation as Control Units.

- **Performance Improvements**
  - Advanced **threads management** facilitates the work with privileged threads and the according security checks, which reduces the memory and CPU time usage.
  - **Thread pooling** and **object pooling** optimize the resources consumption and allow their reuse, which improves the overall performance.
  - **Events delivering** - The framework uses a separate thread to notify bundles about bundle and framework events.

© copyright 2004 by OSGi Alliance. All rights reserved.
## Hardware Requirements

- Flash 8 MB - minimum
- Boot loader capable of starting an OS image
- RAM:
  - 16 MB minimum for the framework with basic set of bundles
  - 32 MB for framework with other applications
- Ethernet adapter or serial port or parallel port

<table>
<thead>
<tr>
<th>Profile</th>
<th>CPU</th>
<th>RAM</th>
<th>Flash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum (Record)</td>
<td>50 MHz</td>
<td>8 MB</td>
<td>4 MB</td>
</tr>
<tr>
<td>Low End</td>
<td>50 MHz</td>
<td>16 MB</td>
<td>8 MB</td>
</tr>
<tr>
<td>Medium</td>
<td>200 MHz</td>
<td>32 MB</td>
<td>16-32 MB</td>
</tr>
<tr>
<td>High End</td>
<td>400 MHz</td>
<td>&gt; 32 MB</td>
<td>&gt; 32 MB</td>
</tr>
</tbody>
</table>

© copyright 2004 by OSGi Alliance. All rights reserved.
mGUI Library

- Lightweight graphics library for embedded devices, such as PDAs, STB, and infotainment systems
- Combines high speed with small memory footprint
- Extensive support for TV-screen resolution and input controls for interactive digital TVs
- Available in two versions:
  - as a stand-alone library
  - as an OSGi-compliant library (wrapped for OSGi deployments)
- Contains a wide range of components

- Layered architecture and support of AWT, SDL and GAPI Draw libraries ensure high portability

© copyright 2004 by OSGi Alliance. All rights reserved.
mBrowser

mBrowser is a compact and efficient web browser for small graphical devices with limited resources.

- Fully-featured browsing is attained with minimal resource consumption
- Support for AWT, SDL and GAPI Draw native libraries through mGUI overlays for high portability
- Extensible architecture allows for use of plugins and custom GUI
- Specially adjusted UI for Desktop/PDA and TV sets
- Support of most Internet Standards

## HTML 4.01
- Lists, tables, forms, frames
- GIF, JPG & PNG
- Applets

## HTTP 1.1
- web-page cache and cookies
- persistent connections
- compressed HTML

## JavaScript 1.5

## XML & XHTML

## DOM 2

## CSS 2

## HTTPS
- SSL3.0 & TLS1.0 client

## Flash
mPower Remote Manager

- Remote administration of all RSGs
- Monitor Service Platform vitals (periodic health check)
- Provides view into system details: troubleshoot and solve problems remotely
- Supports SW downloads of OS, JVM, OSGi and applications
- Initiates delivery and installation of new applications
- Integration with J2EE servers and other server-side systems
Service Platform Components

• mPRM Management Agent installed on behalf of mPRM during initial provisioning

• Communication bundle with the appropriate transport bundle(s)

• Optional mPRM bundles:
  • User management
  • Preferences management
  • JMS
  • Service subscription
  • Accounting
  • Driver locator
mPRM subsystem

- Gateways management
- Bundles inventory
- User management
- Service & subscription management
- Payment system
- Integration with external systems and standards
- Administration tools – mConsole, Web Portal
Service Catalogue

- Maintains database with service packages
- Provides tools for assembling bundles in complete service package
- Supports association and managing of payment information for using each service
- Supports different service variant to be defined for different target customers, possibly on different price
- Provides access (through API and UI) to the information in the catalogue for both service providers and service gateway users
Service Subscription and Service Delivery

• Providing user oriented subscription management
• Providing support for different service delivery policies
  ➔ Automatic - bundle(s) are installed on time of subscription on all gateways belonging to the subscribed user and are ready to use
  ➔ On demand - bundles are installed when the user activates the service on a particular gateway and can be uninstalled when the user finish using the service
• Providing mechanism for different charging models:
  ➔ Initial subscription price; time period base fee; service defined
  ➔ Charge once per user; charge per every gateway
• System must ensure that all bundles supporting given service will be available on the gateways, according to delivery policies
Payment System

• Open and flexible payment infrastructure with transactional behaviour of the payments and enhanced security

• Available integration with external payment systems:
  • Support for Jalda™ Charge Processor
  • Support for PayFlow™ Charge Processor

• Can be easily extended for exchanging data with other payment systems

• Provides automatic and custom generation of charges, as well as generation of billing reports

• Supports various payment policy models:
  • Automatic Charging Model: can use Initial purchase, Pay-for-use or Pay-for-period methods
  • Application-Driven Charging Model: enables the generation of charge requests by the service modules themselves.
mPRM enables developers to integrate applications deployed on J2EE application servers with the mPRM system and to access services running on gateways. It provides the following development facilities for J2EE applications:

• Access to resources managed by mPRM - service gateways, bundles, service packages, service subscriptions, etc.

• Lightweight mechanism for communication and data exchange with applications residing on service gateways.