

ABAP Workbench 4.5



Karl Kessler
SAP AG



Application Engineering Tools

- **Development tools**
ABAP Workbench, ABAP Objects
- **Personalization**
Session Manager, Transaction Variants
- **Simplification**
SAPscript Form Painter and Editor
- **Software quality**
Test Workbench and CATT
- **Online documentation tools**
SAP Knowledge Engineer
- **Customizing tools**
View maintenance

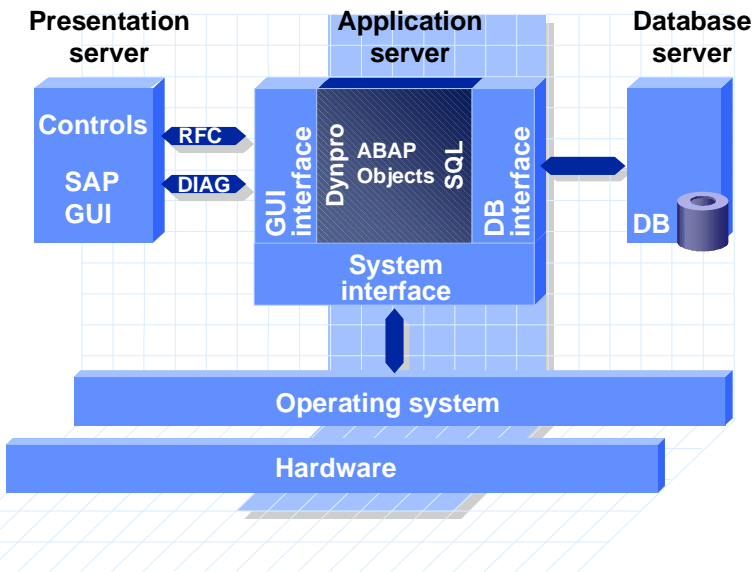


Application Engineering Tools

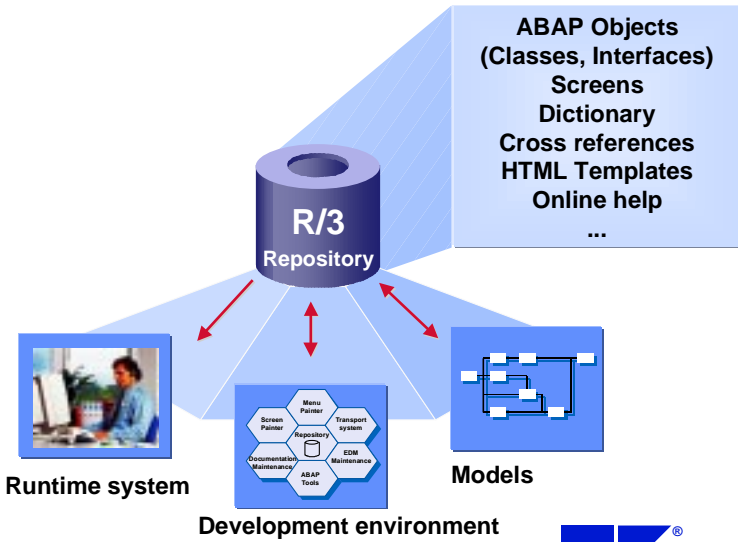
- **Development tools**
ABAP Workbench, ABAP Objects
- **Personalization**
Session Manager, Transaction Variants
- **Simplification**
SAPscript Form Painter and Editor
- **Software quality**
Test Workbench and CATT
- **Online documentation tools**
SAP Knowledge Engineer
- **Customizing tools**
View maintenance



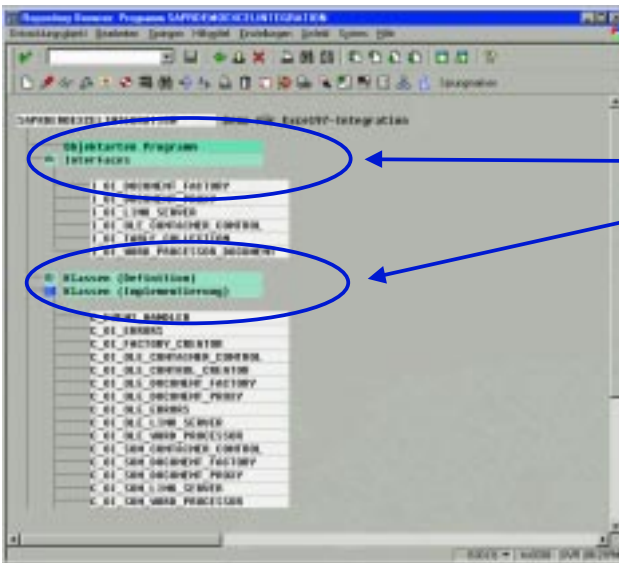
Runtime Architecture for ABAP Objects



R/3 Repository



Repository Browser

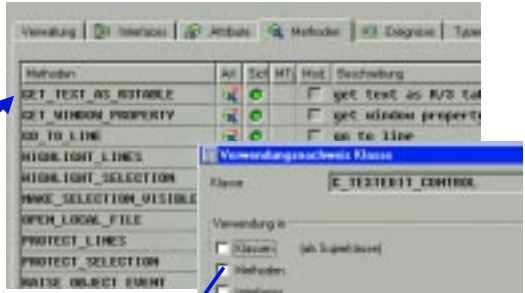


- Where used lists
- Interfaces
- Classes
- Generic access



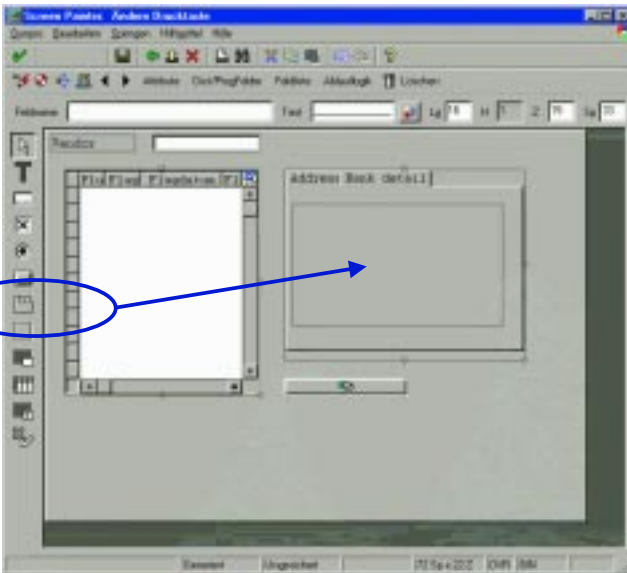
Navigation

= relink control to other container as same id
perform relink_control.
WHEN 'SAPR',
= retrieve table from control
call method editor->get_text_as_rtable
importing table = mtable.



Programm	Funktions/Kurzbeschreibung
<input type="checkbox"/> SAPTEXTEDIT_BCH0_1	
<input checked="" type="checkbox"/> SAPTEXTEDIT_BCH0_2	167 call method editor->get_text_as_rtable importing table = mtable.
<input type="checkbox"/> SAPTEXTEDIT_BCH0_3	79 call method editor->get_text_as_rtable importing table = mtable exception: others = 1.
<input type="checkbox"/> SAPTEXTEDIT_TEST1F01	Include HTEXTEDIT_TEST1F01
<input type="checkbox"/> MTEXT167	Bemprogramm zum binden des Controls an verschie
<input type="checkbox"/> ZHTEXTEDIT_BCH0_1	Bemprogramm zum binden des Controls an verschie
<input type="checkbox"/> ZHTEXTEDIT_BCH0_2	Bemprogramm zum binden des Controls an verschie
<input type="checkbox"/> ZHTEXTEDIT_TEST1F01	Include HTEXTEDIT_TEST1F01

Screen Painter



- Tab Strip
- Table Control
- Icons
- Buttons
- Boxes
- Flow control



News 4.0/4.5/Enjoy

- Object-oriented extensions to ABAP language
- Class Builder
- Control-enabling technology (ActiveX integration)
- Workbench Manager (Enjoy initiative)
- Modification assistant



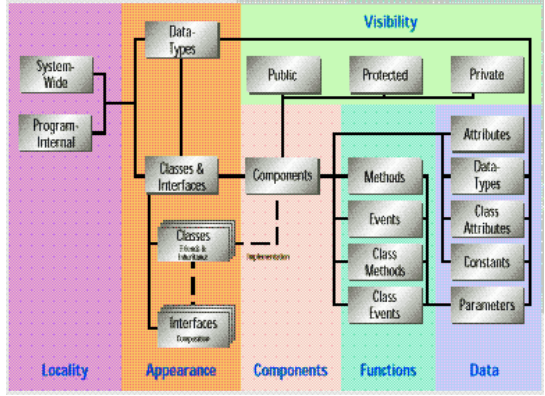
Object Orientation: Design Goals

- Natural extension of conventional ABAP
- Compatibility with earlier releases
- Integration into 3-tier architecture
- Efficiency through kernel implementation
- Features based on: C++, Java
- Encapsulation of remote objects (DCOM, ActiveX)
- Support for GUI objects
- Support for business objects and workflow



ABAP Objects

- **ABAP Objects is available and in use**
Office Integration, Class Builder, ActiveX Integration
- **Local and global classes in R/3 repository**
- **ActiveX Controls as ABAP classes**
- **Java Beans as ABAP classes**
- **Control framework independence**
- **Basis for remote object systems**



© SAP AG 1998

J02 SAPTechEd '98, Karlsruhe (K. Kessler) / 11

Example

```

CLASS Ctruck DEFINITION.
  PUBLIC SECTION.
    DATA: VehicleId TYPE I.
    METHODS: LoadParcel IMPORTING parcel TYPE REF TO CParcel,
             UnloadParcel ...
  PRIVATE SECTION.
    DATA: ParcelTab TYPE REF TO CParcel OCCURS 0.
ENDCLASS.

```

```

CLASS Ctruck IMPLEMENTATION.
  METHOD LoadParcel.
    APPEND parcel TO ParcelTab.
    "-- additional code ...
  ENDMETHOD.
ENDCLASS.

```

```

PROGRAM xy.
  DATA: truck TYPE REF TO Ctruck.
  DATA: parcel TYPE REF TO Cparcel.

  ...    "-- get input data for parcel from somewhere ...
  CREATE OBJECT truck.
  Truck->vehicleid = 123.
  CREATE OBJECT parcel.
  CALL METHOD truck->LoadParcel importing parcel = parcel.

```

© SAP AG 1998

J02 SAPTechEd '98, Karlsruhe (K. Kessler) / 12

ActiveX Example: Web Browser Control



PBO.

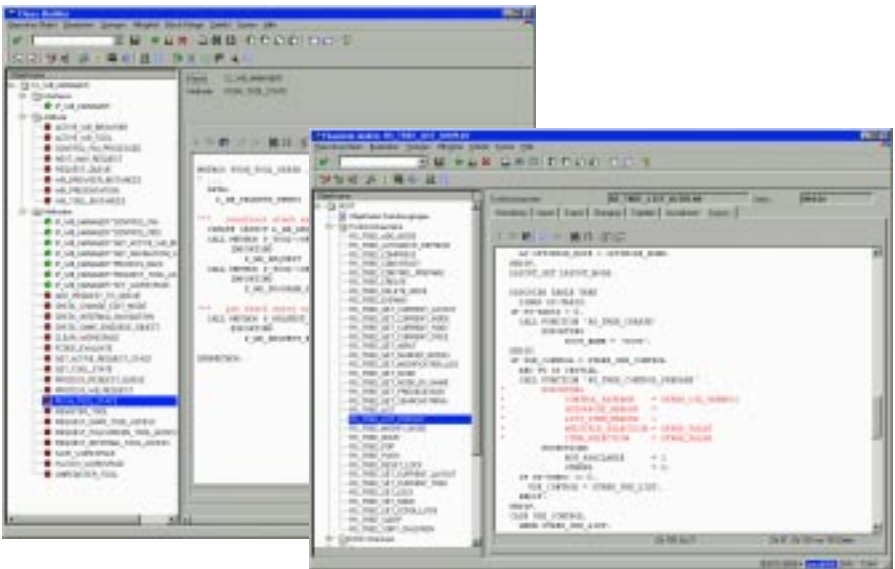
- ...Create instance (constructor)
- ...Link to dynpro (constructor)
- ...Call methods, Set properties (attributes, methods)
- ...Register for events (register methods)

PAI.

- ...Receive special ok-code.
- ...Dispatch control event (dispatch methods)
- ...Handle control event (handler classes)



Workbench Manager (Enjoy Release)

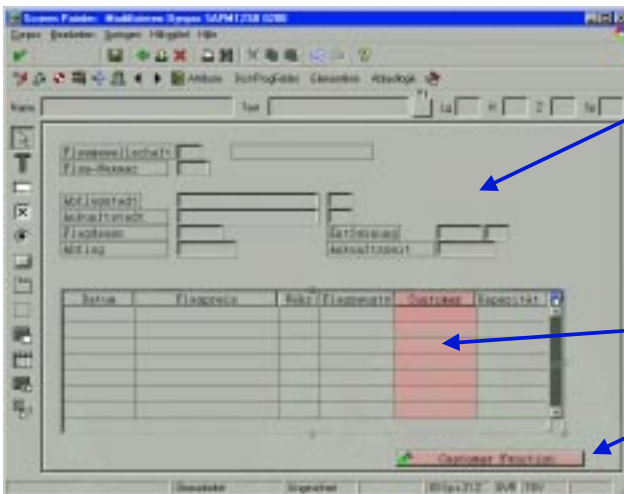


Modification Assistant

- Techniques to adapt the R/3 applications without modification
 - Customizing
 - Report variants, transaction variants
 - Personalization
 - Customer exits
 - ◆ menu, dynpro, field, function, keywords
- Do not allow for *arbitrary* modifications
- Modification assistant
 - Built-in support for *controlled* modifications
 - Upgrade support



Example: extending the user interface



Modifications in ABAP Editor

```
FORM user_command.
  CASE sy-ucomm.
    WHEN fun1.
```

```
*{# SDK0005378 05/25/1997 Replacement
*   IF condition.
*   IF condition OR mycondition.
*}#
```



```
    perform handle_fun1.
  ELSE.
    ...
  ENDIF.
WHEN fun2.
```

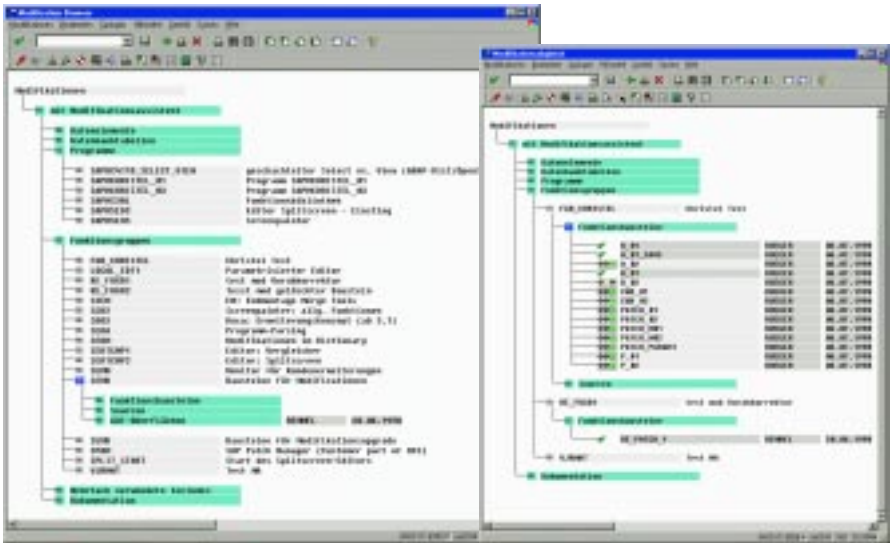
```
*{# SDK0005466 06/19/1997 Insertion
  WHEN myfun.
    perform handle_myfun.
*}#
```



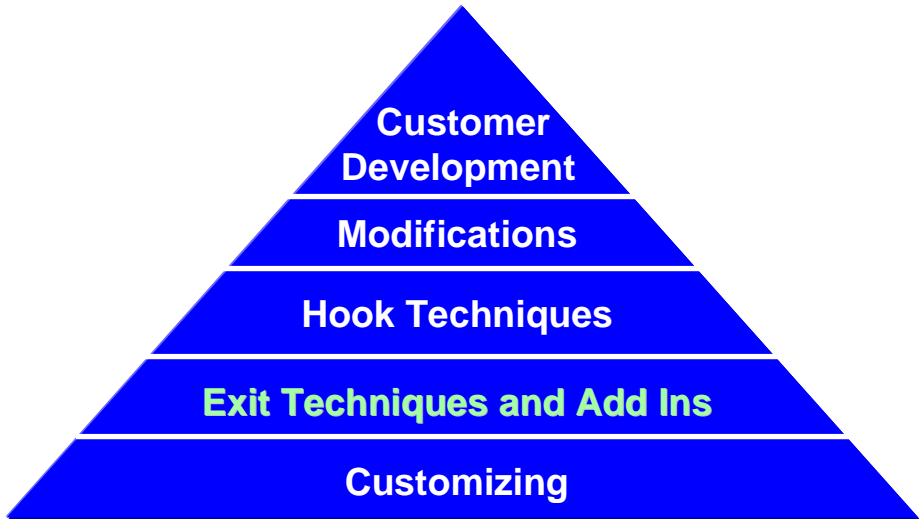
```
  ENDCASE.
ENDFORM.
```



Modification Browser (SE95) and Upgrade Support (SPAU)



Infrastructure for Industry Solutions



© SAP AG 1998

J02 SAPTechEd '98, Karlsruhe (K. Kessler) / 21

Add Ins: Definition

Add ins are points in an object's source code where additional functions or statements can be inserted. Add ins are defined by the object's initial developer and allow other developers to easily include additional code in a subsequent phase of development without having to 'modify' the original. Additional advantages:

- No change licences necessary
- Interfaces are upwardly compatible
- Less work at upgrade



© SAP AG 1998

J02 SAPTechEd '98, Karlsruhe (K. Kessler) / 22

Requirements

- Delivery of implemented add ins (country-specific versions, IBU solutions, partner software, ...)
- Technique can be used by other software vendors
- Delivery and correction of default solutions
- Filter-dependent implementation possible
- 'Event-like' add ins (Publish & Subscribe)
- Integrated administration and documentation
- Trace possibilities



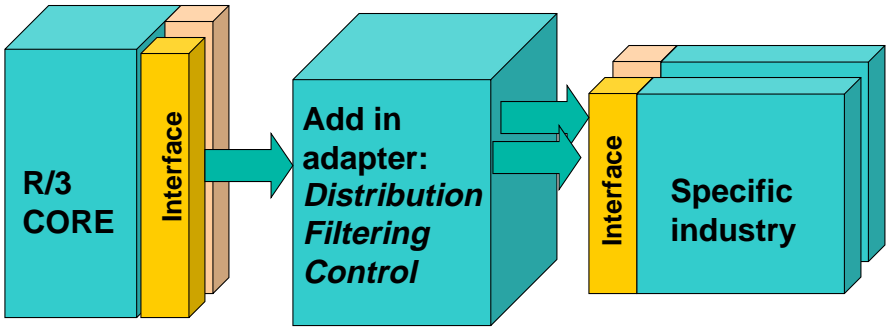
Why Business Add Ins?

- Perform calls in programs / using tables
 - Flexible <-> Fuzzy interface (global data,...)
- Customer exits (SMOD/CMOD)
 - System infrastructure: SAP - customer
 - Naming convention not compatible with namespace extension
- Business Transaction Events (Open FI)
 - No bundle of objects that belong together
 - No interface elements

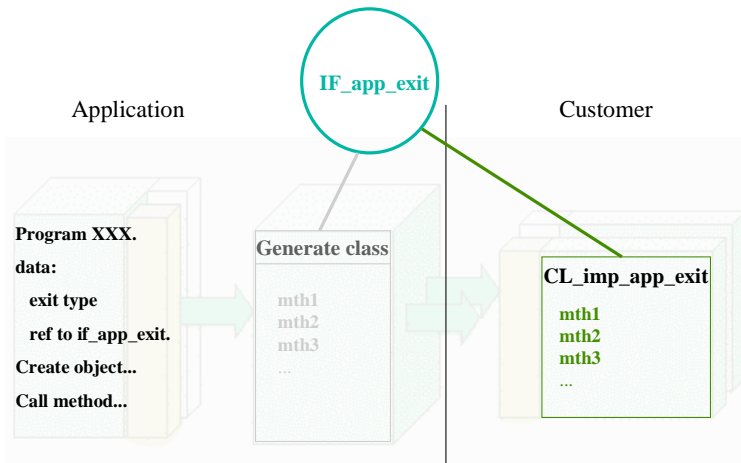
Business add ins should be viewed as expanded and enhanced business transaction events



Business Add Ins: Architecture



Architecture



Example Program

```
REPORT BADI .
CLASS CL_EX_BADI DEFINITION LOAD.
DATA EXIT TYPE REF TO IF_EX_BADI.
DATA WORD(15) TYPE C.

CREATE OBJECT EXIT TYPE CL_EX_BADI.

START-OF-SELECTION.
  WRITE:/ 'Please click here'.

AT LINE-SELECTION.
  NEW-PAGE.
  WORD = 'Business add in'.
  WRITE:/ 'Original word:  ', WORD.

  CALL METHOD EXIT->METHODE
    CHANGING
      PARAMETER = WORT.

  WRITE:/ 'Changed word: ', WORD.
```



Filter Dependence

Sometimes it is important for partners to be able to implement and deliver a single add in with different filter values (for different countries, for example). For this reason, add ins can be defined for specific filter values.

- Type of filter: data element with search help
- Parameter *flt_val* must be suitably filled when the add in is called.
- Add in must be implemented for a specific filter value



Filter-Dependent Add Ins

Add in definition

```
IF_app_exit
Type of filter dependency:
country-specific
```

Exit call

... Exit type ref to IF_app_exit.

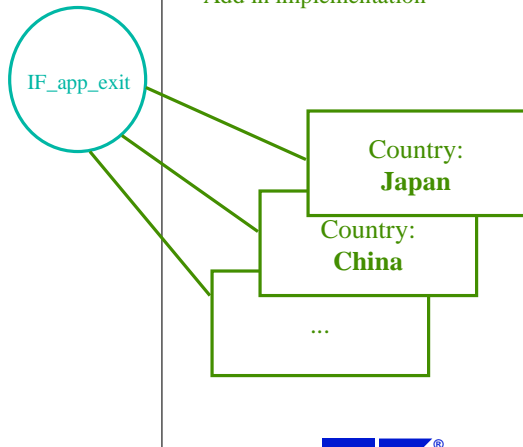
Country = ,J'.

Call method exit->mtl

exporting

flt_val = country.

Add in implementation



Add In Manager

- Definition of related objects
 - Interfaces
 - User interface functions
 - Documentation
- Test implementation
- Assignment to Implementation Guide (IMG)
- Implementation
- Implementation calls from IMG

