BADI—Business Add Ins

When enhancements are done using object-oriented concepts without modifying the original code, it is BADI.

BADIs are technically just an interface as in OOPs concept.

What is SSCR in slide 5?

BADI using repository and SQL trace?

In BADI, all the enhancement components are grouped together.

* Program Enhancements (interface methods)
* Menu Enhancements (function codes in interface definition)
* Screen Enhancements

A BADI can be single, multiple, filter, outside or within SAP.

A Badi interface can have several interface methods.

If the business add-in is filter-dependent, you must define an import parameter flt\_val for each method. Otherwise, you define the interface.

Each BADI consists of the method without implementation called as BADI definition.

We need to create classes to write the abap code by implementing the methods called as BADI implementation.

For e.g. in ZAT\_OOAP\_DEM0011A

INTERFACE lif\_employee.

METHODS: set\_attributes.

ENDINTERFACE. 🡺 no implementation so BADI definition

CLASS employee DEFINITION.

PUBLIC SECTION.

INTERFACES lif\_employee.

METHODS: display\_attributes.

ENDCLASS. 🡺 implementation is done so this is similar to BADI implementation.

SE18 is the T-code for BADI definition, SE19 is the T-code for BADI implementation

Advantages of BADI

* The main advantage of using BADI's is, we can create multiple implementations for a single BADI definition.

Whereas with the exits, we can create a single implementation. i.e. a single project for an enhancement.

We cannot create another project (implementation) for enhancement which is already used. That is why we go for BADI's.

For instance, the existence of the class named CL\_EXITHANDLER tells it is BADI.

If the Exit consists of the term BADI – it shows it has BADI

2 types:

* Standard BADI
* User defined BADI

BADI will have:

BADI Definition

BADI interface

BADI Implementation or in other words a class that implements the interface.