

Modification-Free Adaptions to SAP Programs?

Using the New SAP Enhancement Framework
They're Possible — and Here's How

March 2012

Why should you care?

- Increase your flexibility to meet customer business requirements without modifications. There is almost no requirement that can't be developed and therefore met.
- Because we are not performing MODIFICATIONS to core SAP code, Support Packs and Upgrades are much easier and less labor intensive

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

The Enhancement Paradigm

One of the advantages of SAP software is the possibility to adapt the software to your own requirements and the **possibility of keeping the adaptations during upgrade.**

Ways of adaptation:

The Enhancement Paradigm

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



The Enhancement Paradigm

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



The Enhancement Paradigm

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



The Enhancement Paradigm

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



The Enhancement Paradigm

Disadvantages of modifications

- No support for parallel developments
- Will appear much more often in adjustment tools(SPAU)
- Higher adjustment effort

The Enhancement Paradigm

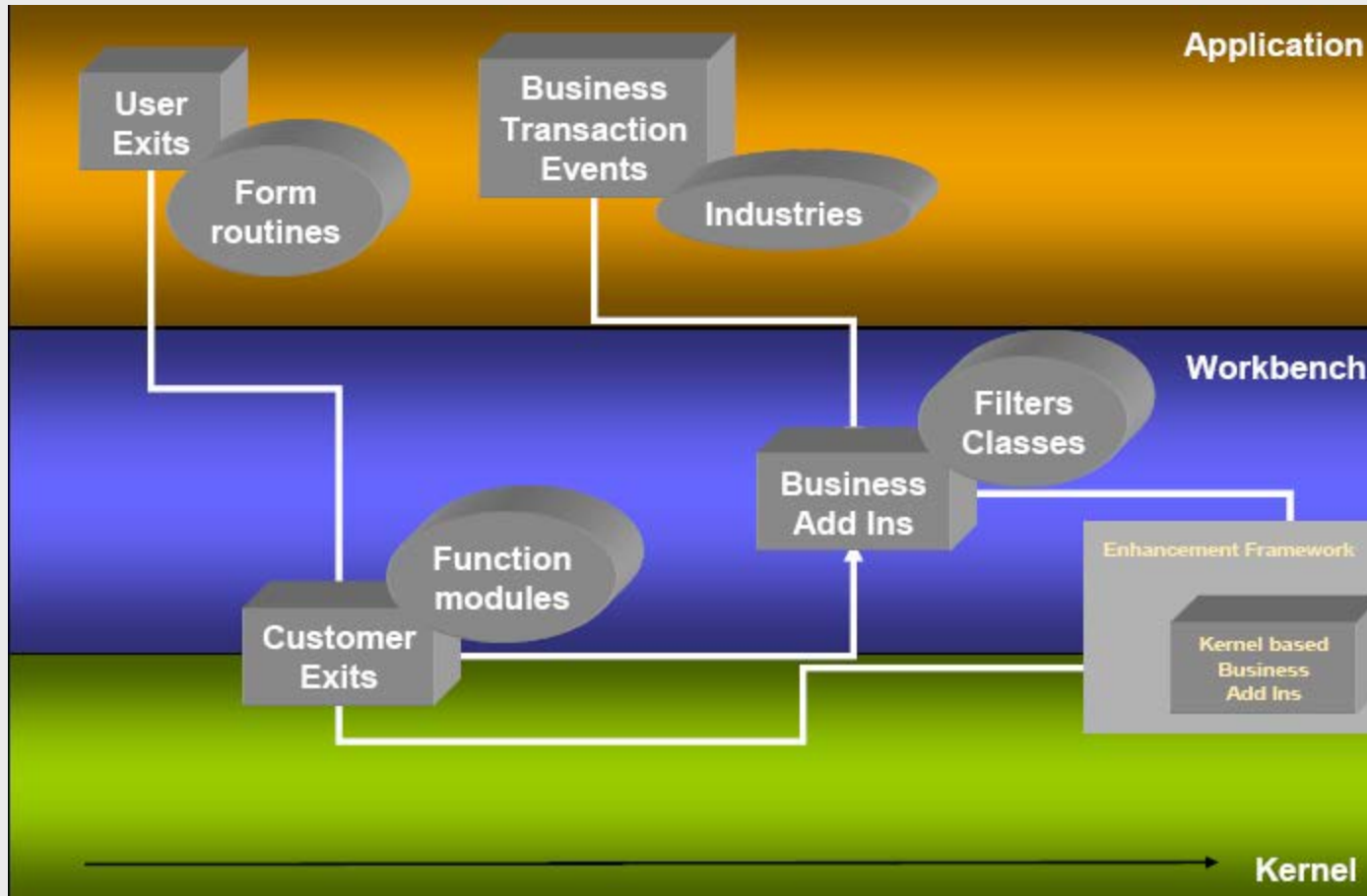
One of the advantages of SAP software is the possibility to adapt the software to own requirements and the **possibility of keeping the adaptations during upgrade.**

Ways of adaptation:

- Customizing
- Enhancement
- Modification



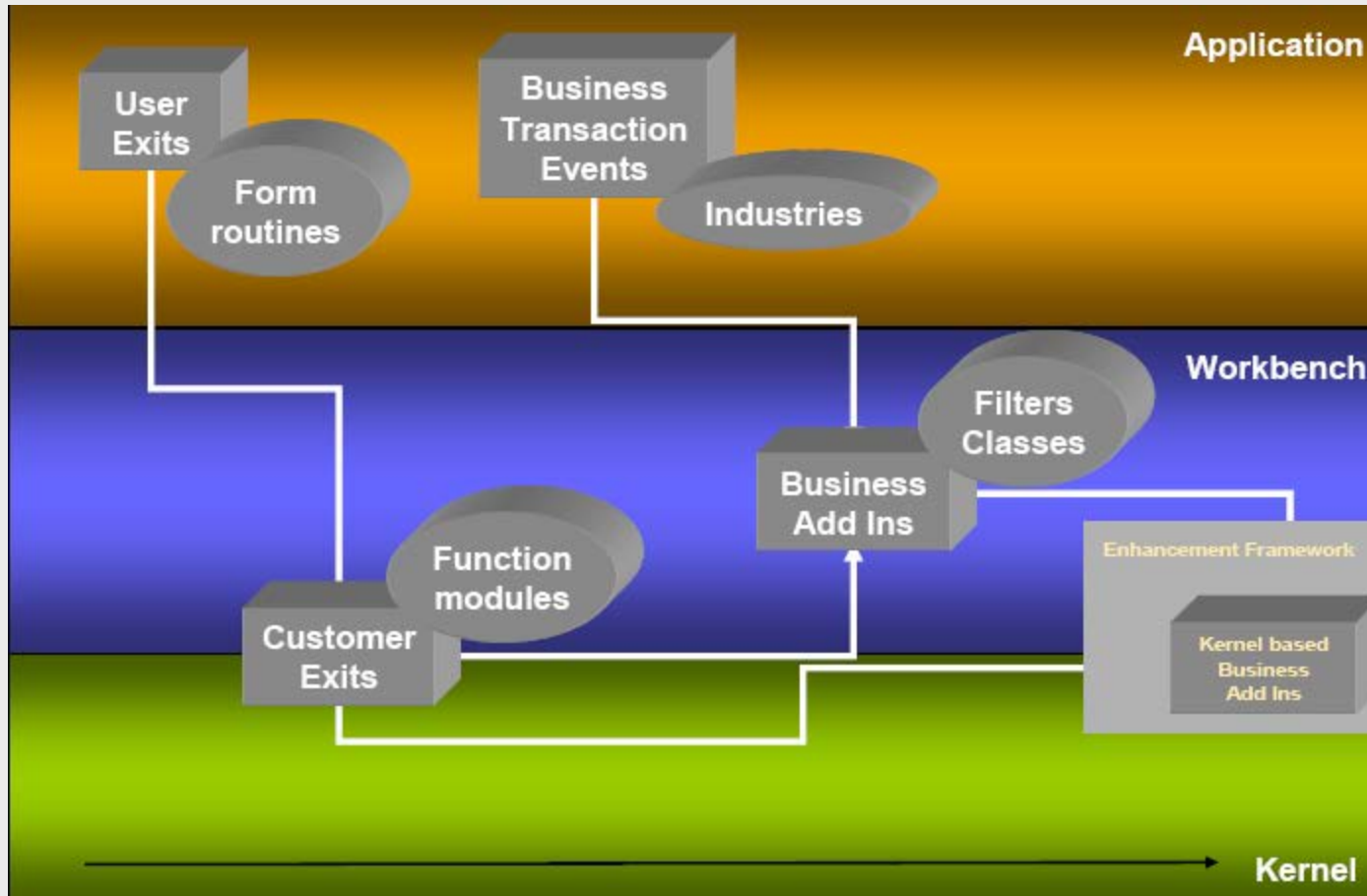
The Evolution of Adapting The Standard



Adapting The SAP Standard

‘User-Exit’ - is one of the very first mechanisms provided by SAP to execute custom code in between the standard SAP control flow. This is implemented as subroutine call (PERFORM xxx). A classical example for User-Exit is MV45AFZZ include in order processing module of SAP R/3. Though this include object doesn't fall under customer namespace, the object doesn't get overwritten during upgrade.

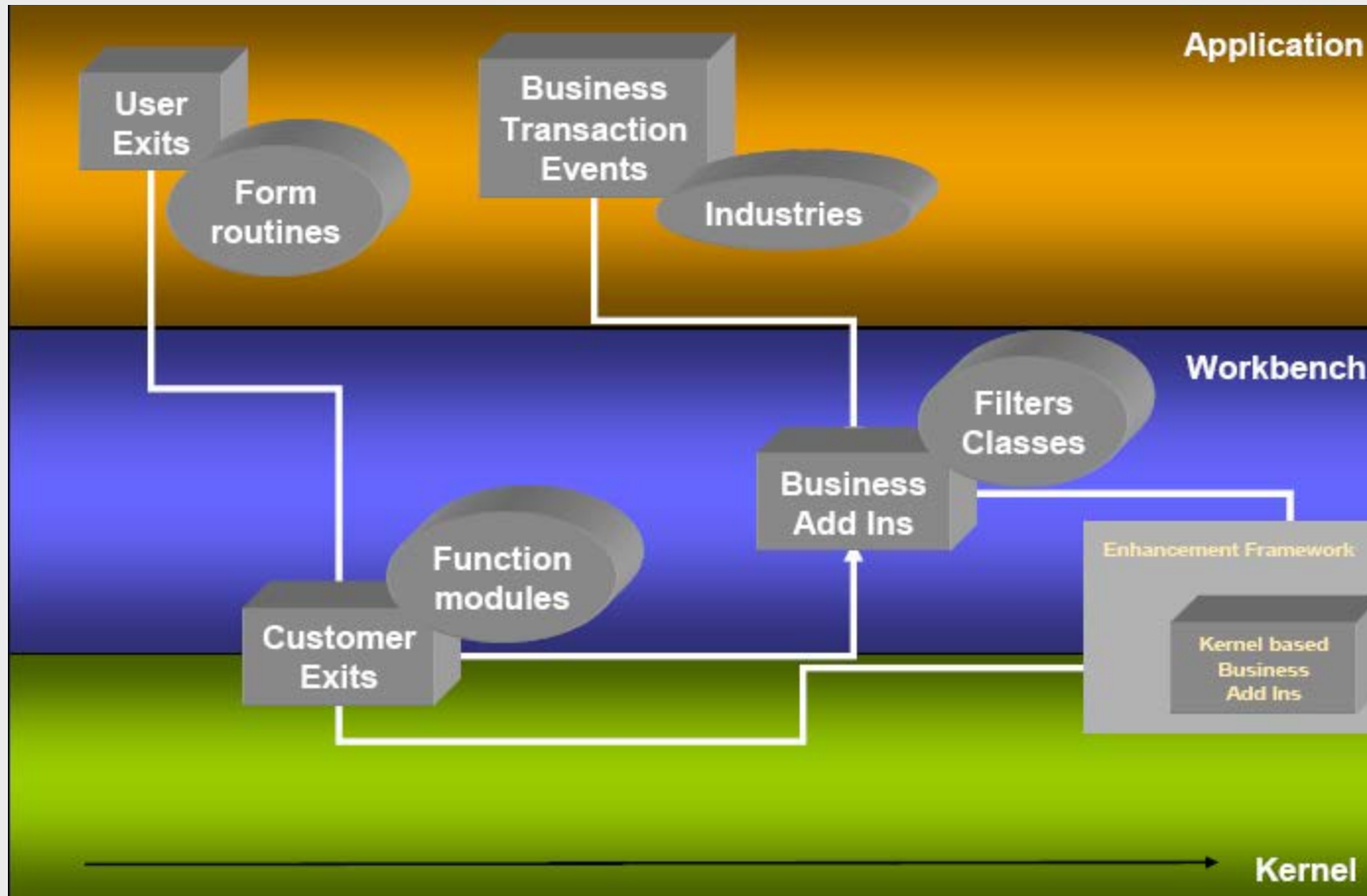
The Evolution of Adapting The Standard



Adapting The SAP Standard

‘Customer-Exit’ - is better than the user-exit, in the sense that it is implemented using Function Modules and so has a well defined parameter interface. Also since the custom coding done as part of these customer-exits is located away from the original SAP code, the maintenance is easier than user-exits.

The Evolution of Adapting The Standard



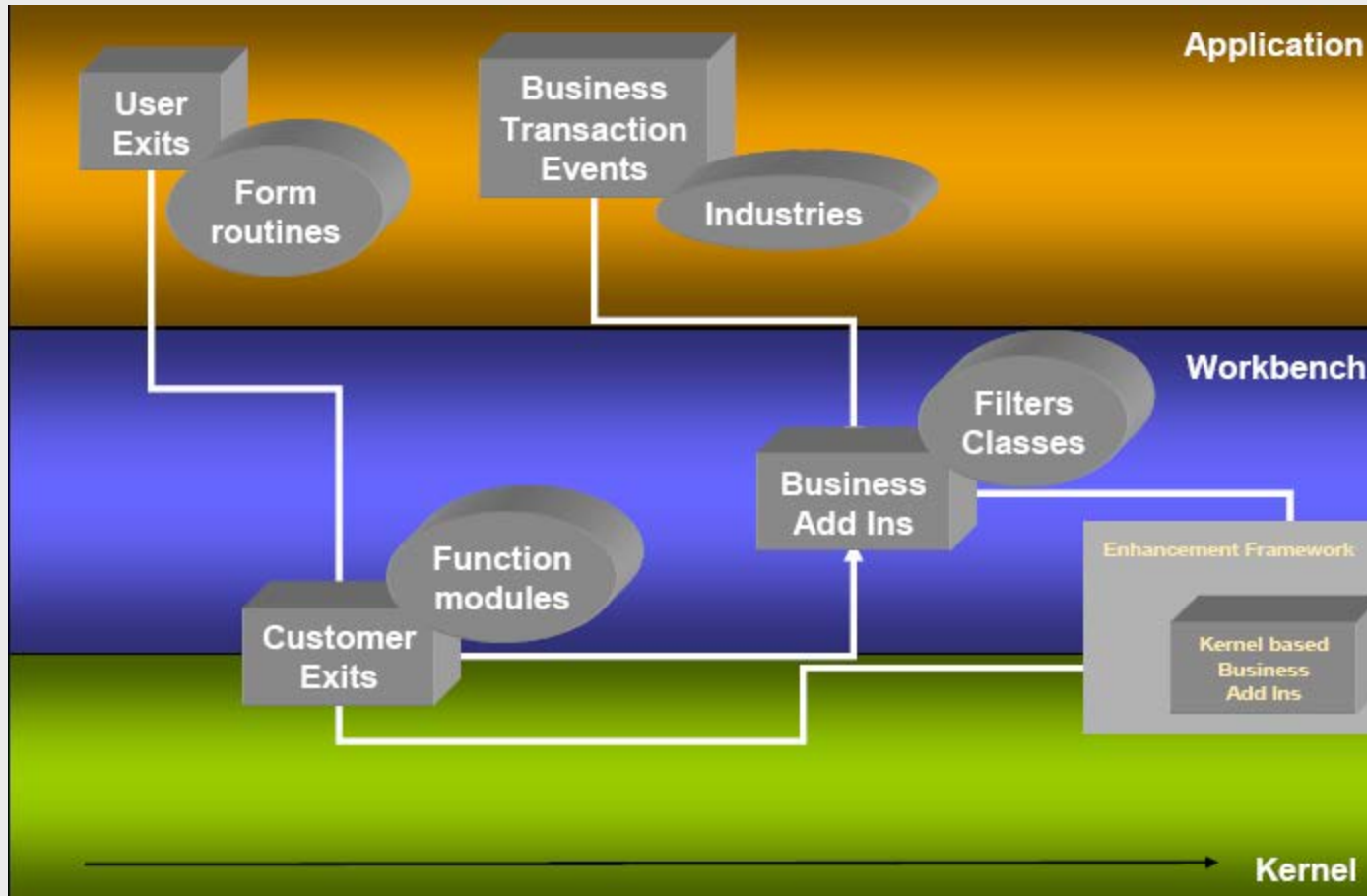
Adapting The SAP Standard

‘BTEs’ (Business Transaction Events) - This enhancement technique was developed for the Financial Accounting component (Open FI)

Application developers must define their interface in a function module, an assignment table is read in the accompanying (generated) code, and the customer function modules assigned are called dynamically

Find for Character String “OPEN_FI_PERFORM” in source code

The Evolution of Adapting The Standard



Adapting The SAP Standard

‘**BADI-s**’ (Business Add-Ins) - as they exist in pre NW04s releases are now called old classic-BADI’s. This was the first object-oriented way to enhance the ABAP system. This, to a certain extent, allows multiple implementations with limited filter support. The classic-BADI’s are implemented using ABAP Objects.

Goal of The New Enhancement Framework

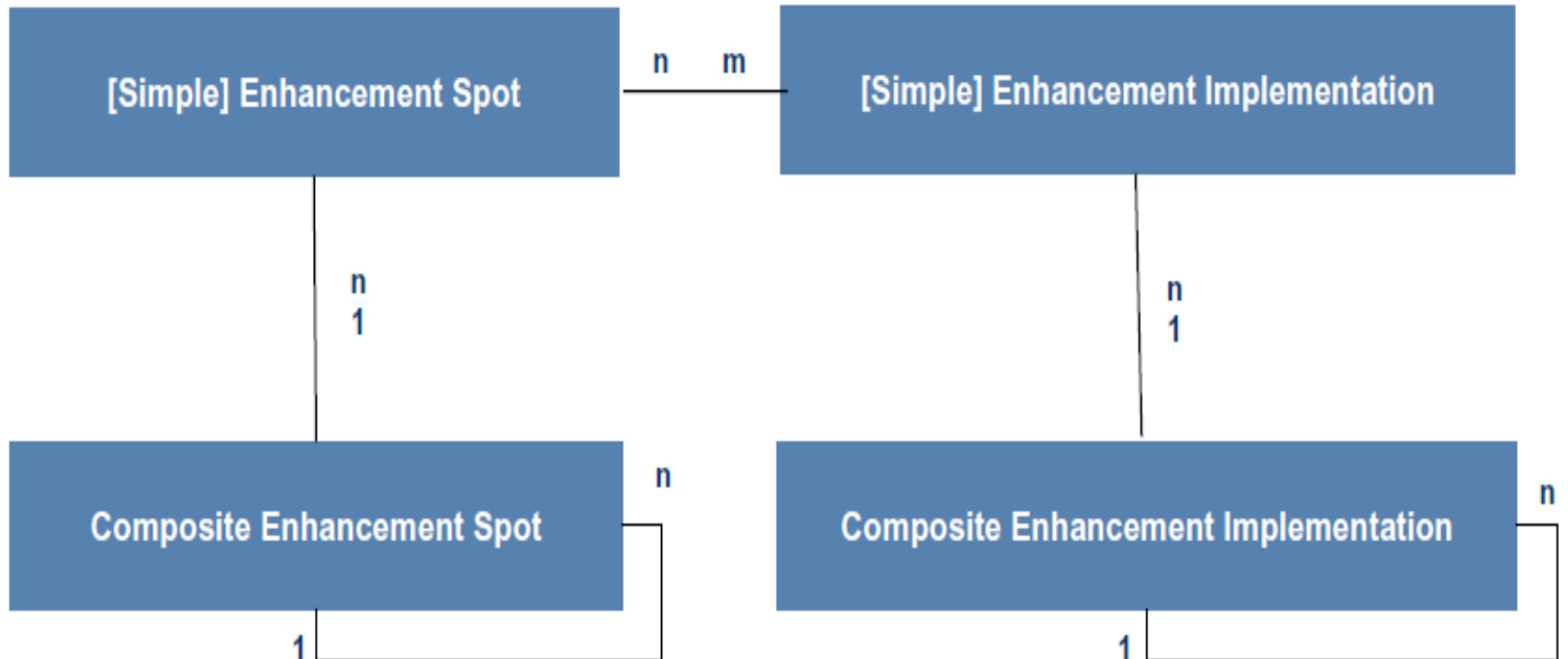
Integration of several enhancement types

- BAdIs
- Function Group Enhancement
- Class/Interface Enhancement
- Source Code Plugins
- WebDynpro Enhancement

into the Enhancement Framework

- Switchable by Switch Framework
- Enhancement Browser
- Upgrade support
- Possibility to document and group enhancements

Relationships



Terminology by Example

Option providers

Composite Enhancement Spot

[Simple] Enhancement Spot

**Enhancement Elements:
Such as a
BAdI-Definition**

Composite Enhancement Spots

- Container Objects
- Can contain
 - Other Composite Enhancement Spots
 - [Simple] Enhancement Spots

[Simple] Enhancement Spots

- Container Objects
- Can contain Enhancement Elements

Explicit Enhancement Options

- Enhancement Definitions

Terminology by Example

Option implementers

Composite Enhancement Implementation

- Container Objects
- Can contain
 - Other Composite Enhancement Implementations
 - [Simple] Enhancement Implementations

[Simple] Enhancement Implementation

- Can contain Enhancement Implementation Elements

Enhancement Implementation Elements

- Implementations

Composite Enhancement Implementation

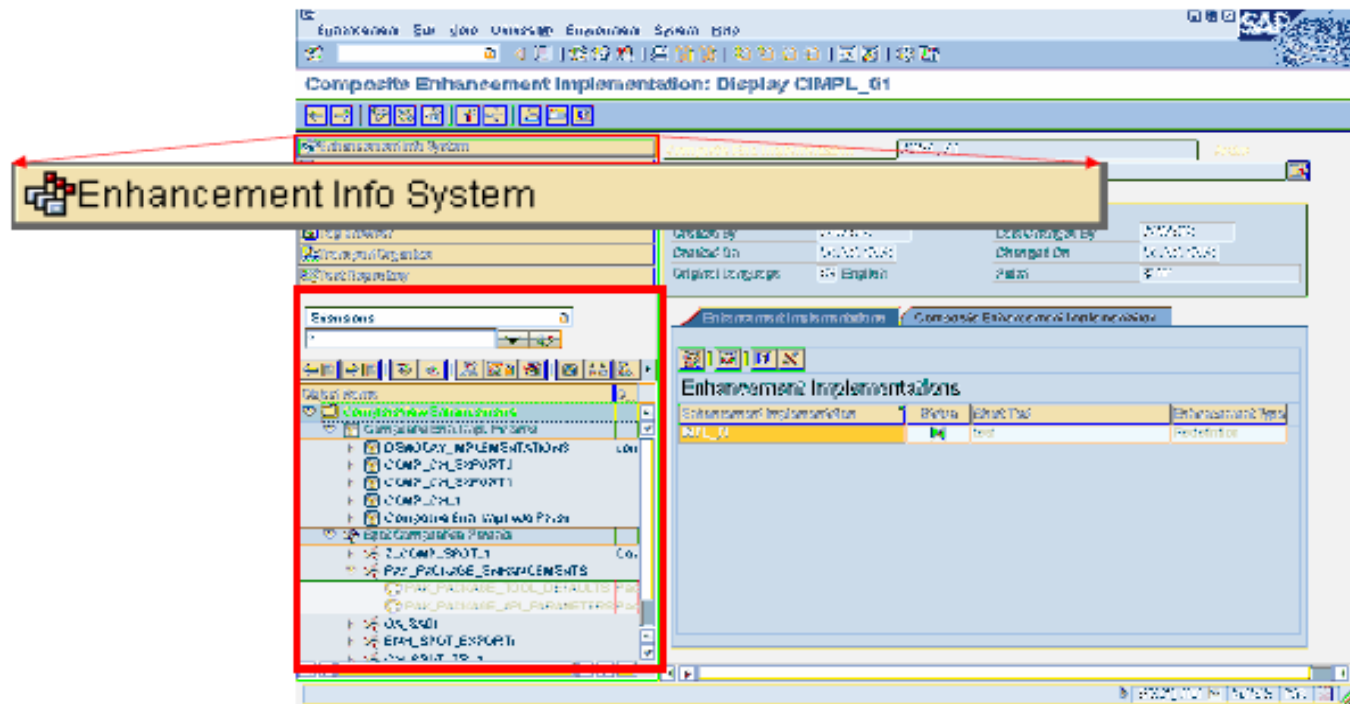
[Simple] Enhancement Implementation

**Enhancement Implementation Elements:
Such as a
BAdI-Implementation**

Enhancement Browser

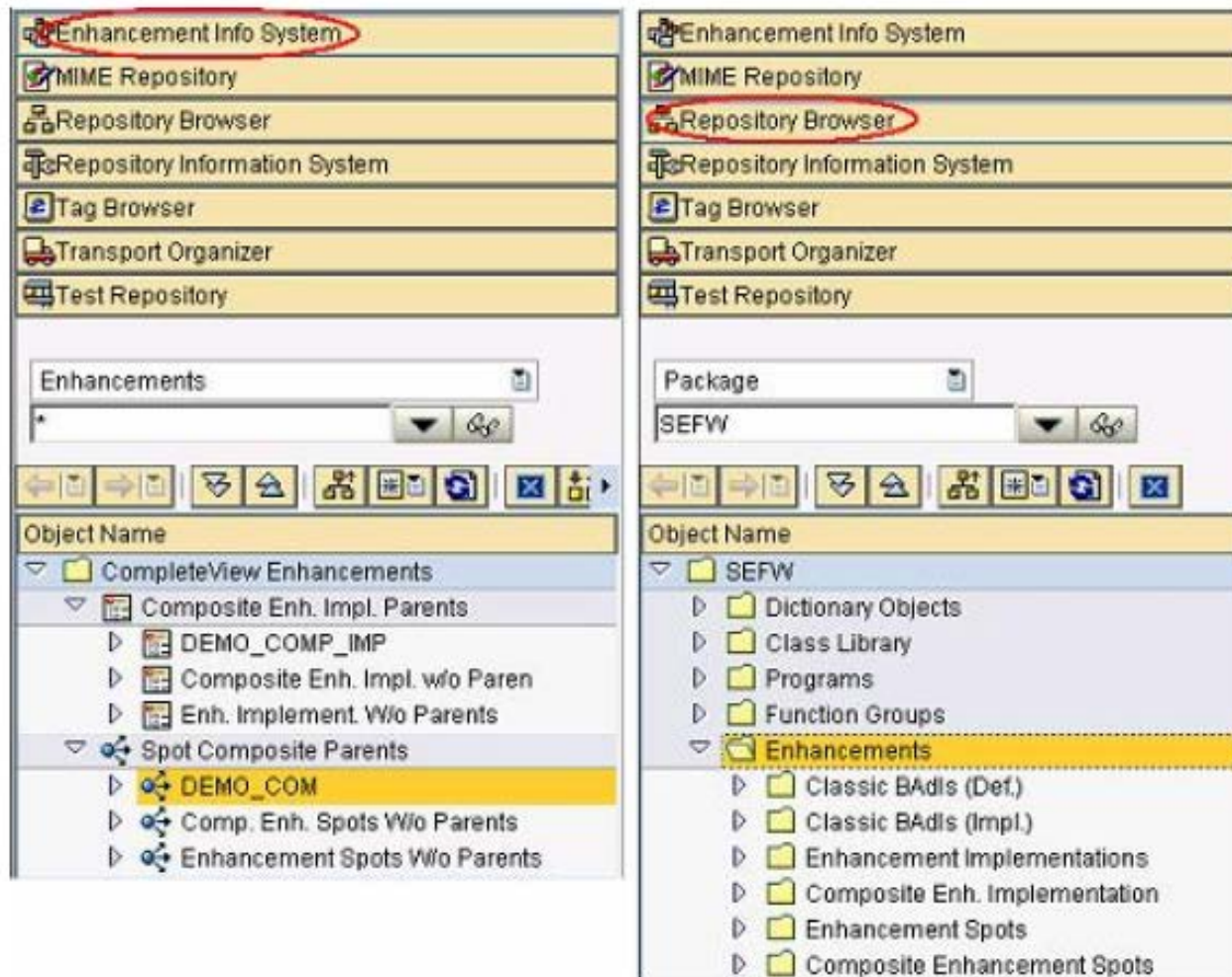
Search for

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



Integrated into Object Navigator (SE80)

Enhancement Browser



Enhancement browser integrated into SE80

Content

- Enhancement Framework Overview
- Source Code Plugin Technology
- Function Group Enhancement Technology
- Class Enhancement Technology
- BAdi Technology
- Switch Framework
- Key Learning
- ROI / TCO
- Best Practices

Source Code Enhancement

Explicit Enhancement Option

- Predefined enhancement options can be defined in source code.

Implicit Enhancement Option


- At common enhancement places, implicit Enhancement Options are available.

- Examples:


- Beginning/End of Include
- Beginning/End of Method/Function Module/Form Routine
- End of a structure
- End of Private/Protected/Public Section of a local class

Editor Modes For Enhancements

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

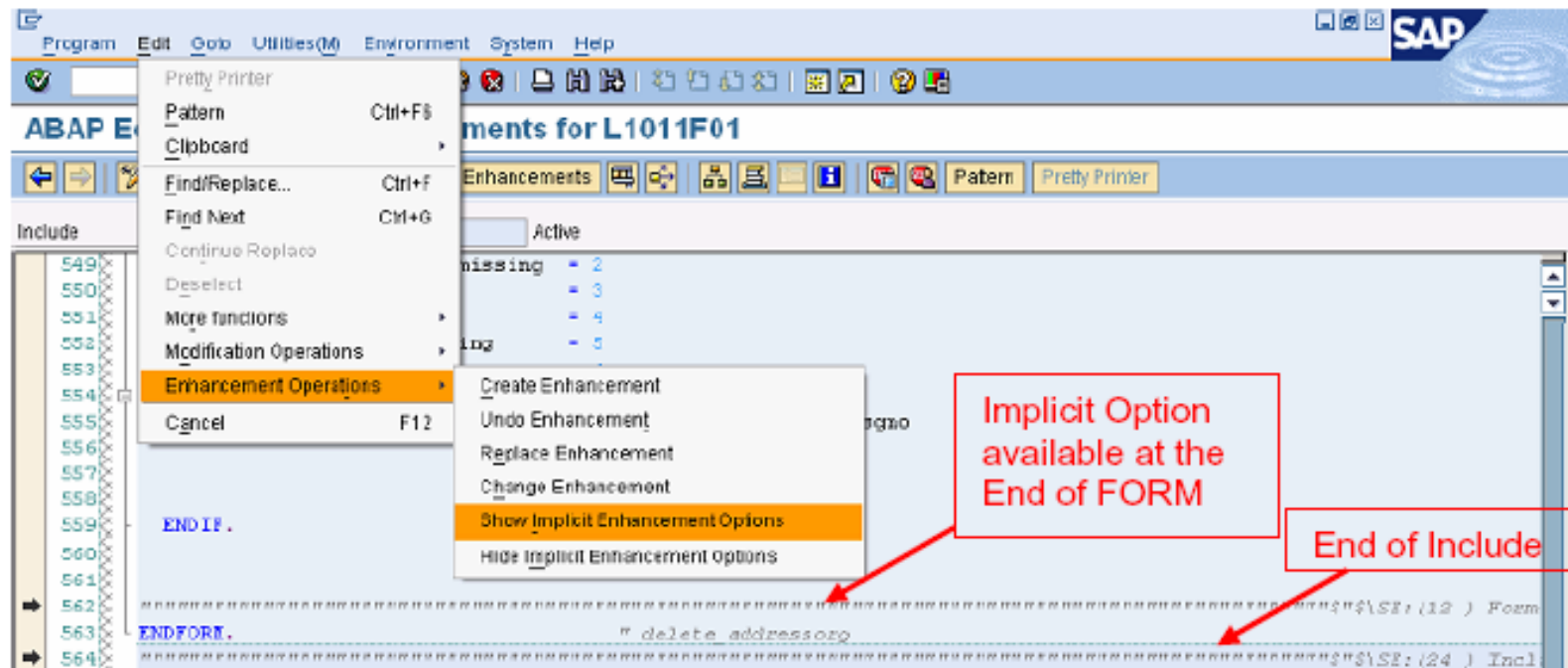
- use button  „Display <-> Change“ to switch to change mode.

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

- use button „**Change Enhancements**“  to switch to Enhancement mode
- use button „Display <-> Change“  to leave Enhancement mode

Implicit Enhancement Options

To view all the implicit options available in a source code, choose 'Edit -> Enhancement Operations -> Show Implicit Enhancement Options' from the editor.



Implicit Source Code Enhancement Options

You don't need to have an explicitly defined enhancement spot in order to implement these enhancements. Just position the cursor on any of these implicit points and choose 'Create Enhancement' from the menu to implement it.

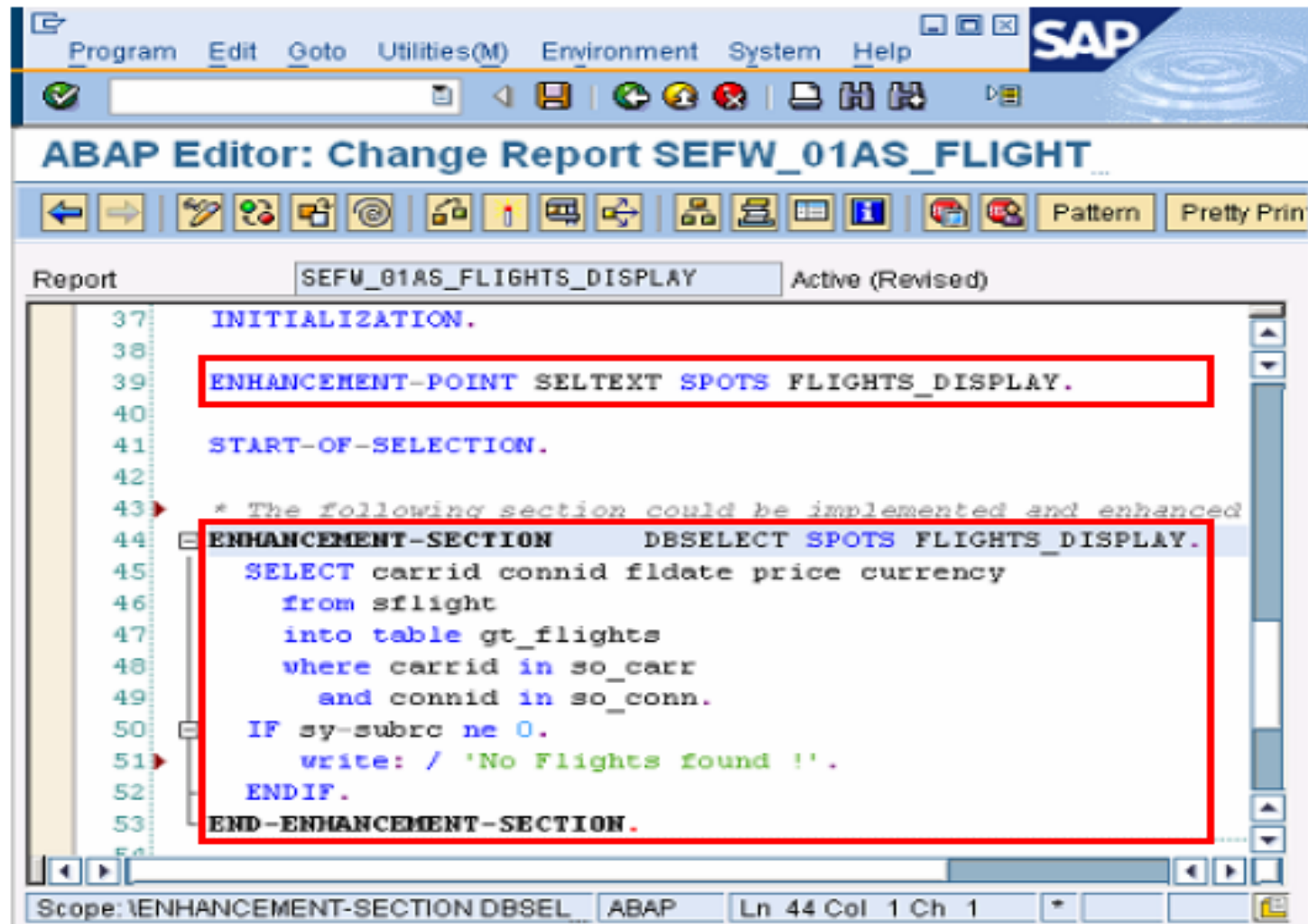
Explicit Enhancement Options

There are two types of Explicit Enhancement options available. One which can be provided at a specific place - called Enhancement Point, and another which can be used to replace a set of statements – called Enhancement Section. For this, we now have two new ABAP statements.

- ENHANCEMENT-POINT
- ENHANCEMENT-SECTION

When the Enhancement-Section is implemented, only the implementation gets executed and the original code doesn't get executed. This is a new technique, which didn't exist previously in any of the old ways of enhancing, to exclude any standard SAP code from execution. Because of this, there can be only one active implementation of an Enhancement-Section. On the other hand, there can be multiple active implementations of an Enhancement-Point, in which case all the implementations will be executed with no guarantee in the order of execution

Explicit Enhancements



```
ABAP Editor: Change Report SEFW_01AS_FLIGHT...

Report: SEFW_01AS_FLIGHTS_DISPLAY Active (Revised)

37  INITIALIZATION.
38
39  ENHANCEMENT-POINT SELTEXT SPOTS FLIGHTS_DISPLAY.
40
41  START-OF-SELECTION.
42
43  * The following section could be implemented and enhanced
44  ENHANCEMENT-SECTION DBSELECT SPOTS FLIGHTS_DISPLAY.
45      SELECT carrid connid fldate price currency
46      from sflight
47      into table gt_flights
48      where carrid in so_carr
49      and connid in so_conn.
50  IF sy-subrc ne 0.
51      write: / 'No Flights found !'.
52  ENDIF.
53  END-ENHANCEMENT-SECTION.
```

Scope: \ENHANCEMENT-SECTION DBSEL ABAP Ln 44 Col 1 Ch 1

Explicit Source Code Enhancement Options

Source Code Plugin – Example

```
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.

Remember!!



When the Enhancement-Section is implemented, **only the implementation gets executed and the original code doesn't get executed.. Because of this, there can be only one active implementation of an Enhancement-Section.**

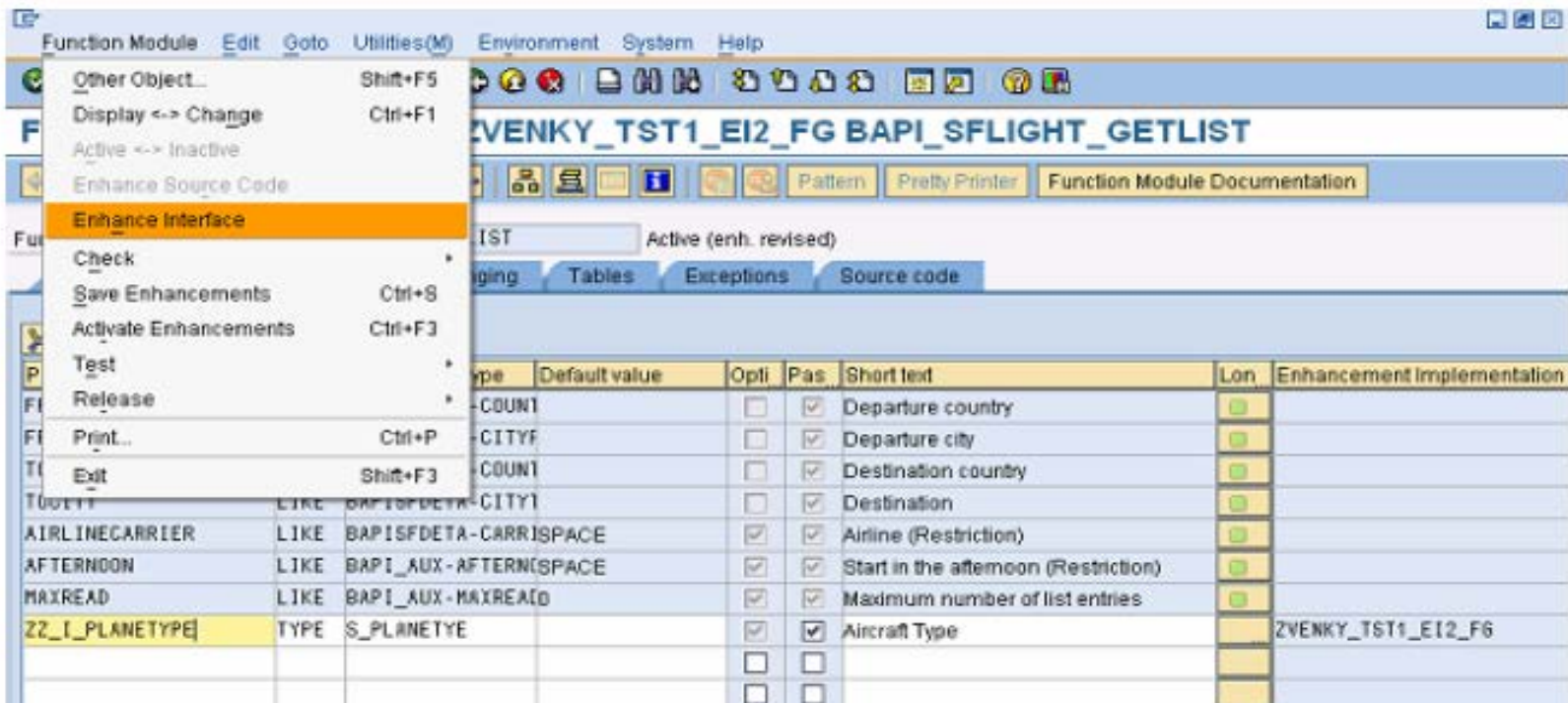
On the other hand, there can be multiple active implementations of an Enhancement-Point, in which case all the implementations will **be executed with no guarantee in the order of execution**

Content

- Enhancement Framework Overview
- Source Code Plugin Technology
- Function Group Enhancement Technology
- Class Enhancement Technology
- BAdi Technology
- Switch Framework
- Key Learning
- ROI / TCO
- Best Practices

Function Group Enhancements

All application function modules can be enhanced by adding parameters to the standard function module interface. These parameters must be 'optional' in nature, since adding a mandatory parameter will require all calls to be changed. Additionally any function module that is part of the Central Basis can not be enhanced (for example: function module 'POPUP_TO_CONFIRM'). From the menu, choose 'Function module -> Enhance interface' to add optional parameters to a function module.



The screenshot shows the SAP Function Builder interface. The menu 'Function Module -> Enhance Interface' is open, highlighting the 'Enhance Interface' option. The function module being enhanced is 'ZVENKY_TST1_EI2_FG BAPI_SFLIGHT_GETLIST'. The 'Source code' tab is selected, displaying a table of parameters for enhancement.

Type	Default value	Opti	Pas	Short text	Lon	Enhancement Implementation
COUNT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Departure country	<input type="checkbox"/>	
CITY		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Departure city	<input type="checkbox"/>	
COUNT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Destination country	<input type="checkbox"/>	
CITY		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Destination	<input type="checkbox"/>	
LIKE	BAPI_SFLIGHT-CARRISPACE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Airline (Restriction)	<input type="checkbox"/>	
LIKE	BAPI_AUX-AFTERNOON	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Start in the afternoon (Restriction)	<input type="checkbox"/>	
LIKE	BAPI_AUX-MAXREAD	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Maximum number of list entries	<input type="checkbox"/>	
TYPE	S_PLANETYPE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Aircraft Type	<input type="checkbox"/>	ZVENKY_TST1_EI2_FG

Function Group Enhancements



Think about all the steps that customers currently go through for adding new parameters to a BAPI. Cloning of BAPI is an option but maintaining and syncing up during upgrade could be a nightmare. Now with a combination of Function Group enhancement and Source code plug-in, BAPI's can easily be enhanced to add new parameters and add custom logic to process those parameters.

Content

- Enhancement Framework Overview
- Source Code Plugin Technology
- Function Group Enhancement Technology
- Class Enhancement Technology
- BAdi Technology
- Switch Framework
- Key Learning
- ROI / TCO
- Best Practices

Class / Interface Enhancements

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 Methods & Parameters

Choose menu option 'Class -> Enhance' to add new methods or parameters.

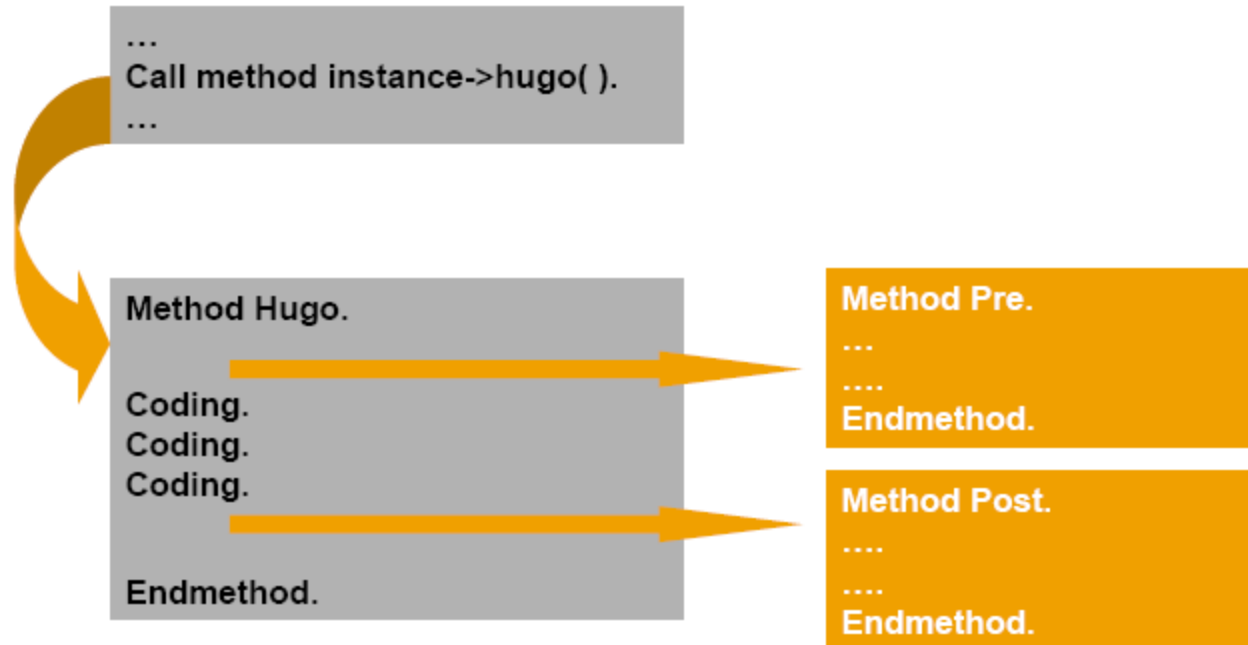
Adding new methods

Method	Level	Visibility	M...	Description	PreExit	PostExit	Enhancement
AFTER_IMPORT	Static	Public					
UPDATE	Static	Public					
READ	Static	Public					
WRITE	Static	Public					
ADD_METHOD_DNE	Instan	Private		additional functionality public			MATECHED2005_001
ADD_METHOD_PRIVATE	Instan	Private		additional functionality private			MATECHED2005_001

Adding optional parameters to existing methods

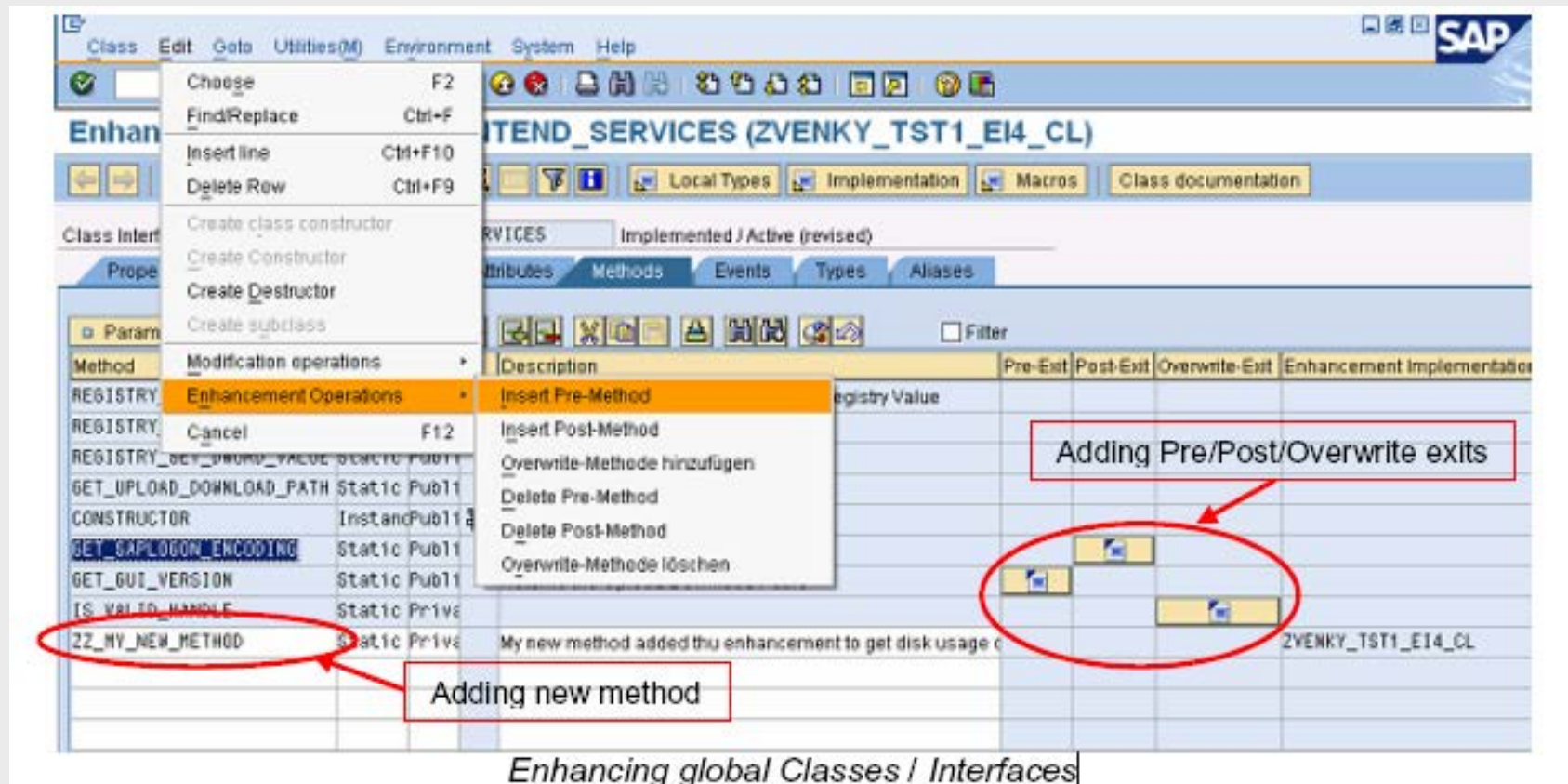
Method parameters AFTER_IMPORT								
Parameter	Type	P...	O...	Typing M...	Associated Type	Default value	Description	Enhancement
OBJ_NAME	Importin	<input type="checkbox"/>	<input type="checkbox"/>	Type	TROBJ_NAME		Object Name in Object Directory	
PROTOCOL	Changin	<input type="checkbox"/>	<input type="checkbox"/>	Type	SPROT_U_TAB		Table Type for SPROT_U (Log In	
NY_ADD_PARAM	Importin	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Type	CHAR10		Characterfeld der Länge 10	MATECHED2005_001
NY_ADD_PARAM_EXP	Exportin	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	INT4		Natürliche Zahl	MATECHED2005_001

Pre/Post Methods



Pre/Post Exits

Choose menu option 'Edit -> Enhancement operations' to add or delete the Pre/Post/Overwrite exit methods.



The screenshot shows the SAP IDE interface for editing a class. The 'Edit' menu is open, and the 'Enhancement operations' option is selected. The 'ZVENKY_TST1_EI4_CL' class is open, and the 'Enhancement Operations' menu is also open, showing options like 'Insert Pre-Method', 'Insert Post-Method', 'Overwrite-Method hinzufügen', 'Delete Pre-Method', 'Delete Post-Method', and 'Overwrite-Method löschen'. The 'ZVENKY_TST1_EI4_CL' class is highlighted in the table. A red circle highlights the 'ZVENKY_TST1_EI4_CL' class in the table. A red circle highlights the 'ZVENKY_TST1_EI4_CL' class in the table. A red circle highlights the 'ZVENKY_TST1_EI4_CL' class in the table.

Adding new method

Adding Pre/Post/Overwrite exits

Enhancing global Classes / Interfaces

Content

- Enhancement Framework Overview
- Source Code Plugin Technology
- Function Group Enhancement Technology
- Class Enhancement Technology
- BAdi Technology
- Switch Framework
- Key Learning
- ROI / TCO
- Best Practices

What is a Business Add In

A BAdI

- is an anticipated point of extension—these points act like sockets and exist in the original source code
- is a predefined anchor for an Object Plugin
- has a well-defined interface in contrast to source code plugins and is therefore more stable to changes in the original source code

Comparison: Usage of Classic BAdi vs. New BAdi

With Classic BAdi

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

```
CALL METHOD c1_exithandler=>  
get_instance  
  EXPORTING  
    exit_name = `BADI_NAME`  
  CHANGING  
    instance = bd.
```

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

selecting implementations and issuing calls is mixed

calls are redirected over a proxy class

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

Implementations are called directly (without a proxy)

Comparison: Usage of Classic BAdi vs. New BAdi

- The new BAdi evaluates as much information as possible during compile time.
- Better Performance/Lower Memory consumption
- Database access only at compile time
- Statically typed comparisons at runtime
- Internal handle-class integrated in SAP Kernel
- **40-600 times faster than Classic BAdis**

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

Switch Framework Goals & Benefits

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

Switchable Objects

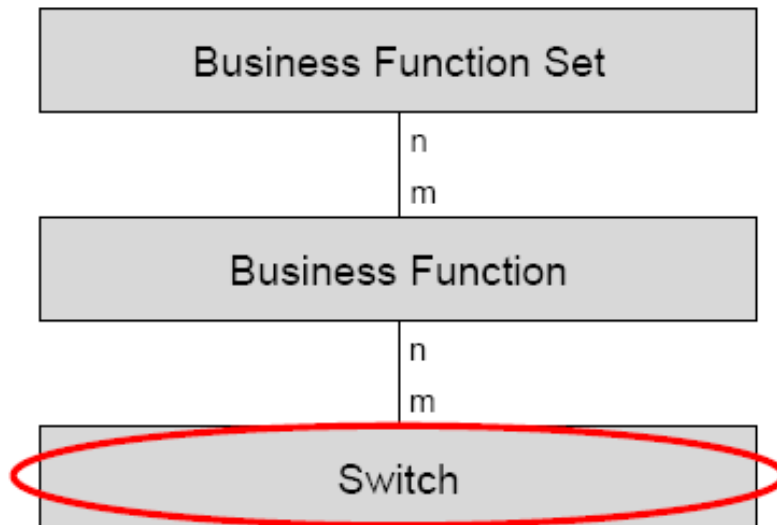
...by package assignment

- 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

Switch



Switch

- Repository Object
- Calculated states: ON, OFF, STANDBY
- Transaction **SFW1**

Switch Framework

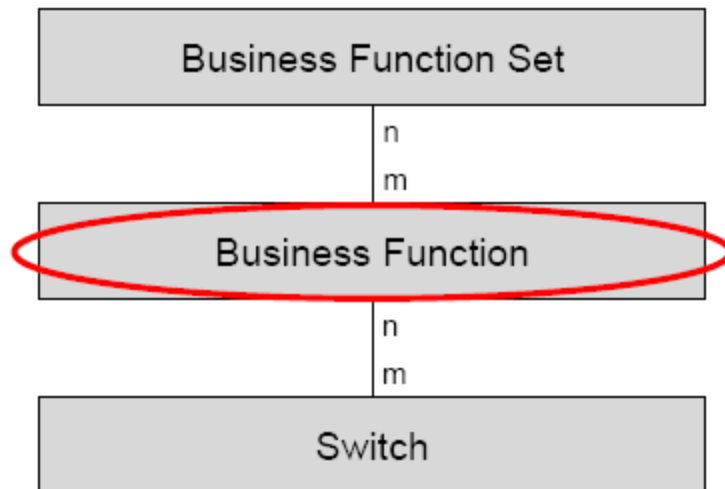
Switch Definition

Switch	Global Status	Short Text	Combi	CDIC-relevant
DECOMO	off		<input type="checkbox"/>	<input type="checkbox"/>
SEFW_SWITCH_2	on		<input type="checkbox"/>	<input type="checkbox"/>
SEFW_SWITCH_3	on		<input checked="" type="checkbox"/>	<input type="checkbox"/>
SHOWEXTENH	off	Display Implementation	<input type="checkbox"/>	<input type="checkbox"/>
STRAMELAG_APP1	on	STRAMELAG Append 1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
SWITCH_1	off		<input type="checkbox"/>	<input type="checkbox"/>

Packages

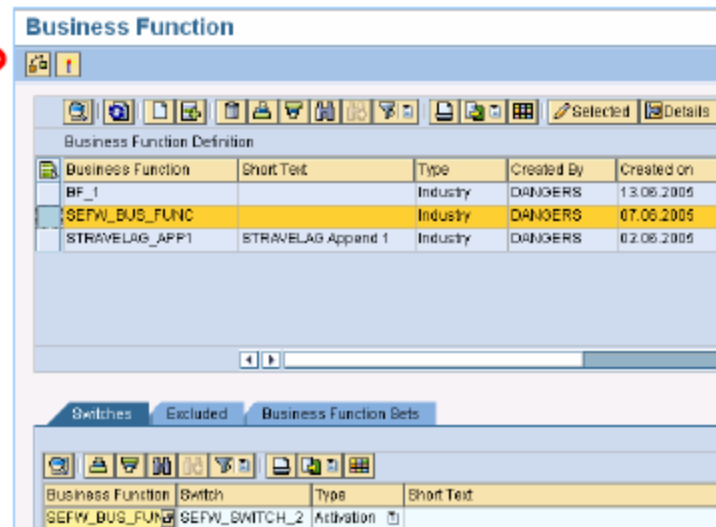
Switch	Package	Short Description
SEFW_SWITCH_2	SEFW_SWITCHED_2	

Business Function



Business Function

- Represents a piece of business functionality
- Contains switches
- Transaction **SFW2**



Business Function

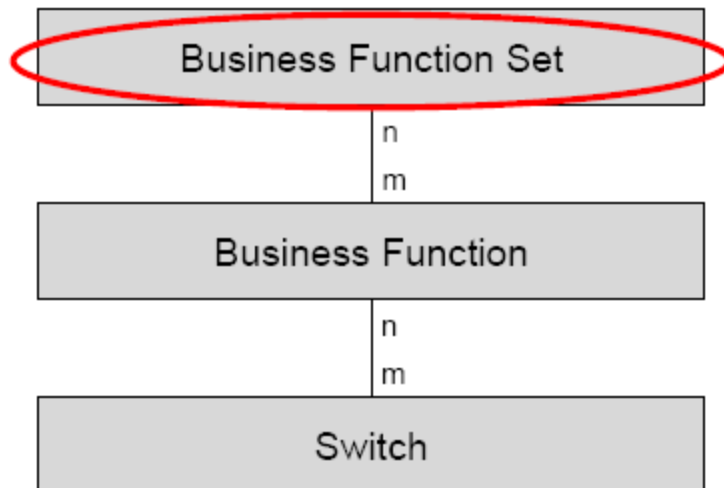
Business Function Definition

Business Function	Short Text	Type	Created By	Created on
BF_1		Industry	DANGERS	13.06.2005
SEFW_BUS_FUNC		Industry	DANGERS	07.06.2005
STRAVELAG_APP1	STRAVELAG Append 1	Industry	DANGERS	02.06.2005

Switches Excluded Business Function Sets

Business Function	Switch	Type	Short Text
SEFW_BUS_FUNC	SEFW_SWITCH_2	Activation	

Business Function Set



Change System Settings

SystemName: R&D
Business Function Set: BFS for package SEFW Active

BFS for package SEFW Enterprise AddOns

Business Function	Short Text	Status	Activated on	Active
SEFW_BUS_FUNC		On	16.12.2005 05:26:24	Active
STRAVELAO_APP1	STRAVELAO Append 1	On	16.12.2005 05:26:24	Active

SFW3

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.

Business Function Set

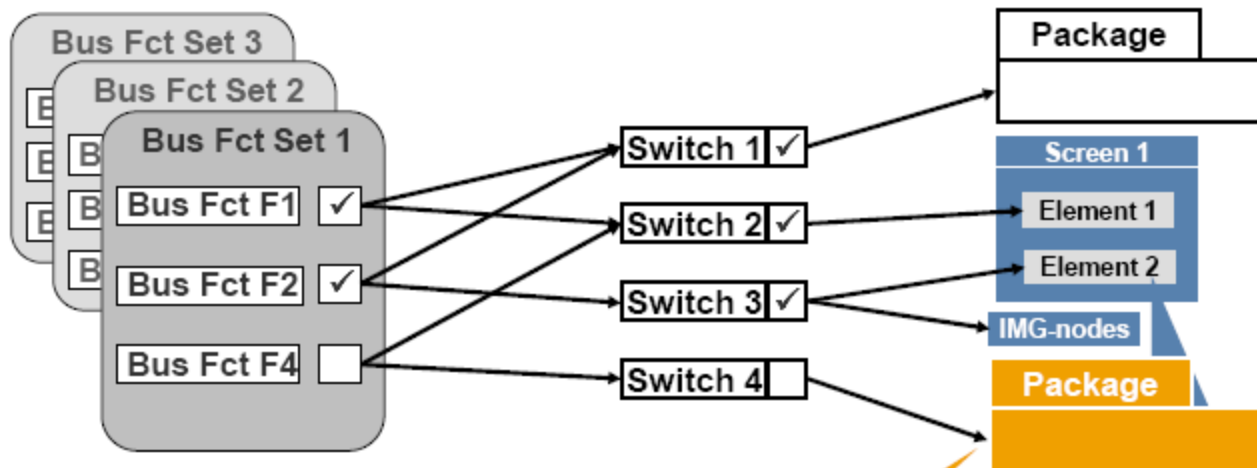
Business Function Set	Short Text	Created By	Created on
SEFW_BFS	BFS for package SEFW	STRAVELAO Append 1	16.12.2005 05:26:24

Business Functions Resolution Control BAPI Level Control

Bus. Func. Set	Business Function	No Display	Always on	Upgrade	Short Text
SEFW_BFS	SEFW_BUS_FUNC	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
SEFW_BFS	STRAVELAO_APP1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	STRAVELAO Append 1

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

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

Key Learning's

- 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.

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

ROI / TCO

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

Content

- ➔ Enhancement Framework Overview
- ➔ Source Code Plugin Technology
- ➔ Function Group Enhancement Technology
- ➔ Class Enhancement Technology
- ➔ BAdi Technology
- ➔ Switch Framework
- ➔ Key Learning
- ➔ ROI / TCO
- ➔ Best Practices

Best Practices

“With great power comes great responsibility”!

So, choose the options wisely. My best-practice recommendation for the order in which the enhancement options should be considered and used is:

- Use a BADI or Customer Exit; if there is no BADI or Customer Exit to suite your need then,
 - Try to solve it using Explicit Source code, Function and Class enhancements; and,
 - Implicit Source code enhancement should be the last option to choose

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>

Thank You & Company Info.

- ☐ In Business Since 1993
- ☐ Women Owned
- ☐ Small Business Certified
- ☐ GSA IT 70 Schedule.



LIKE US ON FACEBOOK - <http://www.facebook.com/itpsapinc>

Visit us at: WWW.ITPSAP.COM

Copyright 2012 , All Rights Reserved



Experience Matters....

- No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of IT Partners Inc.. The information
- contained herein may be changed without prior notice.
- Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.
- Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
- IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP,
- Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation.
- Oracle is a registered trademark of Oracle Corporation.
- UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.
- Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.
- HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.
- Java is a registered trademark of Sun Microsystems, Inc.
- JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.
- MaxDB is a trademark of MySQL AB, Sweden.
- SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only.
- IT Partners Inc. assumes no responsibility for errors or omissions in this document. IT Partners Inc. does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material.
- This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.
- IT Partners Inc. shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.
- The statutory liability for personal injury and defective products is not affected. IT Partners Inc. has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.