Enhancement Framework

- The new way to enhance your ABAP systems

Oliver J. Mayer
Product Manager – SAP NetWeaver
Learning Objectives

As a result of this session, you will be able to:

- Understand the fundamental idea of the Enhancement Framework and Switch Framework (available in SAP NetWeaver 2004s)
- Reduce TCO by using enhancement technologies instead of modifications
- Enhance SAP standard objects
- Understand how Enhancement definitions are created
Motivation & Overview

Enhancement Framework
- Source Code Plugin - Technology
- Function Group Enhancement - Technology
- Class Enhancement - Technology

Kernel-BAdI – Technology

Switch Framework

Summary
Motivation & Overview

Enhancement Framework

- Source Code Plugin - Technology
- Function Group Enhancement - Technology
- Class Enhancement - Technology

Kernel-BAdI – Technology

Switch Framework

Summary
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Unification</td>
<td>Master-Data Harmonization</td>
<td>Master-Data Consolidation</td>
<td>Central Master-Data Management</td>
<td>Enterprise Data Warehousing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business Information Management</td>
<td>Enterprise Reporting, Query, and Analysis</td>
<td>Business Planning and Analytical Services</td>
<td>Enterprise Data Warehousing</td>
<td>Enterprise Knowledge Management</td>
<td>Enterprise Search</td>
<td></td>
</tr>
<tr>
<td>Business Event Management</td>
<td></td>
<td></td>
<td>Business Activity Monitoring</td>
<td></td>
<td>Business Task Management</td>
<td></td>
</tr>
<tr>
<td>Custom Development</td>
<td>Developing, Configuring, and Adapting Applications</td>
<td></td>
<td></td>
<td>Enabling Platform Interoperability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unified Life-Cycle Management</td>
<td></td>
<td></td>
<td>Software Life-Cycle Management</td>
<td>SAP NetWeaver Operations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consolidation</td>
<td>Enabling Platform Interoperability</td>
<td>SAP NetWeaver Operations</td>
<td>Master-Data Consolidation</td>
<td>Enterprise Knowledge Management</td>
<td>Enterprise Data Warehousing</td>
<td></td>
</tr>
<tr>
<td>ESA Design and Deployment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Enabling Enterprise Services</td>
<td></td>
</tr>
</tbody>
</table>
One of the advantages of SAP software is the possibility to adapt the software to own requirements and the possibility of keeping the adaptations during upgrade.

**Ways of adaptation:**

- Customizing
- Enhancement
- Modification
Motivation

Reducing TCO

- Enhancing objects instead of modifying them reduces the effort for adjustment during SP import or upgrade.

Disadvantages of modifications

- No support for multiple users or projects
- No support for parallel developments
- Will appear much more often in adjustment tools
- Higher adjustment effort (during upgrade & SP import)
Evolution of SAP Enhancement Technology

Application

Kernel

Enhancement Framework

Customer Exits

Form routines

Business Transaction Events

Industries

Business Add Ins

Filters Classes

Function modules

Function modules

User Exits

Business Add Ins

Customer Exits

Kernel based Business Add Ins

© SAP AG 2006, Enhancement Framework, Oliver J. Mayer / 9
<table>
<thead>
<tr>
<th>Motivation &amp; Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enhancement Framework</strong></td>
</tr>
<tr>
<td>Source Code Plugin - Technology</td>
</tr>
<tr>
<td>Function Group Enhancement - Technology</td>
</tr>
<tr>
<td>Class Enhancement - Technology</td>
</tr>
<tr>
<td>Kernel-BAdI – Technology</td>
</tr>
<tr>
<td>Switch Framework</td>
</tr>
<tr>
<td>Summary</td>
</tr>
</tbody>
</table>
Search for

- Enhancements possibilities (Definitions – typically provided by SAP)
- Enhancement Implementations (typically done by Customer)

Integrated into Object Navigator (SE80)
Motivation & Overview

**Enhancement Framework**

- Source Code Plugin - Technology
- Function Group Enhancement - Technology
- Class Enhancement - Technology

Kernel-BAdI – Technology

Switch Framework

Summary
Source Code Enhancements Overview

Modification-free enhancement of source code

Implicit Enhancement Option

- At common enhancement places, implicit Enhancement options are available. Examples:
  - End of Executable Program, Include, Function group, Dialog module
  - Begin/End of Form routine / Function module / Method
  - End of a structure
  - End of Private/Protected/Public Section of a local class
  - ...

Explicit Enhancement Option

- Predefined enhancement options can be defined in source code. They are additionally stored inside Enhancement Spots.
Implicit Enhancement Options

```
* Listoutput
  LOOP AT gt_flights INTO gs_flights.
  WRITE / gs_flights-carrid,
  gs_flights-connd,
  gs_flights-fldate,
  gs_flights-price,
  gs_flights-currency.
ENDLOOP.

* METHOD SET_PROPERTY.
  .........................................................."$"SE:(1) Include ZSEFY_00_FLIGHTS_DISPLAY
  .........................................................."$"SE:(2) Klasse CL_GUI_CONTROL, public $"SE:(3) Klasse CL_GUI_CONTROL, public
  .........................................................."$"SE:(4) Klasse CL_GUI_OBJECT, Methode 0
  * (PROPERTY, P1..P4, P_COUNT, QUEUE_ONLY, => VALUE)
  * check handle
    DATA : STRPROPNAME TYPE STRING.
    DATA : CTRLVALUE TYPE STRING.
    DATA : QUEUE_EMPTY TYPE CHAR01.
    ...
```
Explicit Enhancement Options

* Selection screen

SELECT-OPTIONS:
  so_carr FOR gv_carrid,
  so_conn FOR gv_connid.

ENHANCEMENT-POINT FLIGHTS_DECLARATION SPOTS FLIGHTS_DISPLAY STATIC.

START-OF-SELECTION.

* Select Data

ENHANCEMENT-SECTION   FLIGHTS_DBSELECT SPOTS FLIGHTS_DISPLAY.
  SELECT carrid connid fldate price currency
     from sflight
     into table gt_flights
     where carrid in so_carr
           and connid in so_conn.
END-ENHANCEMENT-SECTION.

* Listoutput

    LOOP AT gt_flights INTO gs_flights.
    WRITE: / gs_flights-carrid,
           gs_flights-connid,
           gs_flights-fldate.
PROGRAM p1.
WRITE 'Hello World'.
ENHANCEMENT-POINT ep1 SPOTS s1.
  ..
  ..
  ..
ENHANCEMENT-SECTION ep2 SPOTS s1.
  WRITE 'Original'.
END-ENHANCEMENT-SECTION.

ENHANCEMENT 1.
  WRITE 'Hello Paris'.
ENDENHANCEMENT.

ENHANCEMENT 2.
  WRITE 'Hello London'.
ENDENHANCEMENT.

ENHANCEMENT 3.
  WRITE 'Enhanced'.
ENDENHANCEMENT.
Editor Modes for Enhancements

Use **Change Mode** for creating enhancement points & sections.

- use button „Display <-> Change“ to switch to change mode.

Use **Enhancement Mode** for creating enhancement implementations.

- use button „Change Enhancements“ to switch to Enhancement mode
- use button „Display <-> Change“ to leave Enhancement mode
Motivation & Overview

Enhancement Framework

Source Code Plugin - Technology

Function Group Enhancement - Technology

Class Enhancement - Technology

Kernel-BAdI – Technology

Switch Framework

Summary
Function Group Enhancements allow:

- Adding new optional parameters to existing function modules

Transaction: SE37
Demo

Function Group Enhancement
Motivation & Overview

Enhancement Framework

- Source Code Plugin - Technology
- Function Group Enhancement - Technology
- **Class Enhancement - Technology**

Kernel-BAdI – Technology

Switch Framework

Summary
Class/Interface Enhancements allow addition of:

- optional parameters to existing methods
- methods
- events and event handlers
- references to interfaces
- Exits to existing methods
  - Pre-Exit – Called at the beginning of a method
  - Post-Exit – Called at the End of a method
  - Overwrite-Exit – Replaces the original method
Adding new methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Level</th>
<th>Visibility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFTER_IMPORT</td>
<td>Static Public</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE</td>
<td>Static Public</td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ</td>
<td>Static Public</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE</td>
<td>Static Public</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD_METHOD_DONE</td>
<td>Instant Private</td>
<td></td>
<td>addional functionality public</td>
</tr>
<tr>
<td>ADD_METHOD_PRIVATE</td>
<td>Instant Private</td>
<td></td>
<td>addional functionality private</td>
</tr>
</tbody>
</table>

Adding optional parameters to existing methods

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>P</th>
<th>O</th>
<th>Typing</th>
<th>M</th>
<th>Associated Type</th>
<th>Default value</th>
<th>Description</th>
<th>Enhancement</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJ_NAME</td>
<td>ImportIn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TROBJ_NAME</td>
<td></td>
<td>Object Name in Object Directory</td>
<td></td>
</tr>
<tr>
<td>PROTOCOL</td>
<td>ChangIn</td>
<td></td>
<td></td>
<td>Type</td>
<td></td>
<td>SPROT_U_TAB</td>
<td></td>
<td>Table Type for SPROT_U (Log In)</td>
<td></td>
</tr>
<tr>
<td>MY_ADD_PARAM</td>
<td>ImportIn</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>CHAR10</td>
<td></td>
<td>Characterfeld der Länge 10</td>
<td>MATCHED2005_001</td>
</tr>
<tr>
<td>MY_ADD_PARAM_EXP</td>
<td>ExportIn</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>INT4</td>
<td></td>
<td>Natürliche Zahl</td>
<td>MATCHED2005_001</td>
</tr>
</tbody>
</table>
Call method instance->hugo( ).

...
Demo
Class Enhancement
Motivation & Overview

Enhancement Framework

Source Code Plugin - Technology
Function Group Enhancement - Technology
Class Enhancement - Technology

Kernel-BAdI – Technology

Switch Framework

Summary
BADIs - Overview

What are BAdIs?

- Business Add-Ins
- is an anticipated point of extension – these points act like sockets and exist in the original coding
- has a well-defined interface in contrast to source code plug-ins and is therefore more stable to changes in the original coding

Kernel BAdIs - New Features

- Are integrated directly in the ABAP Language/Runtime
- Improved filter support allows non-character filter types (packed, numeric, string) and complex filter conditions
- Enable reusable implementation instances (Stateful BAdI)
- Control of the lifetime of implementations (BAdI-context)
- Allow for inheritance of implementations
- Can be switched by the Switch Framework
Comparison: Usage of Old BAdIs vs. new BAdIs

With Classic BAdI

DATA: bd TYPE REF TO if_intf.
DATA: flt TYPE flt.

CALL METHOD cl_exithandler=>
get_instance
EXPORTING
exit_name = `BADI_NAME`\nCHANGING
instance = bd.

flt-lang = `D`.
CALL METHOD bd->method
EXPORTING
x = 10
flt_val = flt.

selecting implementations and issuing calls is mixed up
calls cause DB access
calls are redirected over a proxy class

Old BAdIs are more expensive than the new ones.

With New BAdI

data bd type ref to badi_name.
get badi bd filters lang = `D`.
call badi bd->method
    exporting x = 10.

selection occurs when the handle is requested
no DB access during runtime
Implementations are called directly (without a proxy)

Active implementations are evaluated at compile time and included in the load of the BAdI-handle.
New BADI’s and Enhancement Framework

Definition

- Simple Enhancement Spot Spot_1
- BAdI BADI_A
- BAdI BADI_B
- BAdI BADI_C

Implementation

- Simple Enhancement Implementation SEI1
- BAdI Implementation 1
- BAdI Implementation 2
- BAdI Implementation 3
- BAdI Implementation 4
- BAdI Implementation 5
- Simple Enhancement Implementation SEI2

Switch_1

Package A

Switch_2

Package B
Creating BADI in SE80

BADI Definition under Enhancement Spot

BADI Implementation (Creating Filters)
BAdI Migration (Automatic Migration)

Automatic migration by selecting utilities ➔ migrate from BAdI Builder (SE18)

- Specify Enhancement Spot for BAdI Definition
- Specify Enhancement Implementation for BAdI Implementation

➔ no special knowledge necessary
➔ effort: approximately 5 minutes per BAdI.

SE18 ➔ utilities ➔ Migrate

ONLY when you have created Custom BADI “definitions” in old systems
### Performance Comparison

<table>
<thead>
<tr>
<th>Classic BAdI</th>
<th>Migrated BAdI</th>
<th>New BAdI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2-27</td>
<td>40-600</td>
</tr>
</tbody>
</table>

x faster as classic BAdI

- 200%-2600%
- 4000%-60000%

The more implementations defined, the higher is the improvement on performance.
<table>
<thead>
<tr>
<th>Motivation &amp; Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enhancement Framework</strong></td>
</tr>
<tr>
<td>Source Code Plugin - Technology</td>
</tr>
<tr>
<td>Function Group Enhancement - Technology</td>
</tr>
<tr>
<td>Class Enhancement - Technology</td>
</tr>
<tr>
<td>Kernel-BAdI – Technology</td>
</tr>
<tr>
<td><strong>Switch Framework</strong></td>
</tr>
<tr>
<td>Summary</td>
</tr>
</tbody>
</table>
Goal of Switch Framework:

Control visibility of repository objects at runtime through switches

The Switch Framework can be used to

- Switch on industry solutions / Enterprise Add-ons
- Develop new functions without affecting existing ones
- Enhance delivered systems at partner and customer site in the context of the enhancement framework with own functions

Benefits:

- Industry Solutions are available with every release and SP without delay (i.e. timely provision of legal requirements), CRT’s* are no longer necessary for add-on systems
- Industry Solutions can be enriched by generic functions from other industries
- Synchronization of release cycles and planning

* CRT – Conflict Resolution Transport
Switchable Objects

Switchable Objects...

...by package assignment
- Appends, SI-, CI-includes for structures in DDIC
- Fixed value appends to domains
- Secondary Indexes
- Append Search Helps
- Enhancement Implementations
- Switch Business Configuration Sets (Switch BC-Sets)

...by direct assignment
- Screen elements & Flow logic
- Menu entries & functions
- IMG nodes
- Customizing
Direct Assignment: Examples

Screen Painter: Field elements

Screen Painter: Modules

Menu Painter: Function codes
Switch

Repository Object

Calculated states: ON, OFF, STANDBY

Transaction SFW1

Switch Framework
Business Function

- Represents a piece of business functionality
- Contains switches
- Transaction SFW2
Business Function Set

- Pool of business functions
- Represents e.g. one industry solution
- Max. 1 can be active
- Transaction SFW3 to create BFS
- Use Transaction SFW5 to activate a BFS.

Change System Settings

Change System Settings

<table>
<thead>
<tr>
<th>SysName</th>
<th>RH0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Function Set</td>
<td>BFS for package SEFW</td>
</tr>
</tbody>
</table>

Business Function Set

SFW3

SFW5
Switch Framework: Architecture

Most objects are switched via the assignment of their package to a switch, e.g.:
- DDIC objects (e.g. appends)
- Enhancements
- Transactions
- BC-Sets

Objects with no direct package relation are directly assigned to a switch, e.g.:
- Screen elements
- Menu entries
- IMG nodes
Summary

- The Enhancement Framework offers new possibilities to extend the SAP Standard instead of modifying it.
  - Source Code PlugIns
  - Function Group Enhancements
  - Class Enhancements
  - New BAdIs

- The new BAdIs are more flexible and faster than the classic ones.

- The Enhancements offered by Enhancement Framework and some other object types can be switched by the Switch Framework as part of a Business Function Set e.g. an industry solution.
Further Information

Help Portal

http://help.sap.com

- Documentation
- SAP Netweaver (04s)
- Application Platform
- ABAP technology
- ABAP Workbench
- Enhancement Framework

OKP / RKT Learning Maps

Internal SAP:

http://intranet.sap.com/rkt-netweaver

- Consulting
- SAP NW 04s
- Creating Business Applications using ABAP

Ramp-up customers:

Send mail to rkt@sap.com

SDN

http://sdn.sap.com
Questions?

Q&A