Enhancement Framework

By Asit Garg

Introduction

From release 4.7 onwards (Release 7.0 of the SAP NetWeaver Application Server ABAP (SAP NetWeaver 2004s)), SAP has come up with Enhancement framework. **In future Enhancement Framework will going to replace or incorporate the existing enhancement and modification concepts to unify all possible ways of modifying or enhancing SAP products.** This means all our Customer Exit, Function Module Exit, BAdis (Classical BAdis in Enhancement Framework concept) will be unified into single framework.

With enhancement framework SAP has come up with a modification-free enhancement technology, enhancing source code units without modifying them.

Terminology

Before going forward, let's know some of the terminologies which are being used in Enhancement Framework.

Enhancement options: Predefined positions in Repository objects where we can attach our own code using enhancement implementation i.e. its like hooks where we can attach the enhancements. These options are either explicitly defined or exist implicitly.

Source Code Plug-in: It refers to the implementation of an enhancement option.

Enhancement Implementation Element: The enhancement which is attached to an enhancement option.

Enhancement Spot: A repository object which acts like a container of the explicitly created enhancement options. **It can be view in SE80 under enhancement Info system**

Composite Enhancement Spot: Provides a semantic grouping of simple and other composite enhancement spots so as to administer all the similar enhancement spots easily.

How it works

The Enhancement Framework works on enhancement option implementation.

These implementations are part of the development objects which can be enhanced and transported.
We attach a source code plug-in(enhancement implementation) to the predefined enhancement options. At runtime these source code plug-ins behaves as if it belonged to the development object it enhances, while the enhancement as a transport object does not belong to the enhanced object. This means, the source code plug-in behaves at runtime as if it were part of the standard SAP program while in fact it can belong to another package.

Types:

There are two types of enhancements which come under enhancement framework:

1. **Source Code Enhancement**
2. **Enhancement using BADIs**

This document only talks about source code enhancement.

There are 2 types of source code enhancement:

- **Implicit Enhancement Option**
- **Explicit Enhancement Option**

Each one is explained in detail one by one.

1. Implicit Enhancement Option:

Implicit enhancement options exist in the following places in ABAP programs:

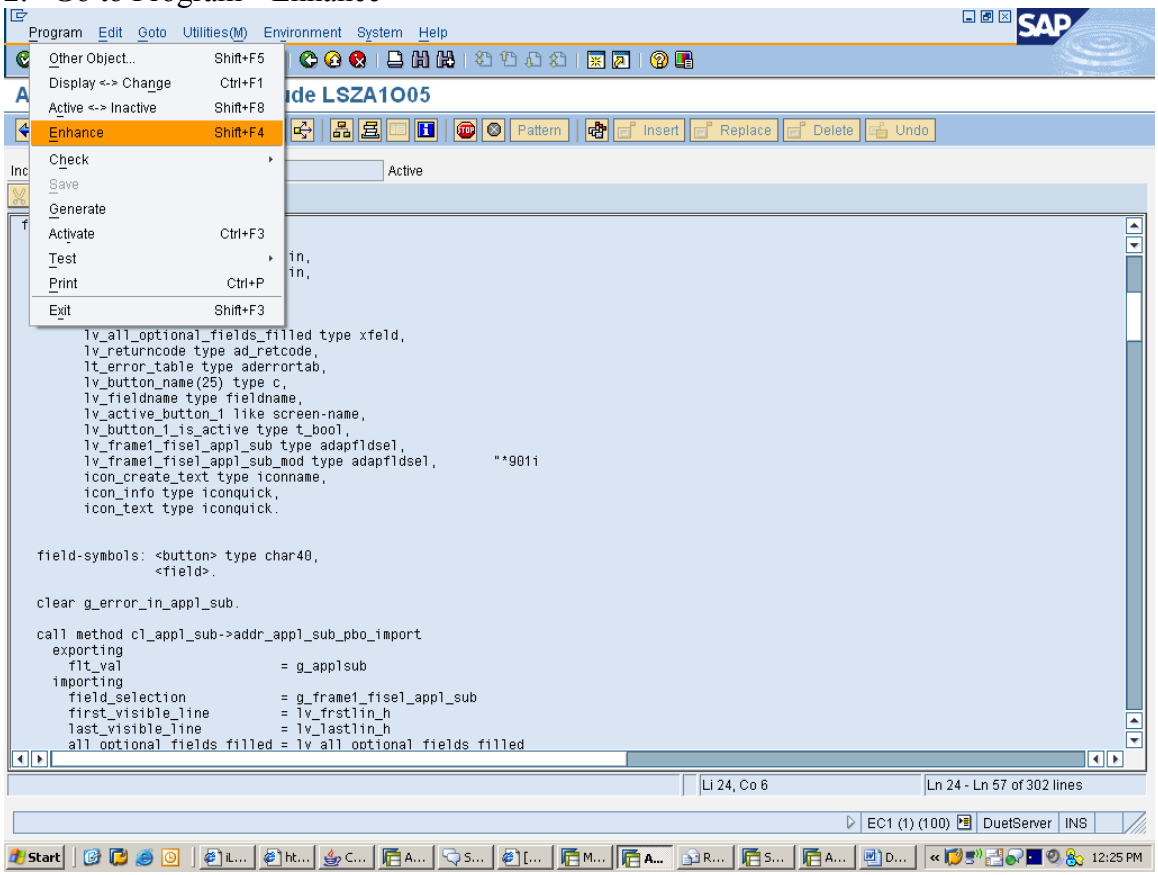
- After the last line of the source code of executable programs, function groups, module pools, subroutine pools, and Include programs.
- Before the first and after the last line of the implementation of a procedure (after the opening statement and before the END statement).
- At the end of a visibility section in the declaration section of a local class.
- At the end of a list of formal parameters of the same type in the declaration of local methods.
- Interface of the standard FM.

Implicit enhancement does not require enhancement spots. They are also enhanced by enhancement implementations.

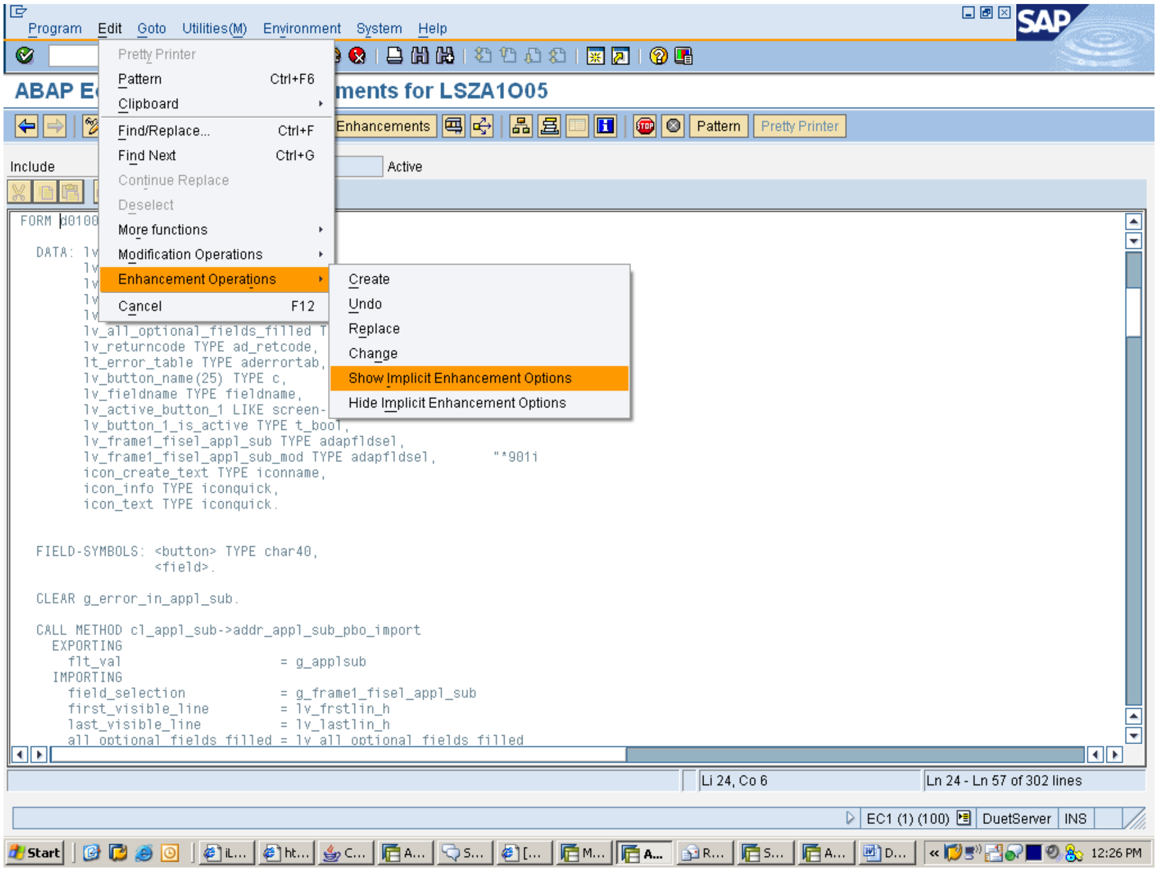
Implicit means we can plug in your code at the start and end of subroutines, performs, programs, includes etc.This doesn't means that all the programs/includes etc can be enhanced.
We need to first check where all we can use implicit enhancement.

For that follow the following steps:

1. Open a program, include which you want to enhance in se38
2. Go to Program->Enhance

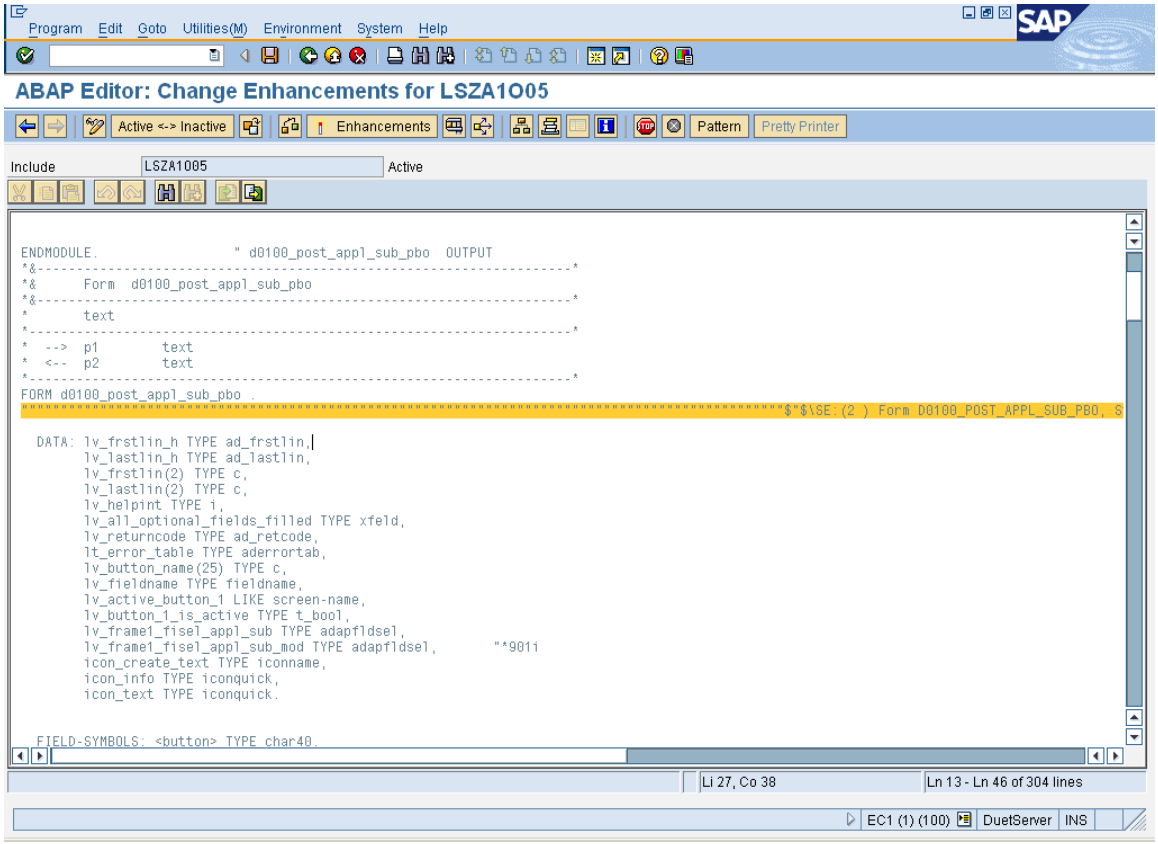


3. After that go to Edit->enhancement operations->show implicit enhancement options



4. You will see certain yellow color lines like this in the code

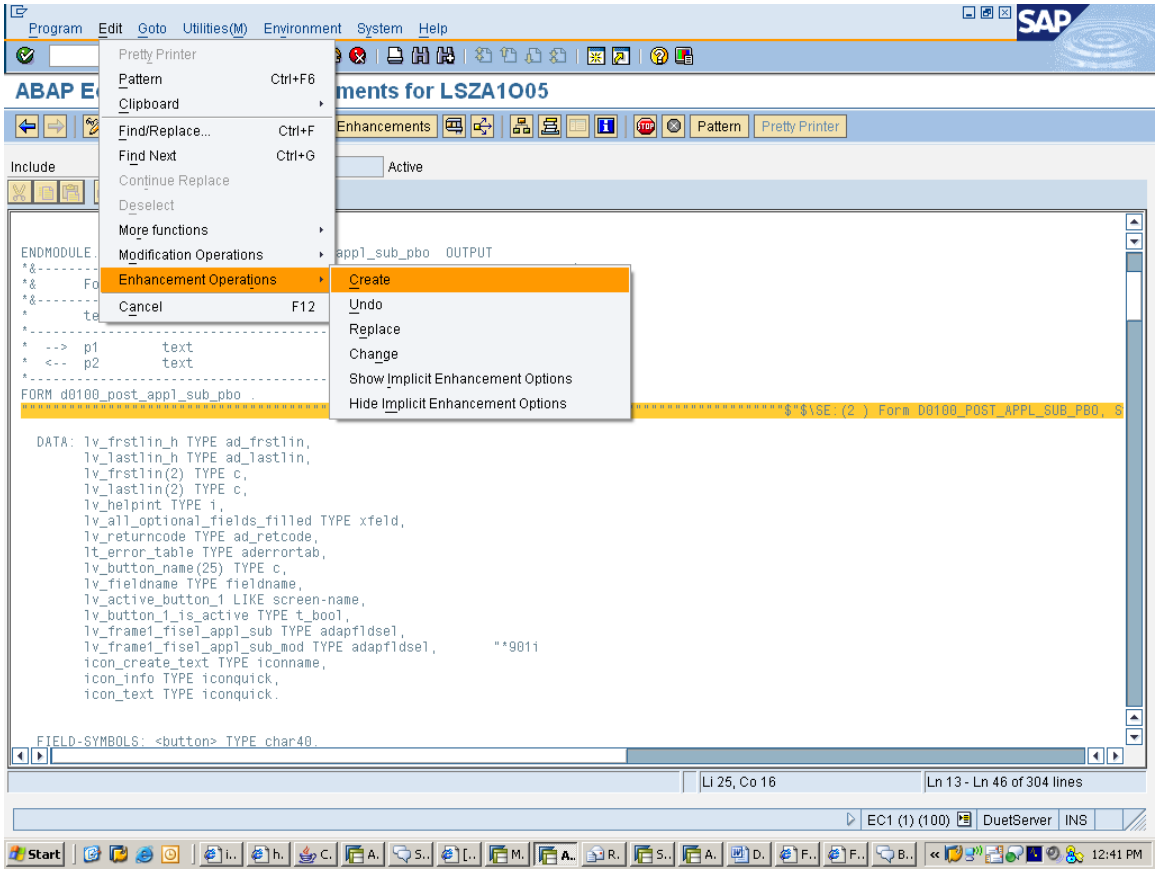
```
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
""""""""""""""""""$"$\SE:(2 ) Form D0100_POST_APPL_SUB_PBO, Start   A
```



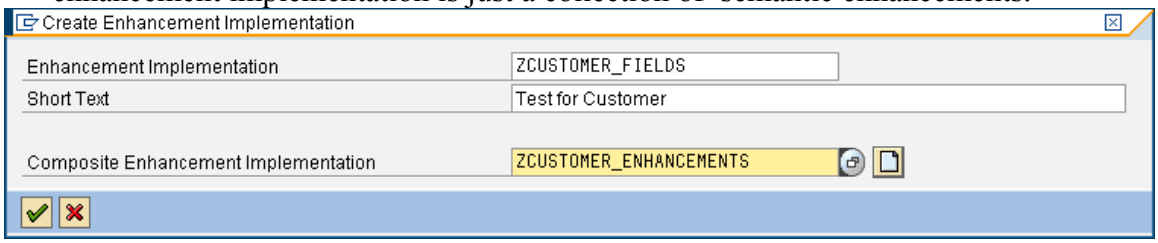
If you look at the end of the enhancement line ("""""""""""""""""""") you will see one letter.

- 'S' means that you can only add static information (data definitions).
- 'D' is dynamic; you can only add commands like MOVE.
- And 'A' stands for ALL that means you can add definitions and commands.

5. Click on the line where you want to enhance the code so that the cursor stands there. Then go to Edit->enhancement operations->create.



6. It will ask for implementation and composite implementation name. Give a Z implementation name and a short text. If you want to group similar enhancements then give a name of Composite enhancement implementation. Composite enhancement implementation is just a collection of semantic enhancements.



7. This will create an implementation which will include your code.
8. Save it and activate it. Now it will behave as if it's a part of the standard SAP code.

2. Explicit enhancement options

Explicit enhancement options are generally defined in a central initial system. Enhancements are made in follow-on systems. They are managed by enhancement spots and enhanced by enhancement implementations.

Explicit enhancement options are created in ABAP programs by the following statements:

- **Enhancement Point**
  Defines a position in an ABAP program as an enhancement option, at which one or more source code plug-ins can be inserted.

- **Enhancement-Section**
  Defines a section of an ABAP program as an enhancement option, which can be replaced by one or more source code plug-ins.

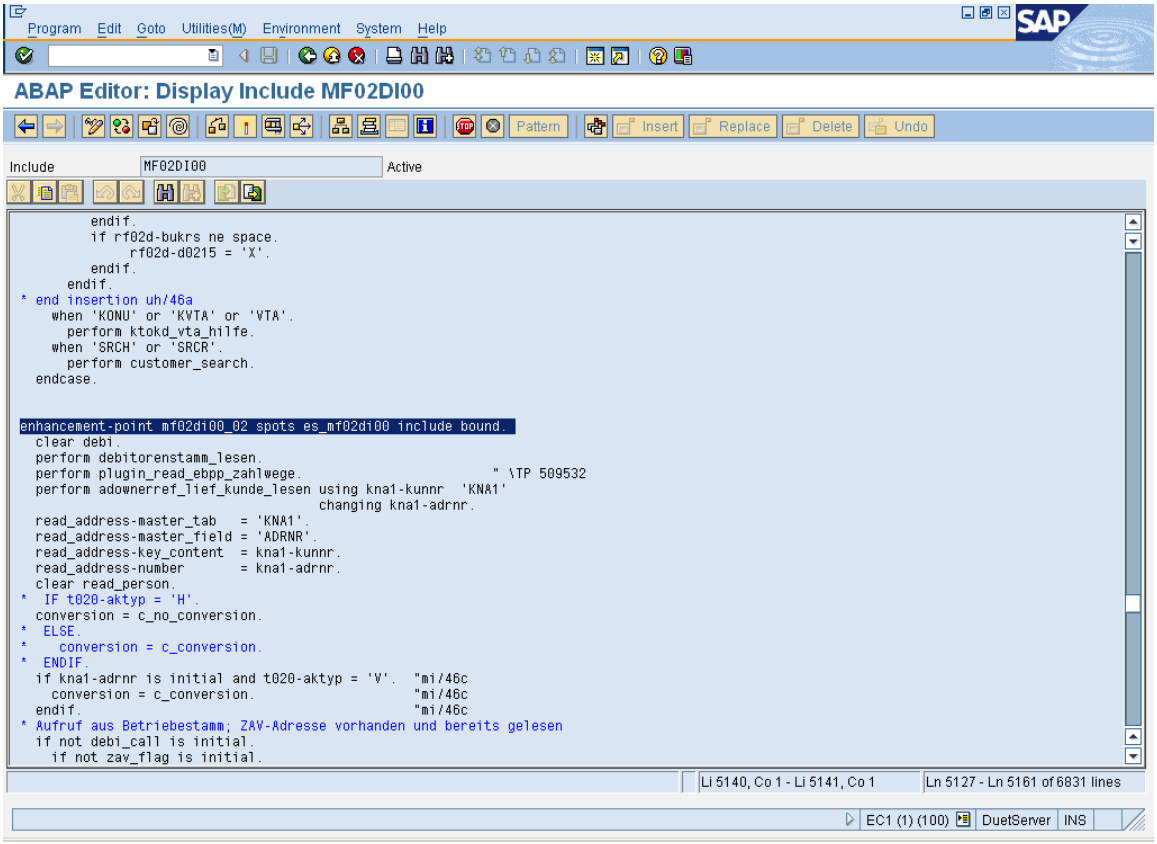
The difference between **enhancement point and enhancement section** is that we can have multiple implementation at an enhancement point, each of which will be called one by one. Enhancement section too allows for multiple enhancement implementations but only one of the active implementation will be called at run time (depending on switch status or using conflict resolution).

Like implicit enhance options, there are some restrictions on putting the type of the code in explicit enhancements. In case of explicit enhancement STATIC addition is intended for the enhancement of data declarations. If STATIC addition is not present than the enhancement option can be used for executable coding.

Enhancement Point

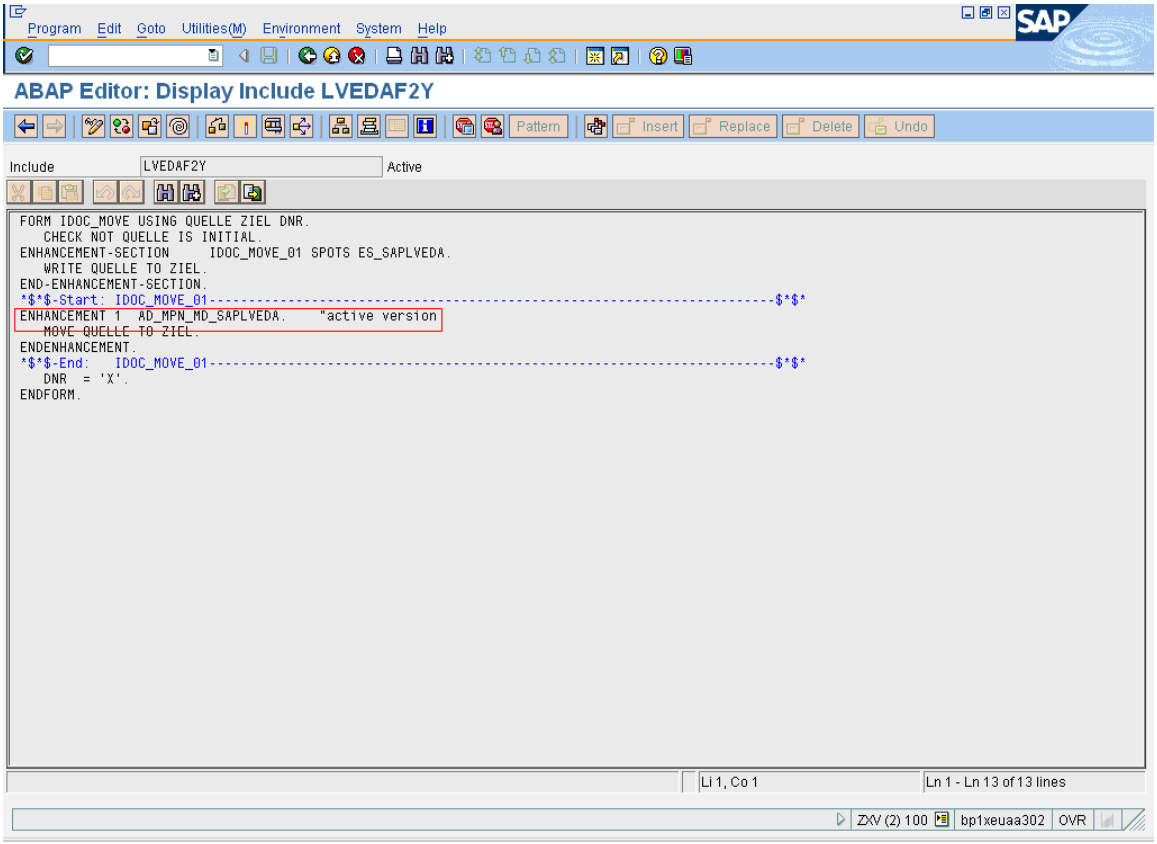
If you see some of the standards programs you can see something like this:

```
enhancement-point mf02di00_02 spots es_mf02di00 include bound.
```



This means we can plug in our code here with any number of enhancements implementation each of which will be called at run time if they are active and its switch is on(if no switch is assigned , the default status is on).

Enhancement Section



If you see the above screen, it shows one enhancement option whose implementation AD_MPN_MD_SAPLVEDA is active. So at runtime this active implementation will ve called

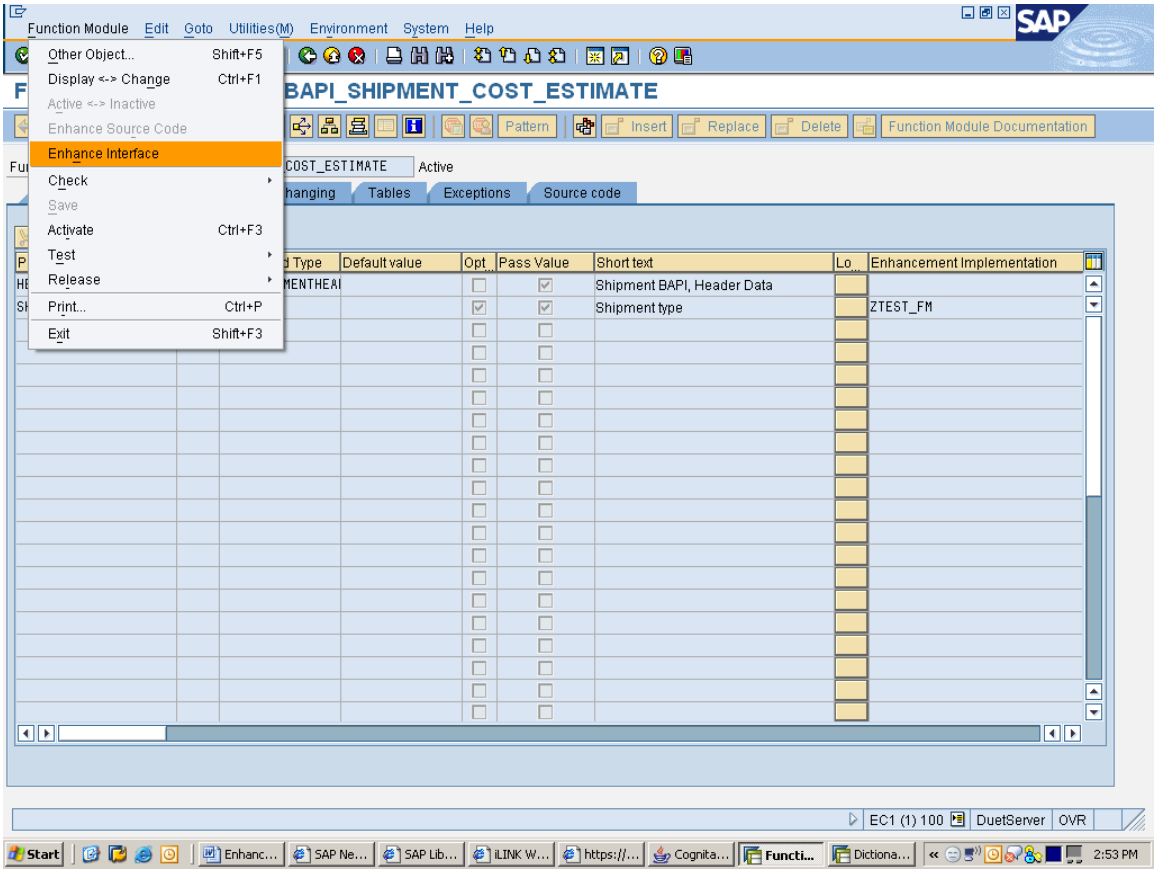
Implementation of Explicit Enhancement Options:

For implementing this follow the same steps as for implicit enhancements but instead put curser on enhance-point or enhancement section line and then go to Edit->enhancement operations->create.

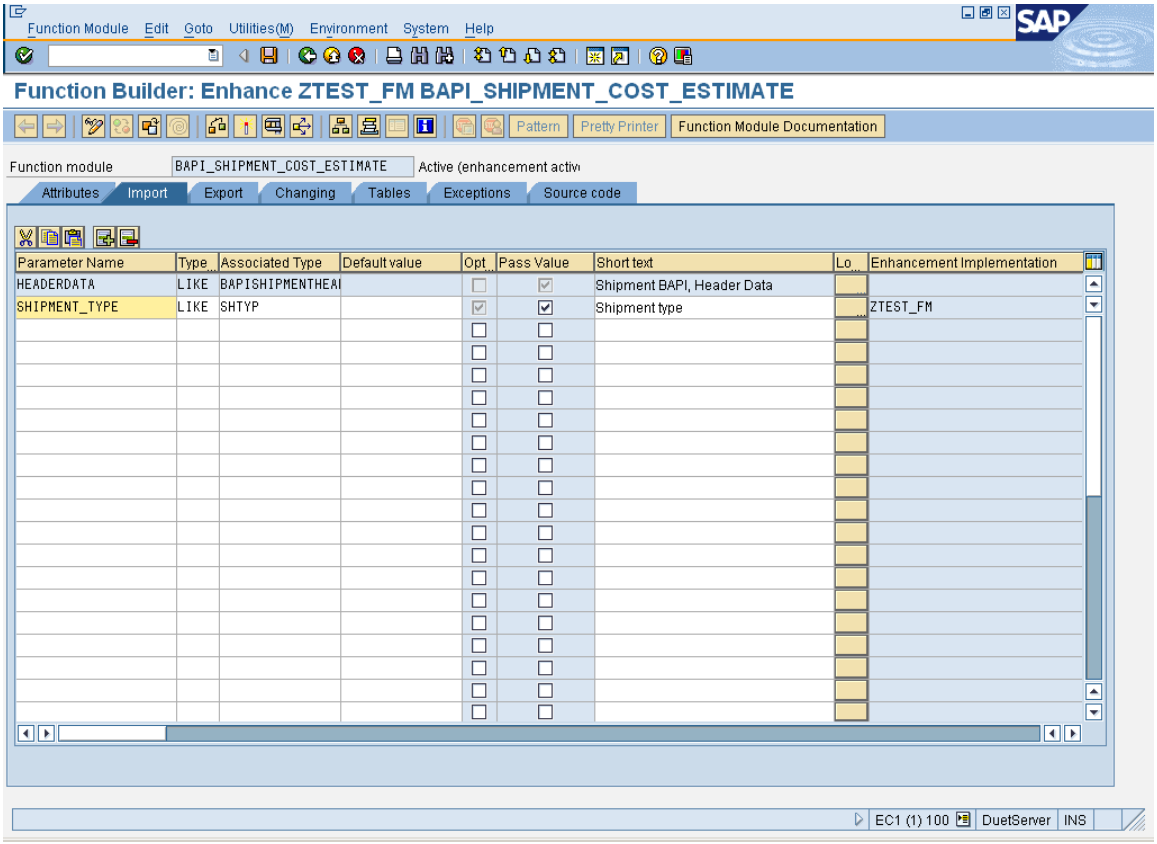
Enhancing FM interface

Optional parameters can be added to the standard FM interface.

1. Open the FM . Go to Menu path Functional Module->Enhance Interface. It will ask for implementation name and composite implementation name.



2. After giving implementation name. Add the parameter you want to add in the Form interface. Then save and activate.



Similarly we can enhance class interface and methods.

Enhancement using BADIs

Although I am not going to dive into this topic but I would like to tell something about this.

The source code enhancements are uncontrolled enhancements. i.e. no one has control on what you write inside the enhancement option. Inside enhancement option you have access to standard sap variables. You can change them unknowingly that it may impact the whole standard SAP functionality. To address this issue a controlled enhancement is there known as Enhancement using BADIs. It basically gives an interface with the parameters to be changed and parameter only to be read etc. so that the variables which should not be touched will be protected.

Enhancement Framework and Switch Framework

Switch framework is a huge topic in its own.

I just want to mention, you can put all your enhancement objects inside one package and activate or deactive that package using switch framework.

It's like this you need the enhancement functionality in one system but does not require it in others system. For example one refinery implementation doesn't require the enhance functionality while others want it, then just deactive the package containing the enhancements in that refinery implementation.

References:

1. http://help.sap.com/saphelp_nw2004s/helpdata/en/94/9cdc40132a8531e10000000a1550b0/frameset.htm
2. https://www.sdn.sap.com/irj/sdn/weblogs?blog=/pub/wlg/2687
3. https://www.sdn.sap.com/irj/sdn/weblogs?blog=/pub/wlg/3336
4. https://weblogs.sdn.sap.com/pub/wlg/3056