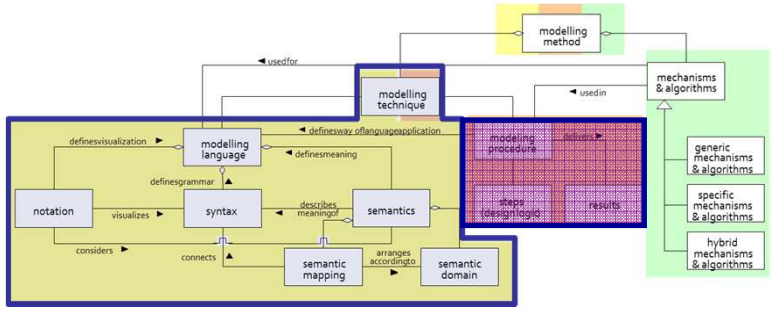


INTERREF Editor – Tree List

SCENARIO:

Building Tree List with Appropriate Objects to Add Interref

Scenario Description

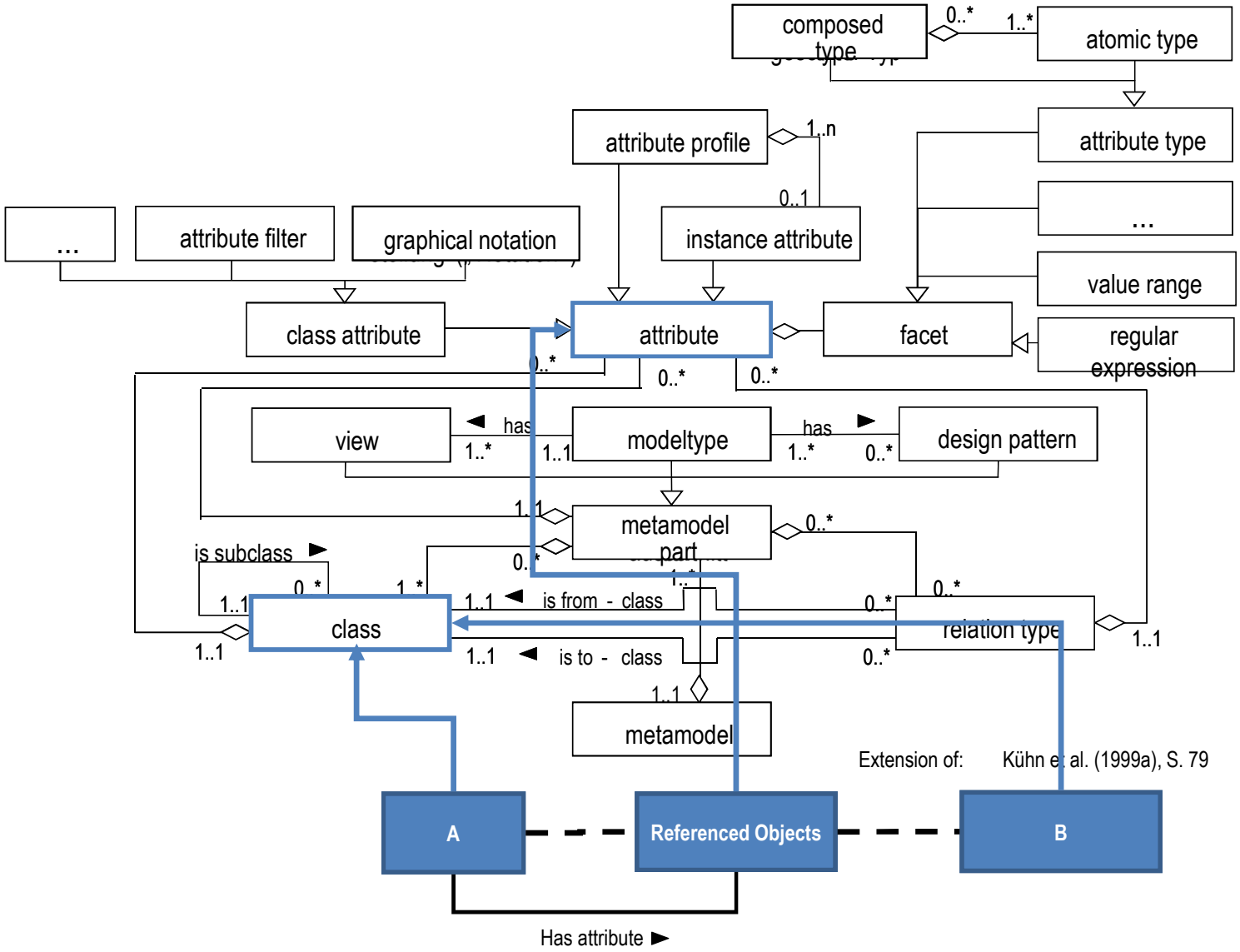
Case:

Realization of tree-list with objects appropriate to set interref from objects of class A.

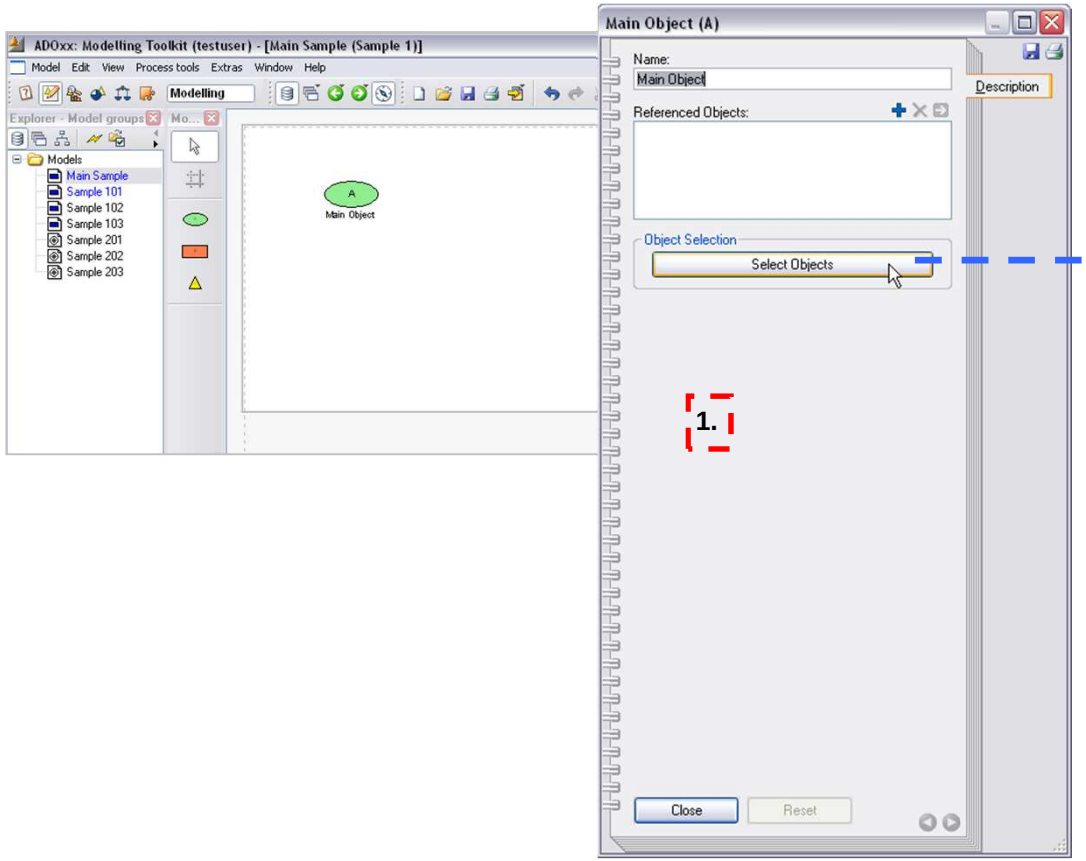
GOAL:

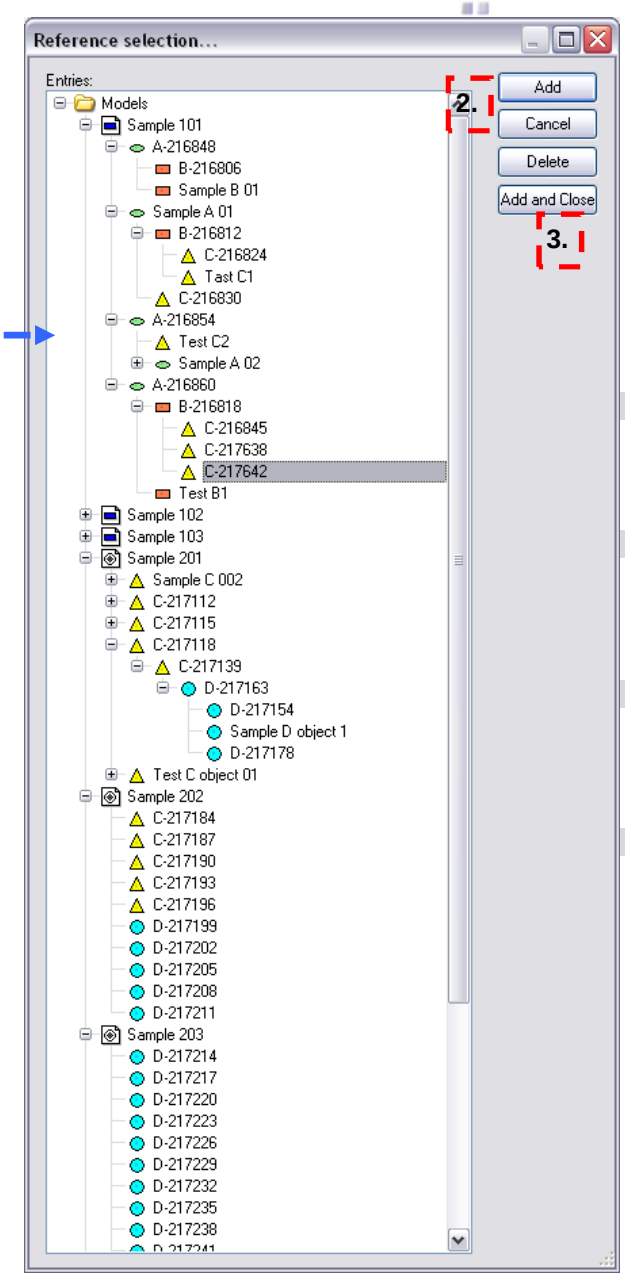
Demonstrate how to build a tree-list with pre-filtered objects.

Meta Model of Meta Modelling Language



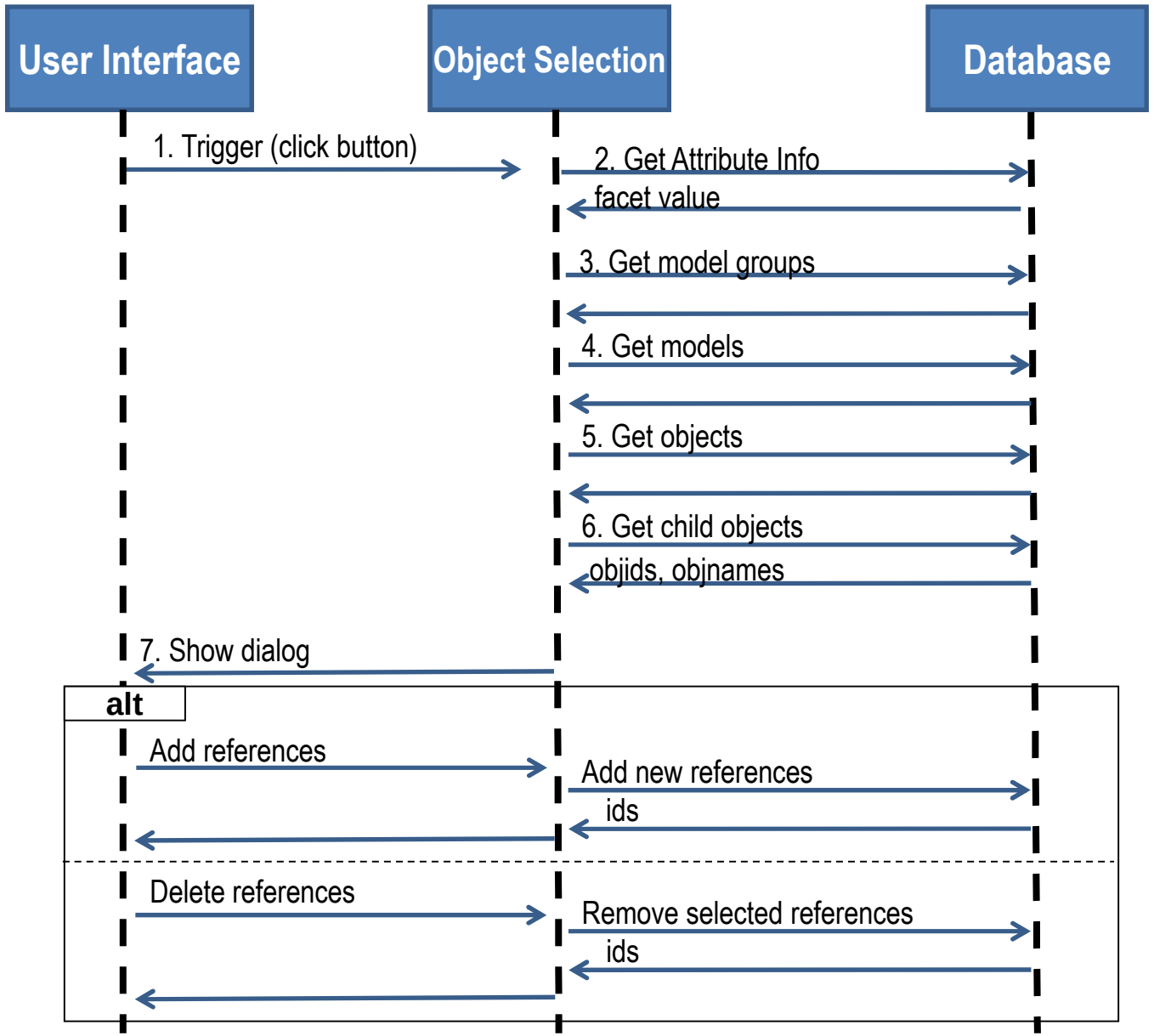
Using the Interref Editor



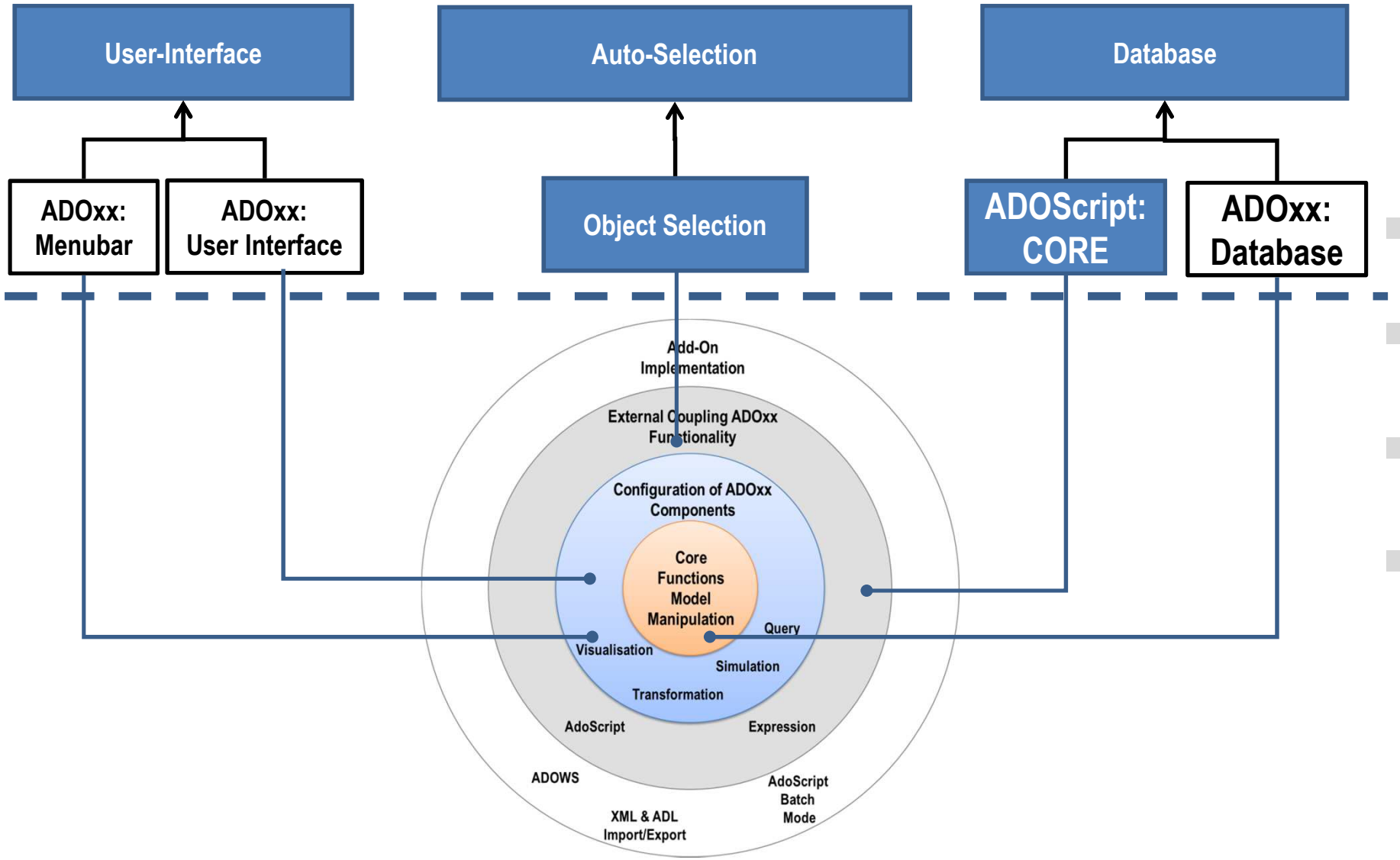


After the Interref Editor is opened, all existing references are automatically selected.
The button „Add“ adds references targeting the selected objects.
If a model or a model group is selected, no relation is added targeting it.
The button „Delete“ removes any existing reference targeting the selected object(s).
The button „Cancel“ closes the dialog without adding the last selection.
The button „Add and Close“ adds references for the selected objects and closes the dialog.

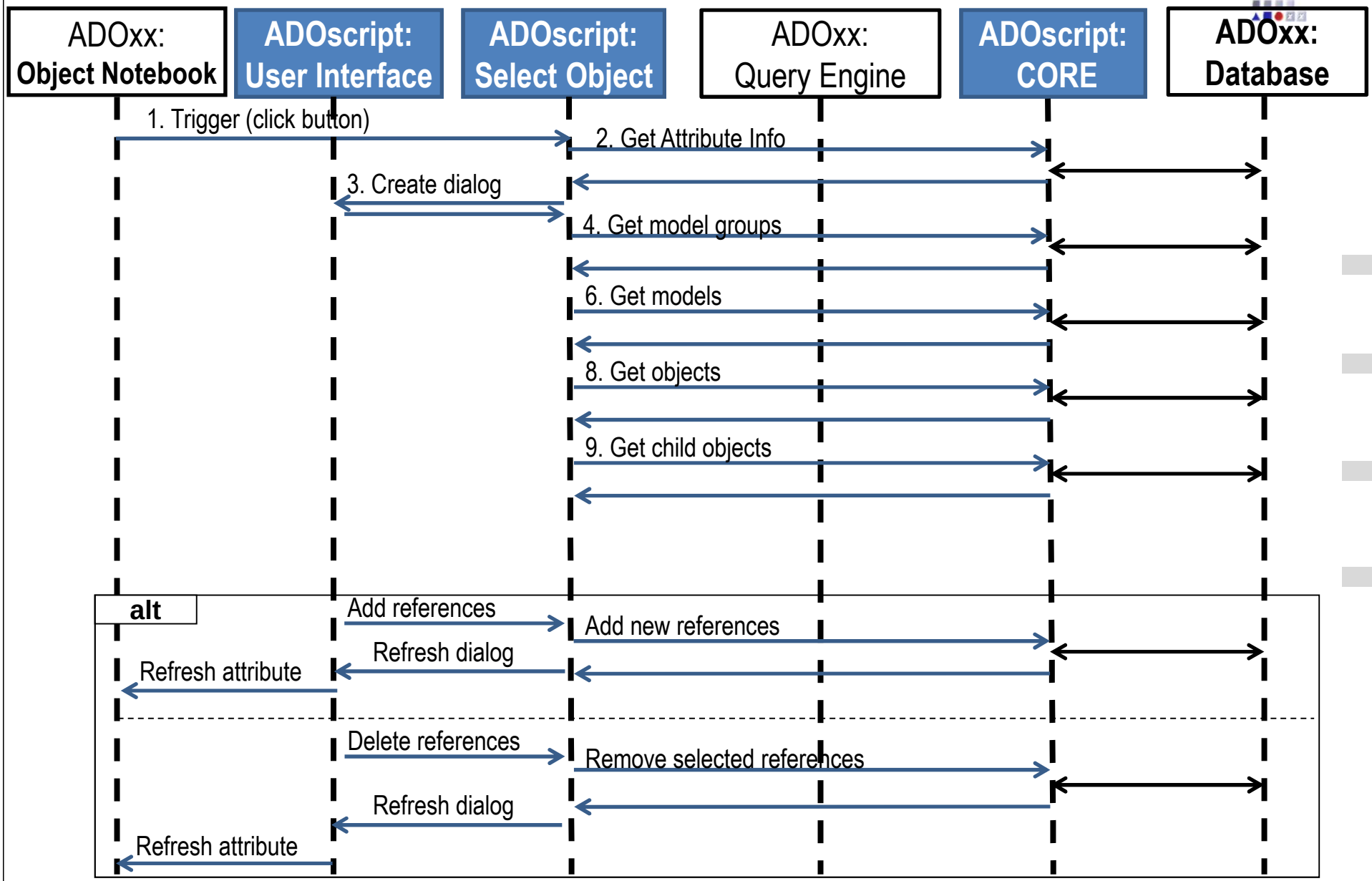
Description of Algorithm



Mapping ADOxx Functionality



ADOxx Realisation Approach



Added Value of Metamodelling Platform

Used meta-modelling functionality for realisation of the scenario:

- **AttrRep (NOTEBOOK):**
- **Attribute Type: PROGRAMCALL**
- **Attribute Type: INTERREF**
- **AdoScript:**

ADOxx Realisation Hands-On

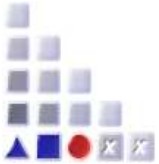
1. Realisation of Modelling Language
    1. Define Model Types “Sample 1”, “Sample 2”
    2. New class “A”, “B”, “C”, “D” and “Aggregation”
    3. Add Attributes
2. Implement Algorithm with ADOscript
    1. Object Selection

Used ADOxx Functionality: Implementing an Algorithm

- Introduction
- Setup of Implementation Environment

- Modelling Language Implementation
  - **Classes**
  - Relations
  - **Class Attributes and Attributes**
    - **GRAPHREP**
    - **ATTRREP**
    - CLASS Cardinality
    - CONVERSION
    - Model Pointer
  - **Attribute Facets**
  - **Model Types**

- Mechanisms & Algorithms Implementation
  - Core Functions for Model Manipulation
    - **Database**
    - Visualisation
    - Query
    - Transformation
  - Configuration of ADOxx Components
    - Visualisation
    - Query
  - **External Coupling ADOxx Functionality**
    - **ADOscript Triggers**
    - ADOscript Language Constructs
    - **Visualisation ADOscript**
    - Visualisation Expression
    - **Query ADOscript**
    - Transformation ADOscript
  - ADD-ON Implementation
    - ADOxx Web-Service
    - XML / ADL Import – Export
    - ADOscriptBatch Mode



www.adoxx.org

HANDS-ON

INTERREF EDITOR MECHANISM

SCENARIO:
IMPLEMENTING A MECHANISM

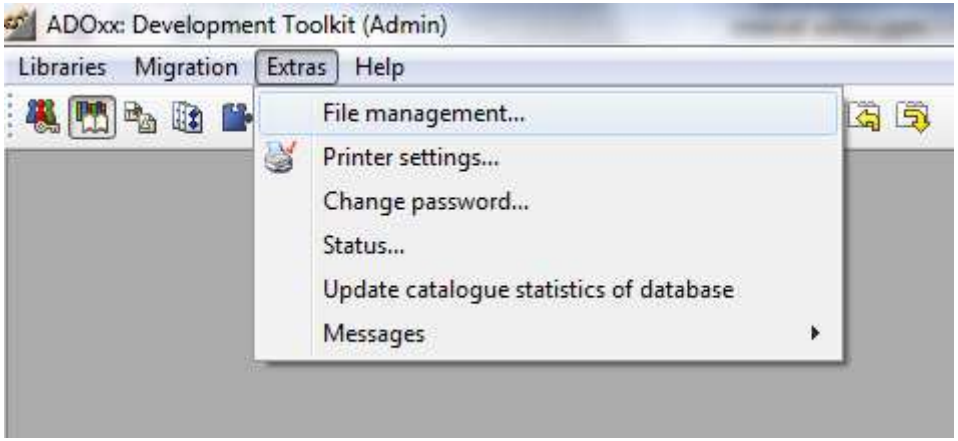
www.adoxx.org

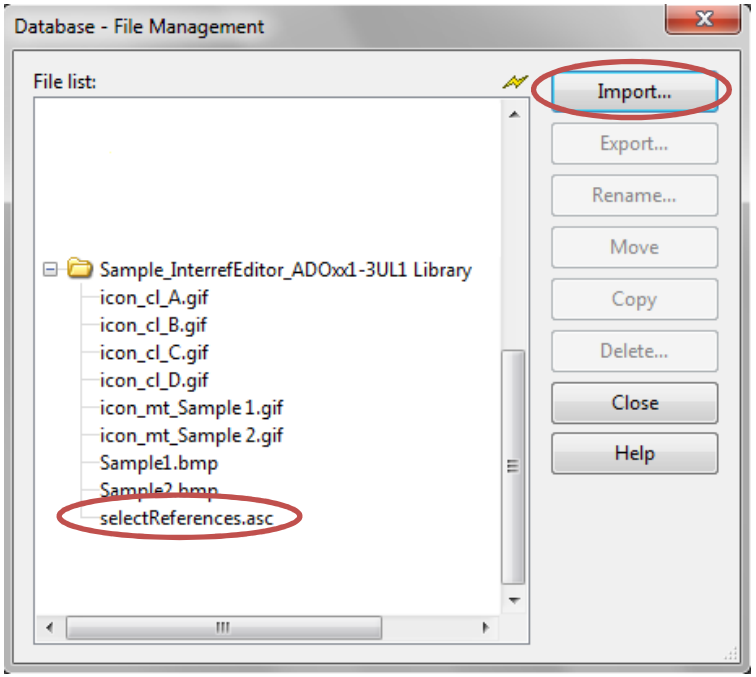
HANDS-ON

1. PREPARATION

Import `Sample_InterrefEditor.abl` library or integrate the *Interref Editor* in your own library.

1. Import AdoScript file

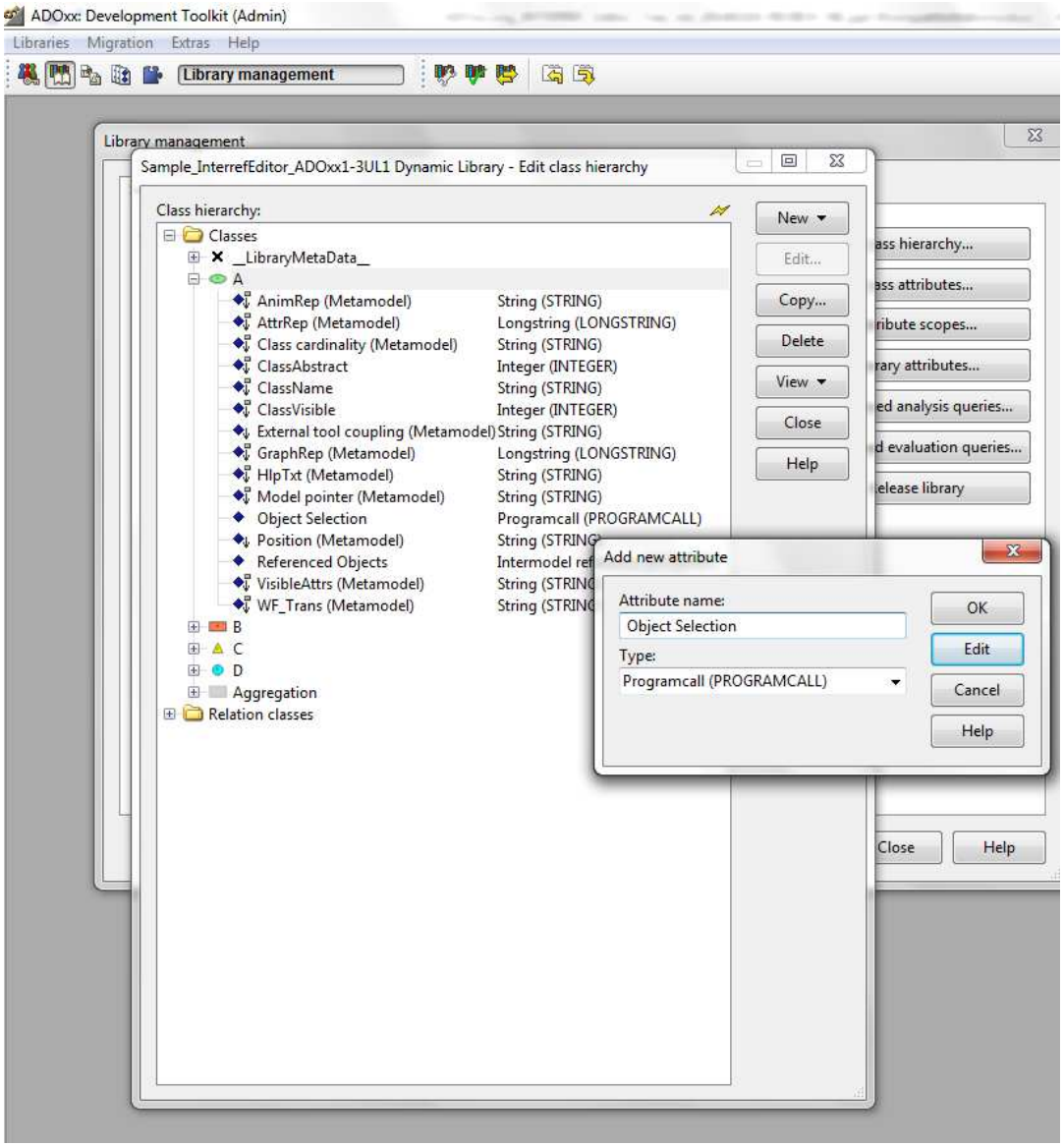




Import file to ADOxx database

- After starting ADOxx Development Toolkit, click *Extras* → *File management…*
- Select a Library, click Import…
- Select the file `selectReferences.asc` and click *Open*.

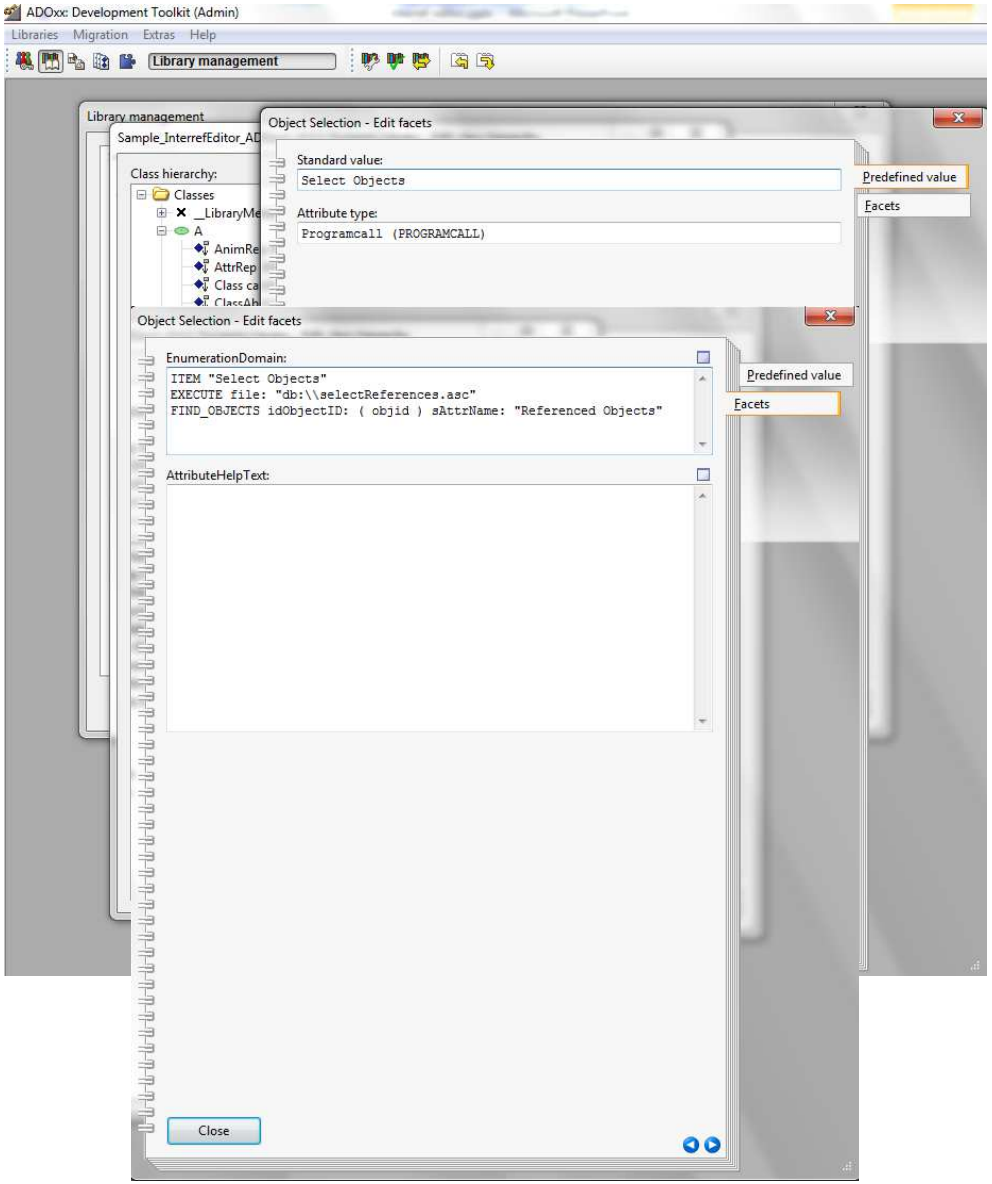
2. Creating *Programcall* attribute



Create Programcall attribute

- Go to your dynamic library
- Select a class which already has an *Interref* attribute.
- Click *New* → *New attribute…*
  - Name: *Object selection*
  - Type: *Programcall*

2. Creating *Programcall* attribute



Configure attribute

- Double-click *Object Selection* attribute
- Set *Standard value*:

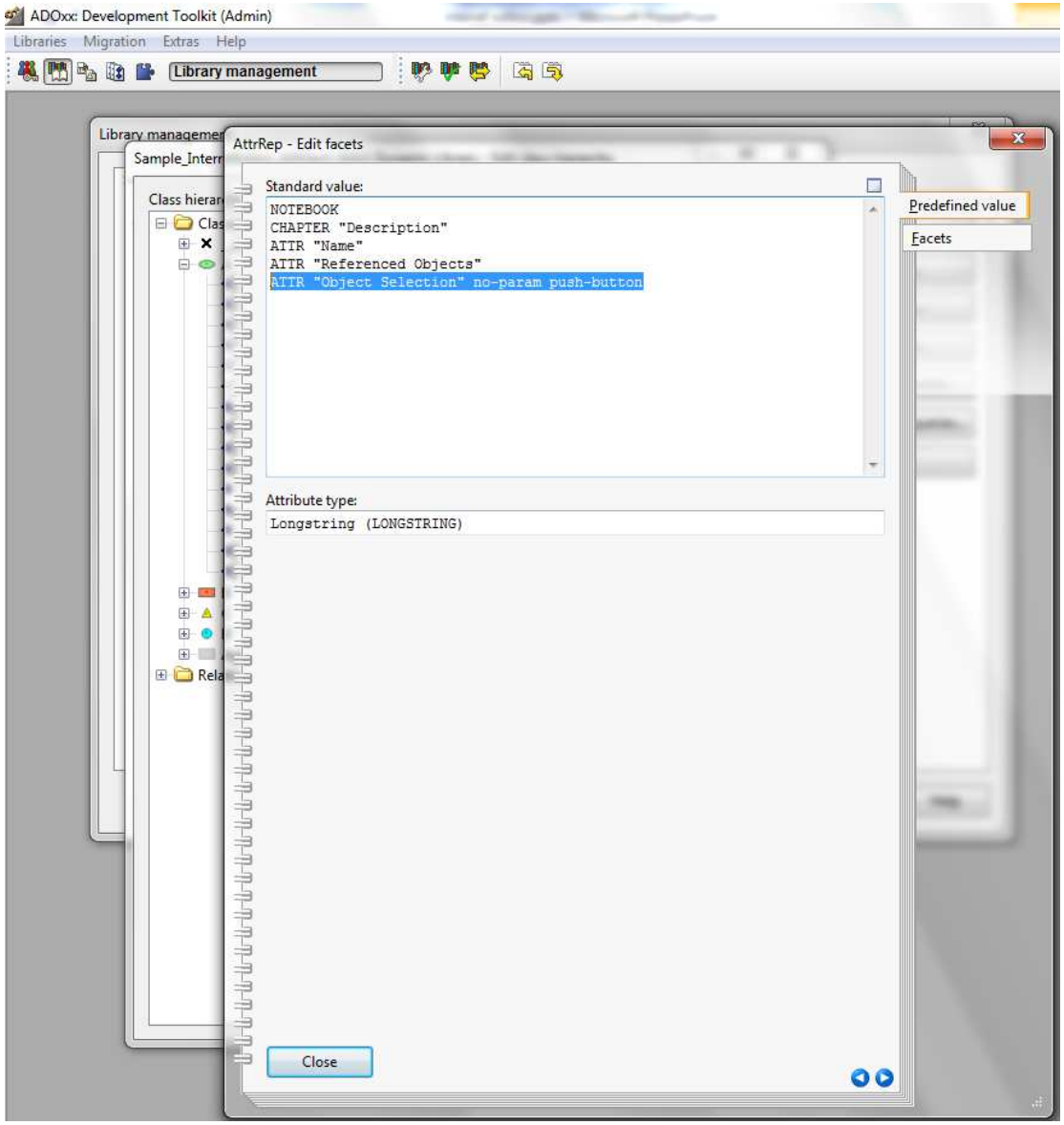
Select Objects

- Click *Facets* and set *EnumerationDomain*

```
ITEM "Select Objects"
EXECUTE file:
"db:\\selectReferences.asc"
FIND_OBJECTS idObjectID: ( objid )
sAttrName: "Referenced Objects"
```

- Replace *Referenced Objects* with the name of your *Interref* attribute.

2. Configuring *AttrRep* attribute



Configure attribute

- Double-click *AttrRep* attribute
- Add the line

ATTR "Object Selection" no-param push-button

www.adoxx.org

HANDS-ON

2. INTERREF EDITOR MECHANISM

AdoScript code in file selectReferences.asc

```
# Interref Editor
##################

# Usage:    - import this script into the application library where you want to use the Interref Editor
#       - define a new attribute of type PROGRAMCALL in the class where you want to insert de Interref Editor
#       - add a new ITEM to the PROGRAMCALL attribute:
#           ITEM <item_name>
#           EXECUTE file: "db:\\selectReferences.asc"
#           FIND_OBJECTS idObjectID: ( objid ) sAttrName: "Referenced Vehicle"
#       - set <item_name> as default value for the PROGRAMCALL attribute
#       - add the PROGRAMCALL attribute to the AttrRep class attribute of the class, with the keywords no-param and push-button:
#           ATTR <attr_name> no-param push-button
#       - in the AttrRep, add the modifier write-protected to the Interref attribute (optional)
#       - in the procedure ADD_OBJECT_TO_TLB, extend the object hierarchy if needed (optional)


PROCEDURE global FIND_OBJECTS idObjectID: integer sAttrName: string
{
    #set constants that may be configured...
    SET c_sDialogTitle: "Reference selection..."
    SET c_sDialogText: ("Select objects for " + sAttrName)
    SET c_sDeleteWarnMsg: "The selected references will be deleted.\nDo you want to continue?"
    SET c_sDeleteWarnTitle: "Deleting References..."
    SET c_sCancelMsg: "The selection process has been cancelled by the user!"
    SET c_sAddBtnTitle: "Add"
    SET c_sAddCloseBtnTitle: "Add and Close"
    SET c_sDeleteBtnTitle: "Delete"

    SET c_sMsgWinLoadPointerDef: "Reading pointer definition..."
    SET c_sMsgWinLoadModelgroups: "Loadin modelgroups ..."
    SET c_sMsgWinLoadModelgroup: "Loading modelgroup "
    SET c_sMsgWinLoadModel: "Loading model "
    SET c_sMsgWinLoadObject: "Loading object "

    #get information about the object and the model
    CC "Core" GET_OBJ_NAME objid: (idObjectID)
    SET sObjectName: ( objname )
    CC "Core" GET_CLASS_ID objid: (idObjectID)
    SET idObjectClassID: ( classid )
    CC "Core" GET_MODEL_ID objid: (idObjectID)
    SET idCrtModelID: ( modelid )

```

AdoScript code in file selectReferences.asc

```
42
43      #get information about the INTERREF attribute
44      CC "AdoScript" MSGWIN (c_sMsgWinLoadPointerDef)
45      CC "Core" GET_ATTR_ID classid: (idObjectClassID) attrname: (sAttrName)
46      SET idInterrefAttrID: ( attrid )
47      CC "Core" GET_FACET_VAL attrid: (idInterrefAttrID) facetname: "AttributeInterRefDomain"
48      SET sInterrefDomainFacet: ( val )
49
50      PARSE_INTERREF_DOMAIN sInterrefDomain: ( val ) aListOfPairs: aPairList
51      SET sNewOpenModels: ""
52
53      CC "AdoScript" TLB_CREATE title: (c_sDialogTitle) id: (1) text: (c_sDialogText)
54                        x: 150 y: 25 w: 400 h: 500 sizeable: 1 min-w: 200 min-h: 200 max-w: 600 max-h: 800 searchable: 1 sorted: 0 flat: 0 multi-sel: 1
                          no-child-sel: 0 checklistbox: 0 setdblclick: 1 no-help: 1 oktext: (c_sAddBtnTitle)
55
56      CC "AdoScript" MSGWIN (c_sMsgWinLoadModelgroups)
57      CC "Core" GET_ROOT_MODELGROUP_ID
58       # --> RESULT ecode: intValue mgroupid: id .
59      CC "Core" GET_MODELGROUP_CHILDREN mgroupid: ( mgroupid )
60       # --> RESULT ecode: intValue submgroupids: idStr .
61
62      ADD_MODELGROUP_TO_TLB idModelGroupId: ( VAL token ( submgroupids , 0, " ") ) idParentId: 0 aListOfPairs: (aPairList)
63
64      CC "AdoScript" TLB_EXPAND_ALL
65      CC "AdoScript" TLB_ADD_BUTTON text: (c_sAddCloseBtnTitle) name: "end" index: 2 disable_if_no_selection: 1
66      CC "AdoScript" TLB_ADD_BUTTON text: (c_sDeleteBtnTitle) name: "delete" index: 2 disable_if_no_selection: 1
67      SET sEndButton: "ok"
68
69
70      WHILE ((sEndButton != "end")) {
71          SET aqlGetAllRefObjects: ("({\"" + sObjectName + "\"} --> \"" + sAttrName + "\")")
72          CC "AQL" EVAL_AQL_EXPRESSION expr: (aqlGetAllRefObjects) modelid: (idCrtModelID)
73          FOR sRefTopicId in: ( objids ) sep: " " {
74              CC "AdoScript" TLB_SELECT id: ( VAL sRefTopicId) select: 1
75          }
76          CC "AdoScript" TLB_SHOW
77           # --> RESULT ecode: intValue selectedids: idlist [ endbutton: strValue ]
78           # Errorcodes: 0 = No error, 1 = Dialogue aborted by the user, 2 = Before calling this function, TLB_CREATE has to be executed, 3 = An
              argument or parameter is missing,4 = An ID provided is invalid
79          SET sEndButton: ( endbutton )
80
81          IF (sEndButton = "cancel") {
82              # CC "AdoScript" INFOBOX (c_sCancelMsg)
83              EXIT
84          }
```

AdoScript code in file selectReferences.asc

```
85          IF (sEndButton = "delete") {
86              CC "AdoScript" WARNINGBOX (c_sDeleteWarnMsg)  title: (c_sDeleteWarnTitle) yes-no
87              IF ( endbutton = "yes") {
88                  CC "Core" GET_INTERREF_COUNT objid: (idObjectID) attrid: (idInterrefAttrID)
89                   # --> RESULT ecode: intValue count: intValue .
90                  FOR j from: ( count - 1 ) to:  0 by: -1 {
91                      CC "Core" GET_INTERREF objid: (idObjectID) attrid: (idInterrefAttrID) index: (j)
92                       # --> RESULT ecode: intValue tmodeltype: strValue tmodelname: strValue tmodelver: strValue (( type:"modelreference" tmodelid:
                          intValue ) | ( type:"objectreference" tclassname: strValue tobjname: strValue tmodelid: intValue tclassid: intValue tobjid:
                          intValue )) .
93                      IF ( tokindex ( selectedids , STR tobjid , " " ) > -1 ) {
94                          CC "Core" REMOVE_INTERREF objid: (idObjectID) attrid: (idInterrefAttrID) index: (j)
95                          CC "AdoScript" TLB_SELECT id: ( tobjid ) select: 0
96                      }
97                  }
98              }
99          }
100         ELSE {
101             FOR sTopicID in: ( selectedids ) sep: " " {
102                 CC "Core" GET_CLASS_ID objid: ( VAL sTopicID)
103                 CC "Core" GET_CLASS_NAME classid: ( classid )
104                 CC "Core" ADD_INTERREF objid: (idObjectID) attrid: (idInterrefAttrID) tobjid: ( VAL sTopicID)
105             }
106         }
107     }
108
109     FOR sOpenModelID in: (sNewOpenModels) {
110         CC "Core" DISCARD_MODEL modelid: ( VAL sOpenModelID)
111     }
112 }
113
114
115
116 PROCEDURE global PARSE_INTERREF_DOMAIN sInterrefDomain: string aListOfPairs: reference
117 {
118     SETG nCount: ( floor ( tokcnt (sInterrefDomain, "\n") / 4))
119     SETL aCrtListOfPairs: ( set(dummyArray, array(0,2)), dummyArray)
120
121     FOR i from: 0 to: (nCount - 1) {
122         SETL sMTNameRef: ( token ( token (sInterrefDomain, i*4 + 2, "\n"), 1, ":"))
123         SETL sMTNameRef: ( copy (sMTNameRef, 1, LEN sMTNameRef - 2))
124         SETL sClassNameRef: ( token ( token (sInterrefDomain, i*4 + 3, "\n"), 1, ":"))
125         SETL sClassNameRef: ( copy (sClassNameRef, 1, LEN sClassNameRef - 2))
126         SETL dummyArray: (aappend(aCrtListOfPairs,{sMTNameRef,sClassNameRef}))
127     }
128     SET aListOfPairs: (aCrtListOfPairs)
129 }
130
```

AdoScript code in file selectReferences.asc

```
PROCEDURE global ADD_MODELGROUP_TO_TLB idModelGroupId: integer idParentId: integer aListOfPairs: array
{
    CC "Core" GET_MODELGROUP_NAME mgroupid: (idModelGroupId)
     # --> RESULT ecode: intValue mgroupname: strValue
    CC "AdoScript" MSGWIN (c_sMsgWinLoadModelgroup + mgroupname )
    IF (idParentId = 0) {
        CC "AdoScript" TLB_INSERT id: (idModelGroupId) text: ( mgroupname ) is-parent: 1
    }
    ELSE {
        CC "AdoScript" TLB_INSERT id: (idModelGroupId) text: ( mgroupname ) parentid: (idParentId) is-parent: 1
    }

    CC "Core" GET_MODELGROUP_CHILDREN mgroupid: (idModelGroupId)
     # --> RESULT ecode: intValue submgroupids: idStr .
    FOR sSubMGroupId in: ( submgroupids ) sep: " " {
        ADD_MODELGROUP_TO_TLB idModelGroupId: ( VAL sSubMGroupId) idParentId: (idModelGroupId) aListOfPairs: (aListOfPairs)
    }

    CC "Core" GET_MODELGROUP_MODELS mgroupid: (idModelGroupId)
     # --> RESULT ecode: intValue modelids: idlist .
    FOR sModelId in: ( modelids ) sep: " " {
        ADD_MODEL_TO_TLB idModelThreadId: ( VAL sModelId) idParentId: (idModelGroupId) aListOfPairs: (aListOfPairs)
    }
}



PROCEDURE global ADD_MODEL_TO_TLB idModelThreadId: integer idParentId: integer aListOfPairs: array
{
    CC "Core" GET_ALL_MODEL_VERSIONS_OF_THREAD modelthreadid: (idModelThreadId)
     # --> RESULT ecode: intValue modelversionids: ids
    SET idModelId: ( VAL token ( modelversionids , 0, " "))
    CC "Core" GET_MODEL_INFO modelid: (idModelId)
     # --> RESULT ecode: intValue modelname: strValue ver: strValue version: strValue threadid: id modeltype: strValue libid: id libname: strValue
     access: "none" | "read" | "write"

    IF ( refContainsMT(aListOfPairs, modeltype ) ) {
        CC "AdoScript" MSGWIN (c_sMsgWinLoadModel + modelname )
        CC "Core" IS_MODEL_LOADED modelid: (idModelId)
        IF ( NOT isloaded ) {
            CC "Core" LOAD_MODEL modelid: (idModelId)
            SET sNewOpenModels: ( tokunion (sNewOpenModels, STR idModelId, " ") )
        }

        CC "Core" GET_MODEL_INFO modelid: (idModelId)
         # --> RESULT ecode: intValue modelname: strValue ver: strValue version: strValue threadid: id modeltype: strValue libid: id libname:
         strValue access: "none" | "read" | "write
```

AdoScript code in file selectReferences.asc

```
178
179         SETL sIconFileName: ("db:\\icon_mt_" + modeltype + ".gif")
180         CC "AdoScript" FILE_EXISTS file: (sIconFileName)
181          # --> RESULT exists: boolValue .
182         IF ( NOT exists ) {
183             CC "Drawing" GEN_MODELTYPE_ICON_STR modeltype: ( modeltype ) gfx-format: "gif"
184              # --> RESULT ecode: intValue gfx: strValue
185             CC "AdoScript" FWRITE file: (sIconFileName) text: ( gfx ) append: 0 binary: 0 base64: 1
186              # --> RESULT ecode: intValue written: intValue
187         }
188
189         CC "AdoScript" TLB_INSERT id: (idModelId) text: ( modelname ) parentid: (idParentId) is-parent: 1 bitmap: (sIconFileName) bitmap2:
            (sIconFileName)
190
191         CC "Core" GET_ALL_OBJS modelid: (idModelId)
192          # --> RESULT ecode: intValue objids: list .
193
194         FOR sObjID in: ( objids ) {
195             SET idObjID: ( VAL sObjID)
196             CC "Core" GET_OBJ_NAME objid: (idObjID)
197             #generate a hierarchy for the current model
198             SET aqlGetAllParents: ("({\"" + objname + "\"} -> \"belongs to\")")
199             CC "AQL" EVAL_AQL_EXPRESSION expr: (aqlGetAllParents) modelid: (idModelId)
200             IF ( LEN objids = 0) {
201                 ADD_OBJECT_TO_TLB idObjId: ( VAL sObjID) idParentId: (idModelId) idModelId: (idModelId) aListOfPairs: (aListOfPairs)
202             }
203             #end hierarchy
204             #if no hierarchical display is required, just remove the hierarchy block above
205         }
206     }
207 }
208
209
210
```

AdoScript code in file selectReferences.asc

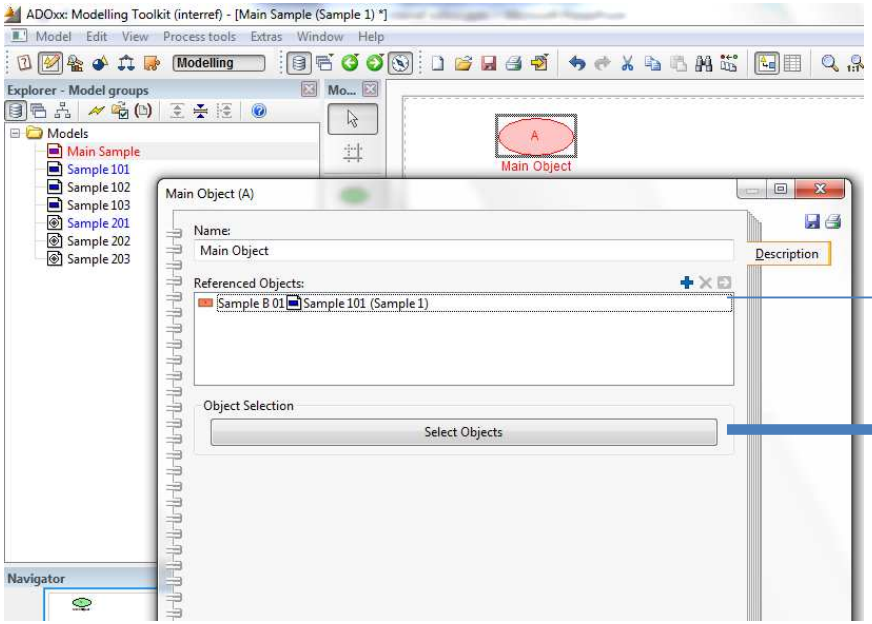
```
211  PROCEDURE global ADD_OBJECT_TO_TLB idObjId: integer idParentId: integer idModelId: integer aListOfPairs: array
212  {
213      CC "Core" GET_MODEL_INFO modelid: (idModelId)
214      SETL sMTName: ( modeltype )
215      CC "Core" GET_OBJ_NAME objid: (idObjId)
216      SETL sObjName: ( objname )
217      CC "Core" GET_CLASS_ID objid: (idObjId)
218      SETL idClassID: ( classid )
219      CC "Core" GET_CLASS_NAME classid: ( idClassID )
220      SETL sClassName: ( classname )
221      IF ( refContainsCLS(aListOfPairs, sMTName , sClassName )) {
222          SETL sIconFileName: ("db:\\icon_cl_" + sClassName + ".gif")
223          CC "AdoScript" FILE_EXISTS file: (sIconFileName)
224           # --> RESULT exists: boolValue .
225          IF ( NOT exists ) {
226              CC "Drawing" GEN_CLASS_ICON_STR classid: ( idClassID ) gfx-format: "gif" w: (16) h: (16)
227               # --> RESULT ecode: intValue gfx: strValue
228              CC "AdoScript" FWRITE file: (sIconFileName) text: ( gfx ) append: 0 binary: 0 base64: 1
229               # --> RESULT ecode: intValue written: intValue
230          }
231          CC "AdoScript" MSGWIN (c_sMsgWinLoadObject + sObjName )
232          CC "AdoScript" TLB_INSERT id: (idObjId) text: ( sObjName ) parentid: (idParentId) bitmap: (sIconFileName) bitmap2: (sIconFileName)
233
234          #generate hierarchy "below" this object
235          SET aqlGetAllChildren: ("({\"" + sObjName + "\":\"" + sClassName + "\"} <- \"belongs to\")")
236          CC "AQL" EVAL_AQL_EXPRESSION expr: (aqlGetAllChildren) modelid: (idModelId)
237          FOR sChildID in: ( objids ) sep: " " {
238
239              ADD_OBJECT_TO_TLB idObjId: ( VAL sChildID) idParentId: (idObjId) idModelId: (idModelId) aListOfPairs: (aListOfPairs)
240          }
241          #end hierarchy
242          #if no hierarchical display is required, just remove the block above
243          #hierarchical display may be extended using AQL expressions
244      }
245  }
246
247
248
```

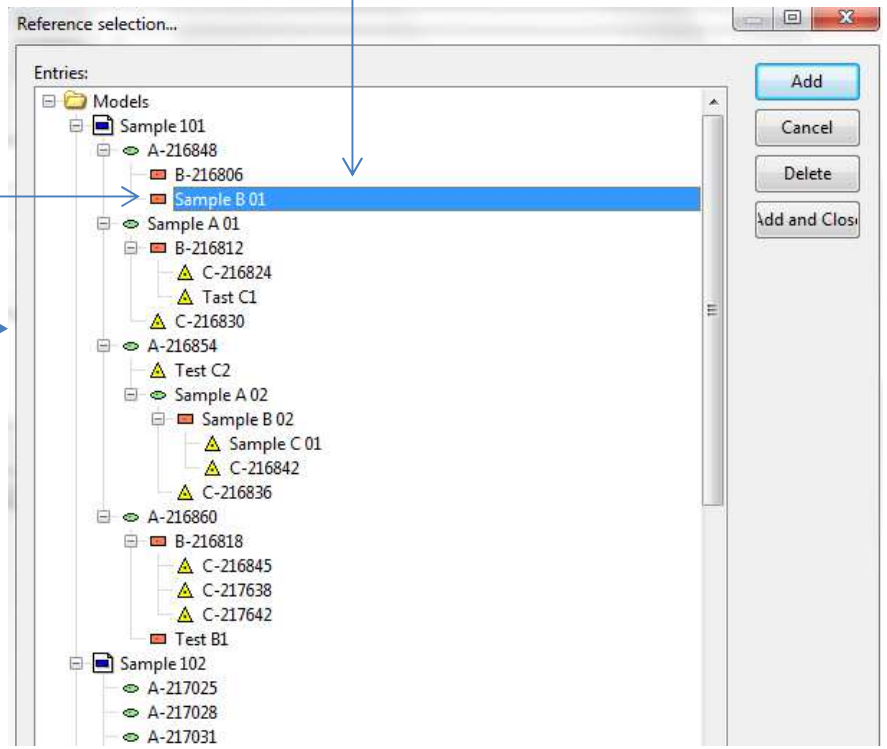
AdoScript code in file selectReferences.asc

```
249  FUNCTION refContainsMT: global aListOfPairs: array sMTName: string
250      return: (set(bContains,0),
251               set(n, (LEN (aListOfPairs) - 1)),
252               for(i, 0, n, (set(bContains, cond(aListOfPairs[i,0] = (sMTName), 1, bContains)))),
253               bContains)
254
255  FUNCTION refContainsCLS: global aListOfPairs: array sMTName: string sCLSName: string
256      return: (set(bContains,0),
257               set(n, aListOfPairs.length - 1),
258               for(i, 0, n, (set(bContains, cond(((aListOfPairs[i,0]=sMTName) AND (aListOfPairs[i,1]=sCLSName)), 1, bContains)))),
259               bContains)
260
```

Result

When opening the *Interref Editor*, all already existing references are automatically selected.





The *Add* button adds references pointing to selected objects.
If a model or a model group is selected, nothing happens.

The „Cancel“ button closes the dialog without adding the last selection.

The *Delete* button removes existing references to selected objects.

The *Add and Close* button adds references for the selected objects and closes the dialog.

Further Questions?

www.adoxx.org

tutorial@adoxx.org