

ADOxx: A Meta-Modeling Platform

3 Jan 2020

Prof. Moon Kun Lee

**Chonbuk National Univ.
Republic of Korea**

Contents

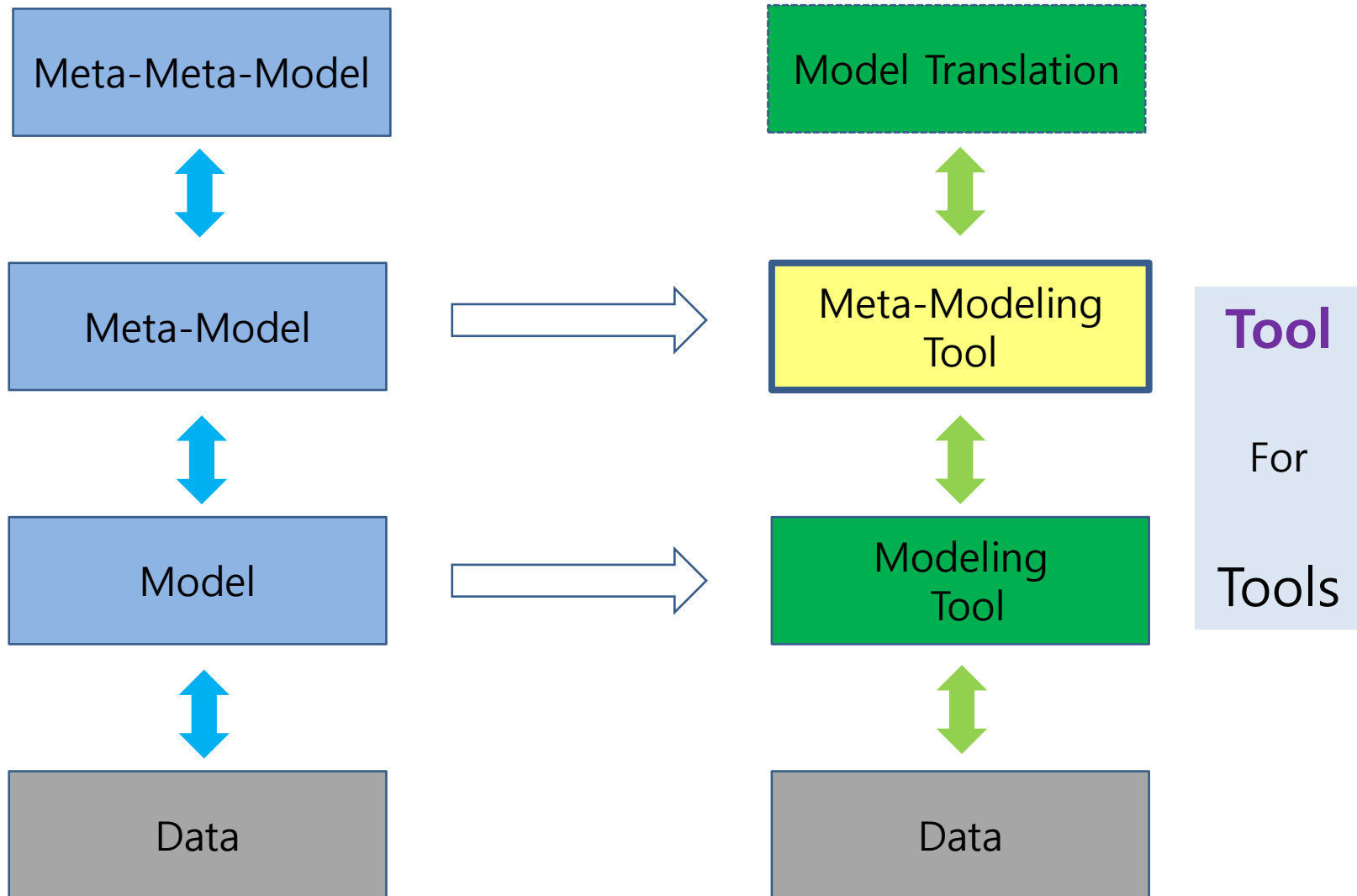
1. Overview

- 1) Meta Model
- 2) ADOxx

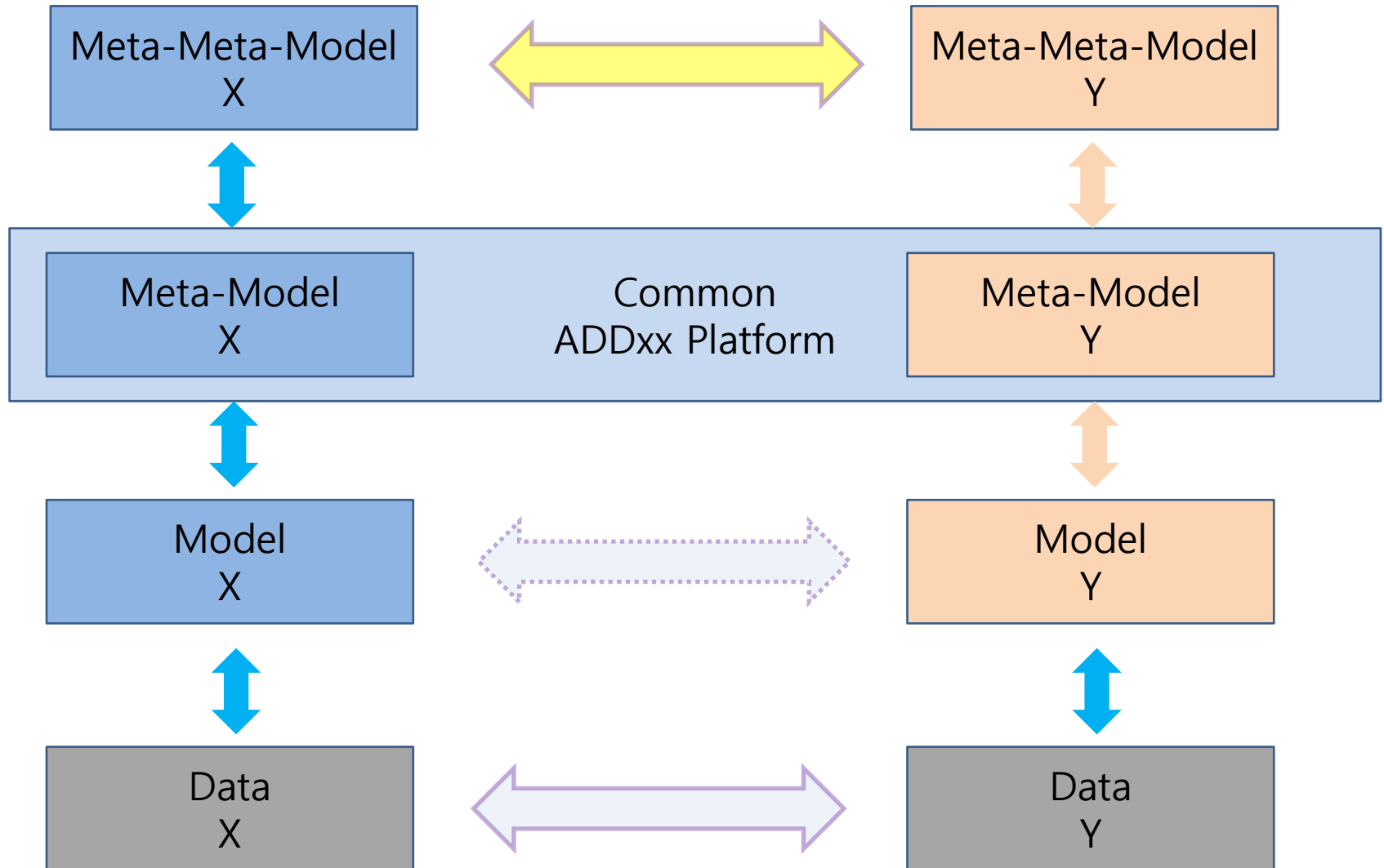
2. Details

- 1) ADOxx Toolkit
- 2) Modeling Language
- 3) Core Functions
- 4) Adoscript
- 5) Simulation

Modeling Hierarchy



Model Transformation/Translation



1. Overview

- 1) Meta Model
- 2) ADOxx

2. Details

- 1) ADOxx Toolkit
- 2) Modeling Language
- 3) Core Functions
- 4) Adoscript
- 5) Simulation

1. OVERVIEW

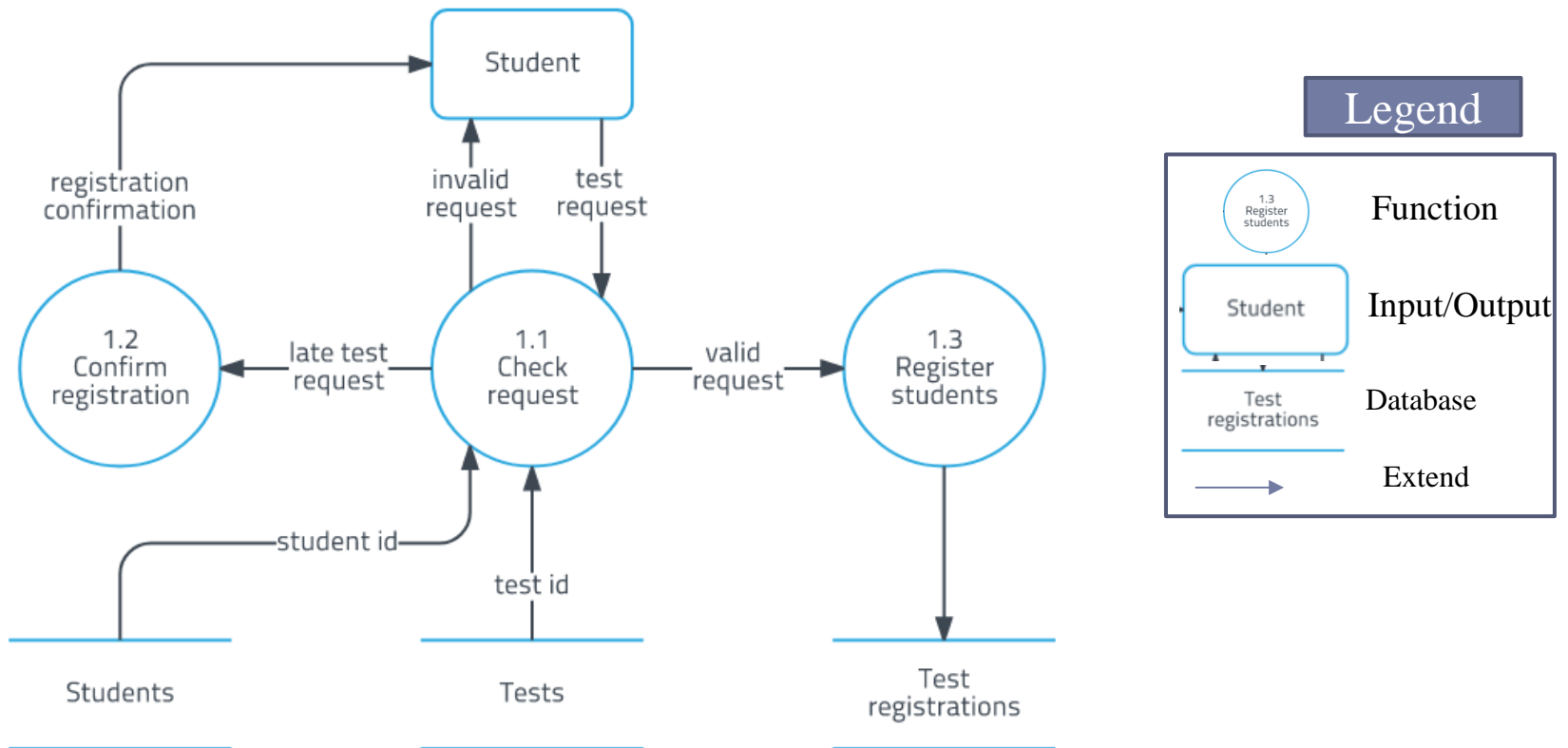
1. Overview
 - 1) **Meta Model**
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language
 - 3) Core Functions
 - 4) Adoscript
 - 5) Simulation

1) META MODEL

Meta model

► Example

► Development of Data Flow Diagram modeling tool

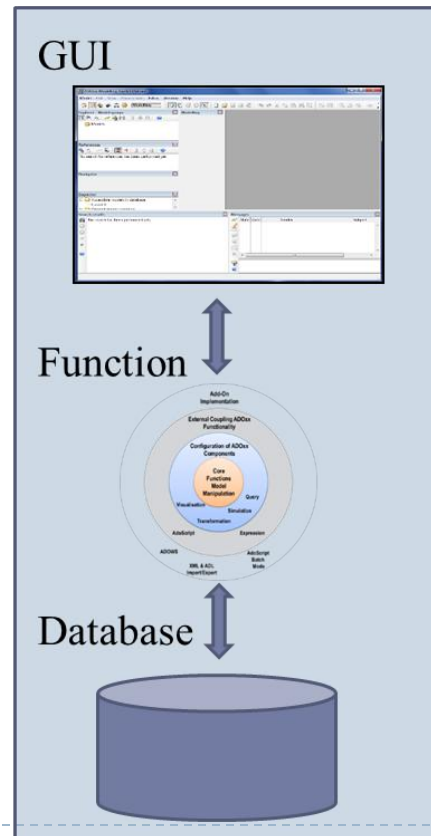


Meta model

► Example

- Development of Data Flow Diagram modeling tool
 - GUI
 - Function
 - Database

Tool



Meta model

► Example

► Development of Data Flow Diagram modeling tool

► GUI

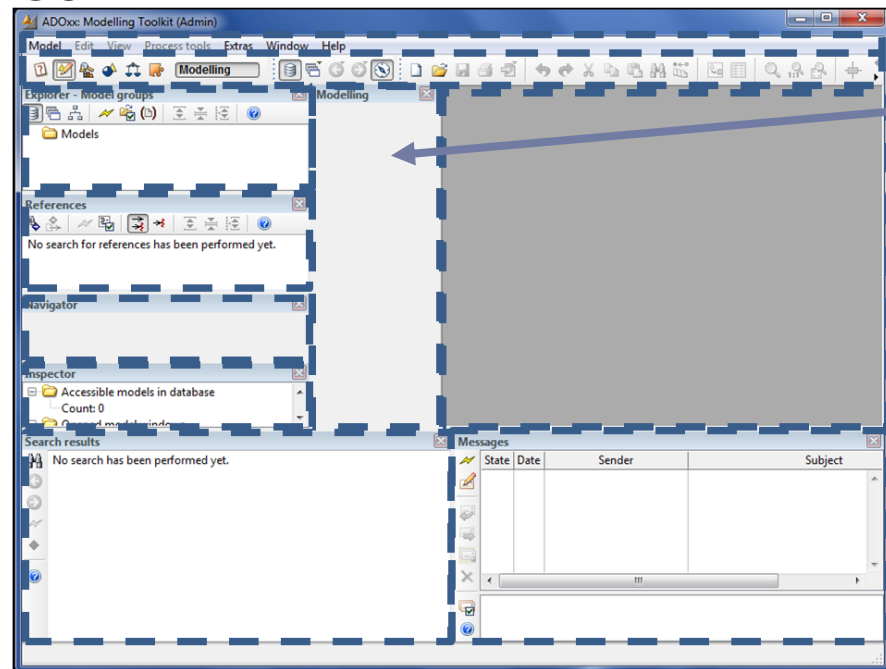
- ☐ Menubar
- ☐ Actionbar
- ☐ Explorer
- ☐ References
- ☐ Navigator
- ☐ Inspector
- ☐ Search Result
- ☐ ...

► Function

► Database

Menubar
Actionbar
Explorer
References
Navigator
Inspector
Search Results

GUI



Modelling

Drawing Area

Messages

Meta model

▶ Example

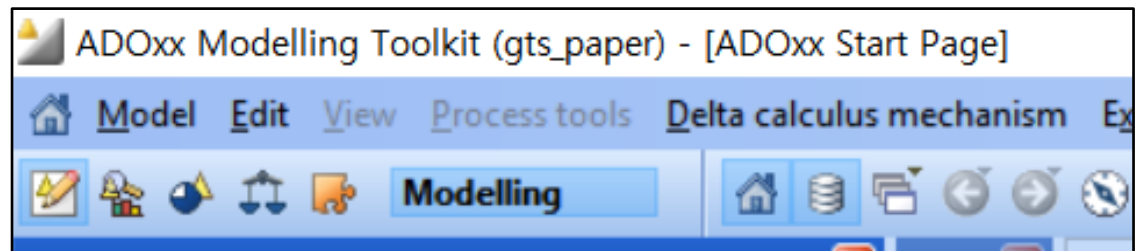
▶ Development of Data Flow Diagram modeling tool

▶ GUI

- Menubar
- Actionbar
 - Basic components of modeling tool

▶ Function

▶ Database



Meta model

► Example

► Development of Data Flow Diagram modeling tool

► GUI

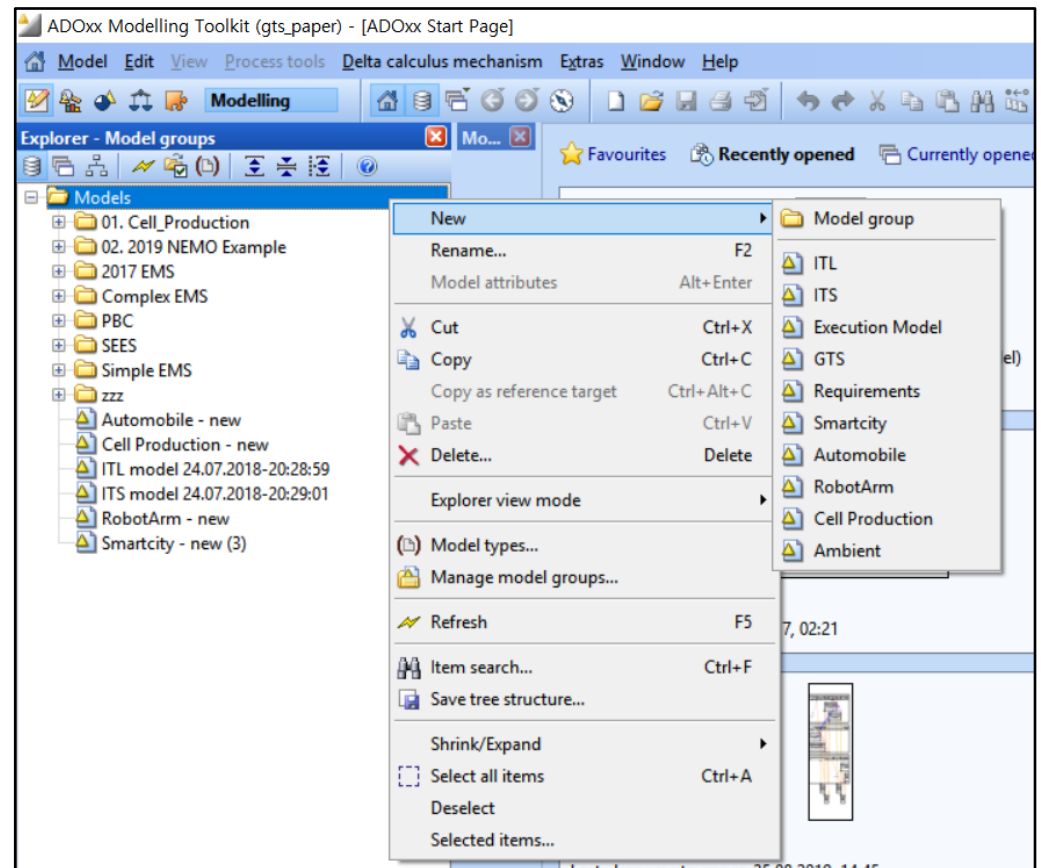
□ Explorer

□ Manage models

- Create model
- Delete model
- Model group
- ...

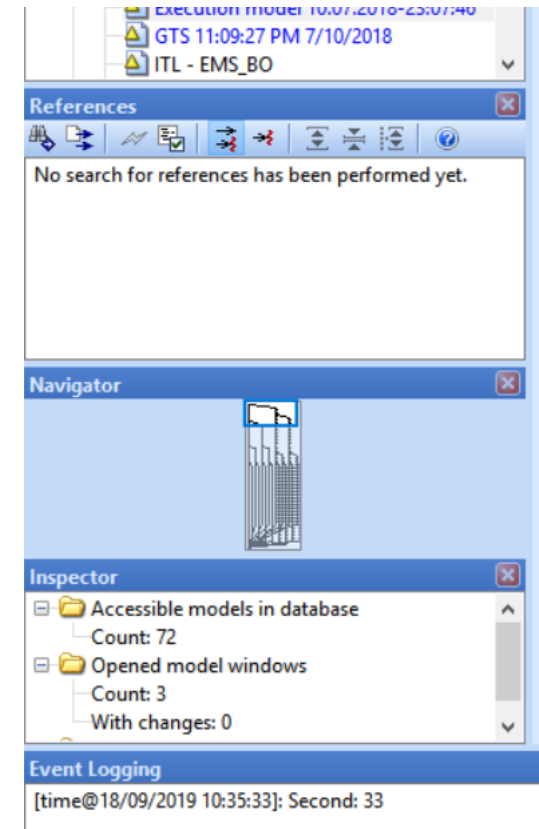
► Function

► Database



Meta model

- ▶ Example
 - ▶ Development of Data Flow Diagram modeling tool
 - ▶ GUI
 - ☐ References
 - ☐ Show inter-model references
 - ☐ Navigator
 - ☐ Inspector
 - ☐ The user with various details about the active modelling environment.
 - ▶ Function
 - ▶ Database



Meta model

► Example

► Development of Data Flow Diagram modeling tool

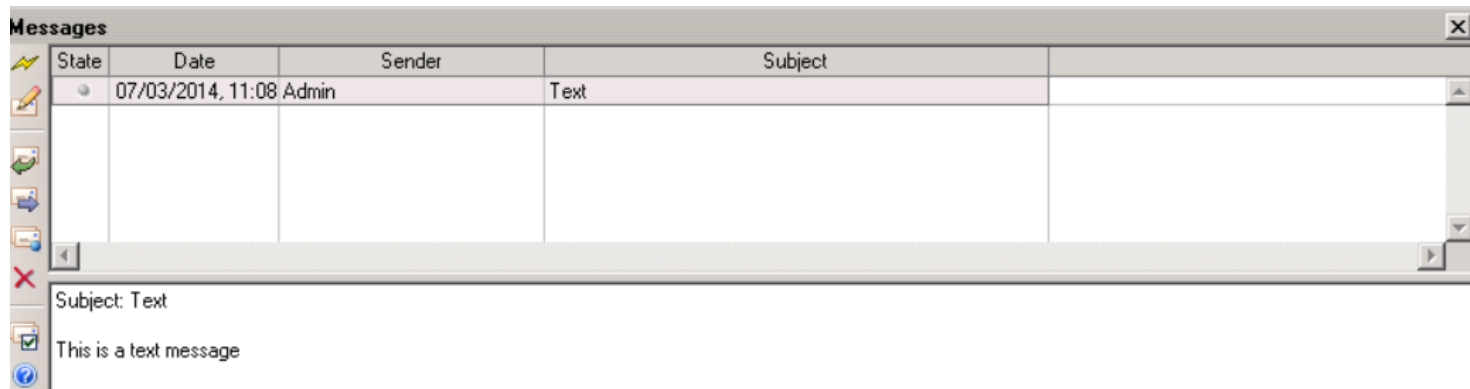
► GUI

☐ Message

- ☐ Send message to other ADOxx users directly from within ADOxx

► Function

► Database



Meta model

► Example

► Development of Data Flow Diagram modeling tool

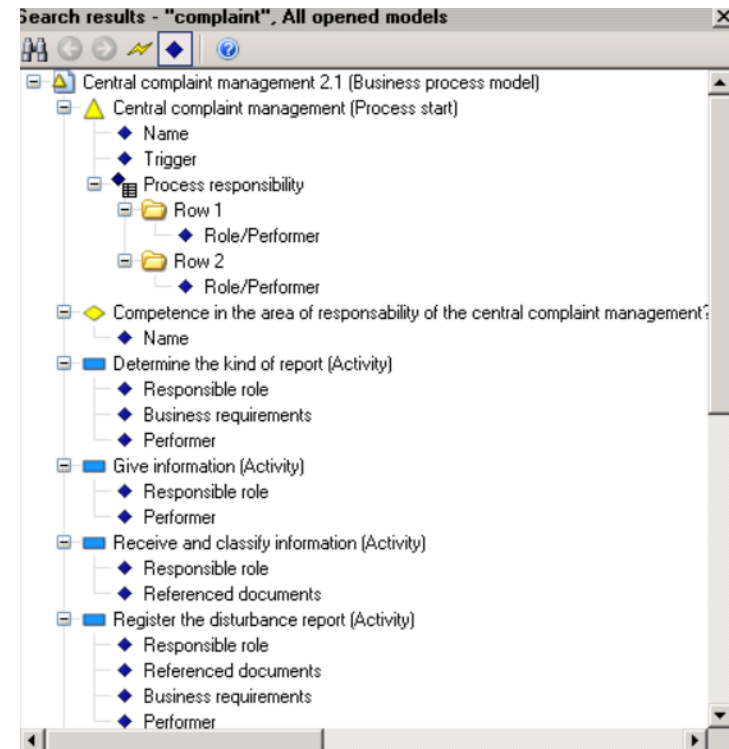
► GUI

☐ Search results

☐ Search specific text in the model or object

► Function

► Database



Meta model

► Example

► Development of Data Flow Diagram modeling tool

► GUI

- Modeling

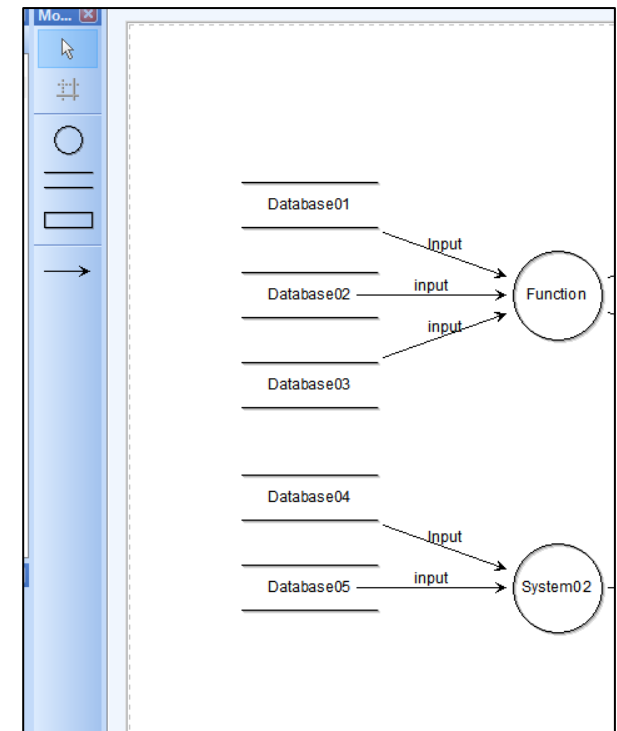
- Show the class and relation panel

- Drawing area

- Create the object and relation in Drawing area

► Function

► Database



Meta model

▶ Example

▶ Development of Data Flow Diagram modeling tool

▶ GUI

□ Drawing area

□ Object

▶ Resize

▶ Shape

▶ ...

□ Relation

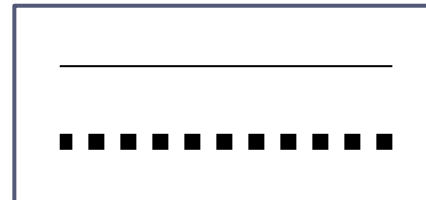
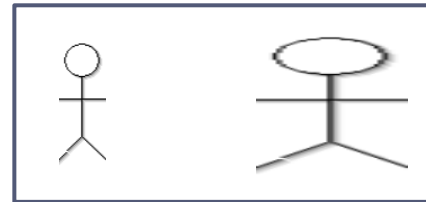
▶ Thickness

▶ Linetype

▶ ...

▶ Function

▶ Database



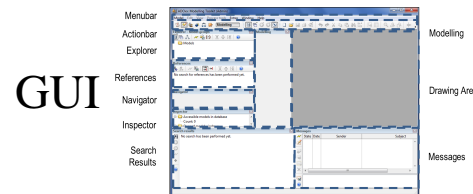
Meta model

► Example

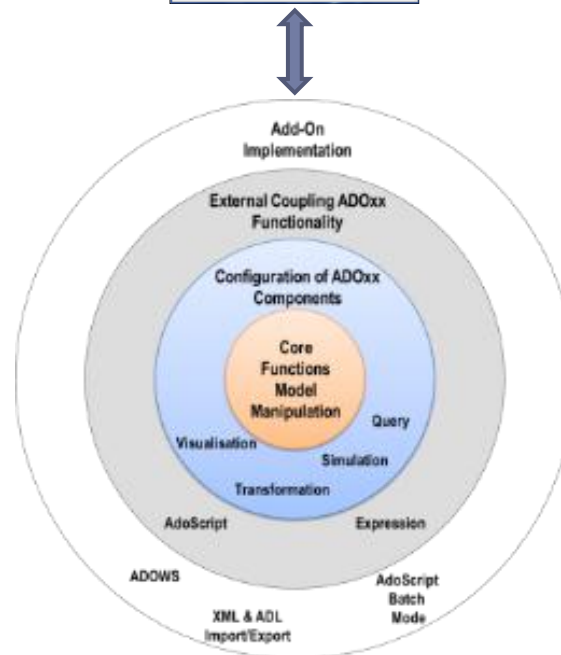
► Development of Data Flow Diagram modeling tool

► Function(Engine)

- ☐ Support GUI
 - ☐ Modeling
 - ☐ Simulation
 - ☐ Analysis
 - ☐ Etc.



Components/
Functions



Meta model

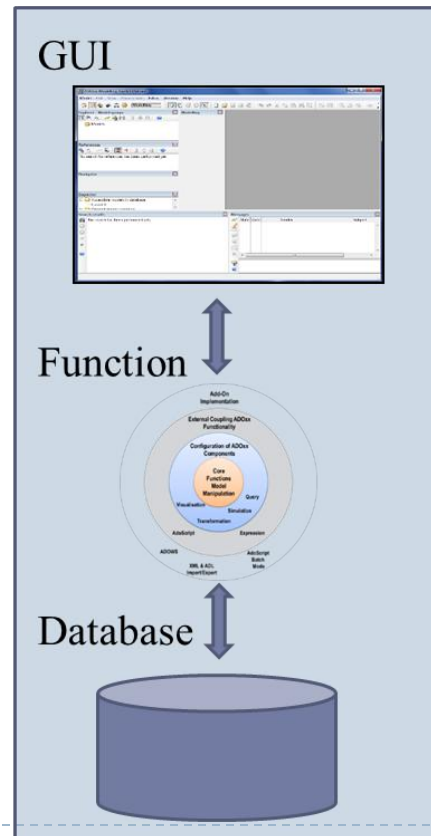
► Example

► Development of Data Flow Diagram modeling tool

► Database

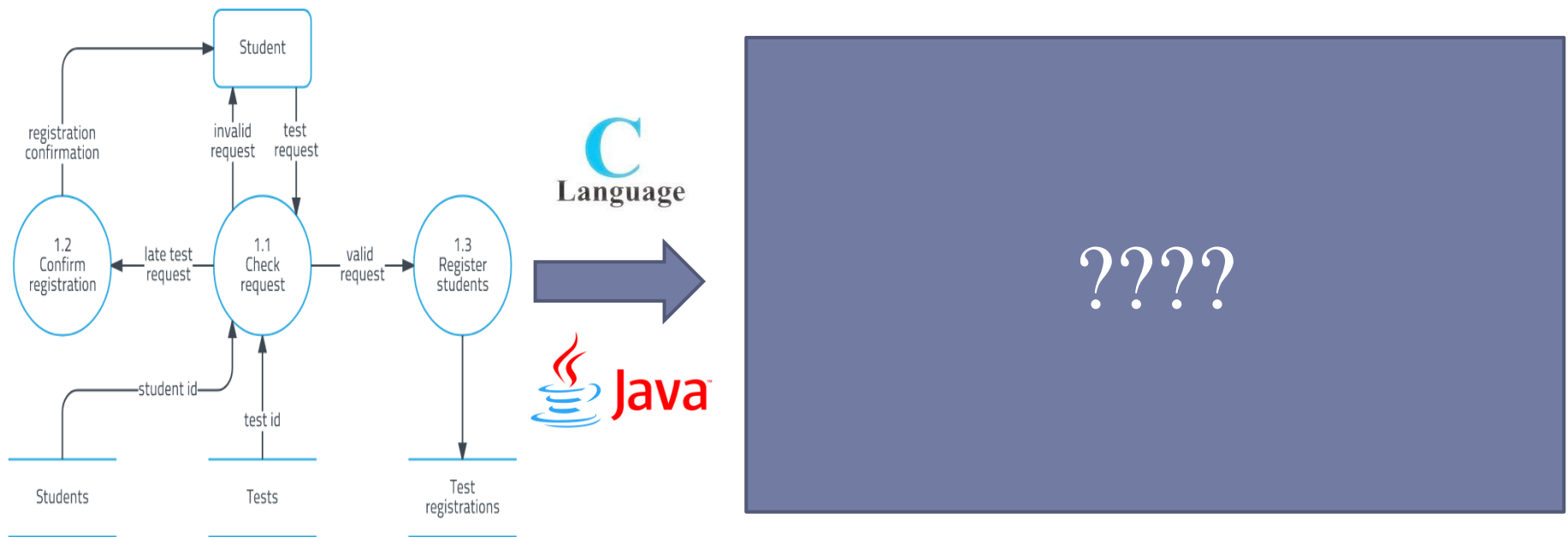
- ☐ Store data
- ☐ Load data
- ☐ Query
- ☐ ...

Tool



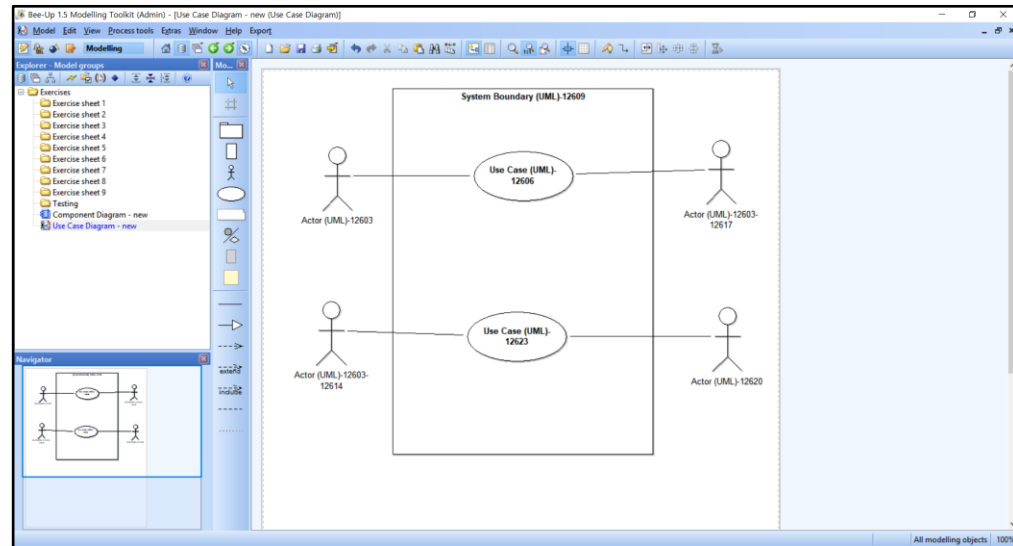
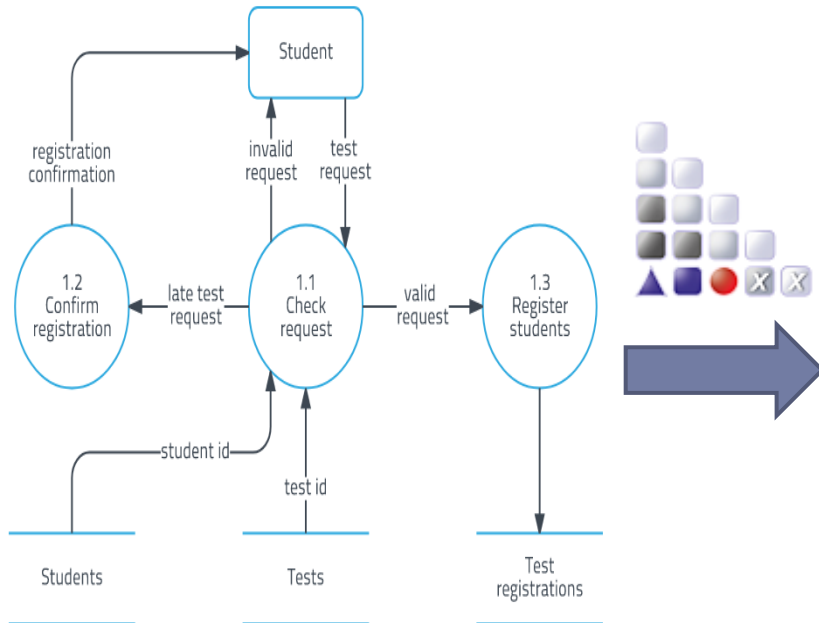
Meta model

- Can you develop these tools using C or JAVA?

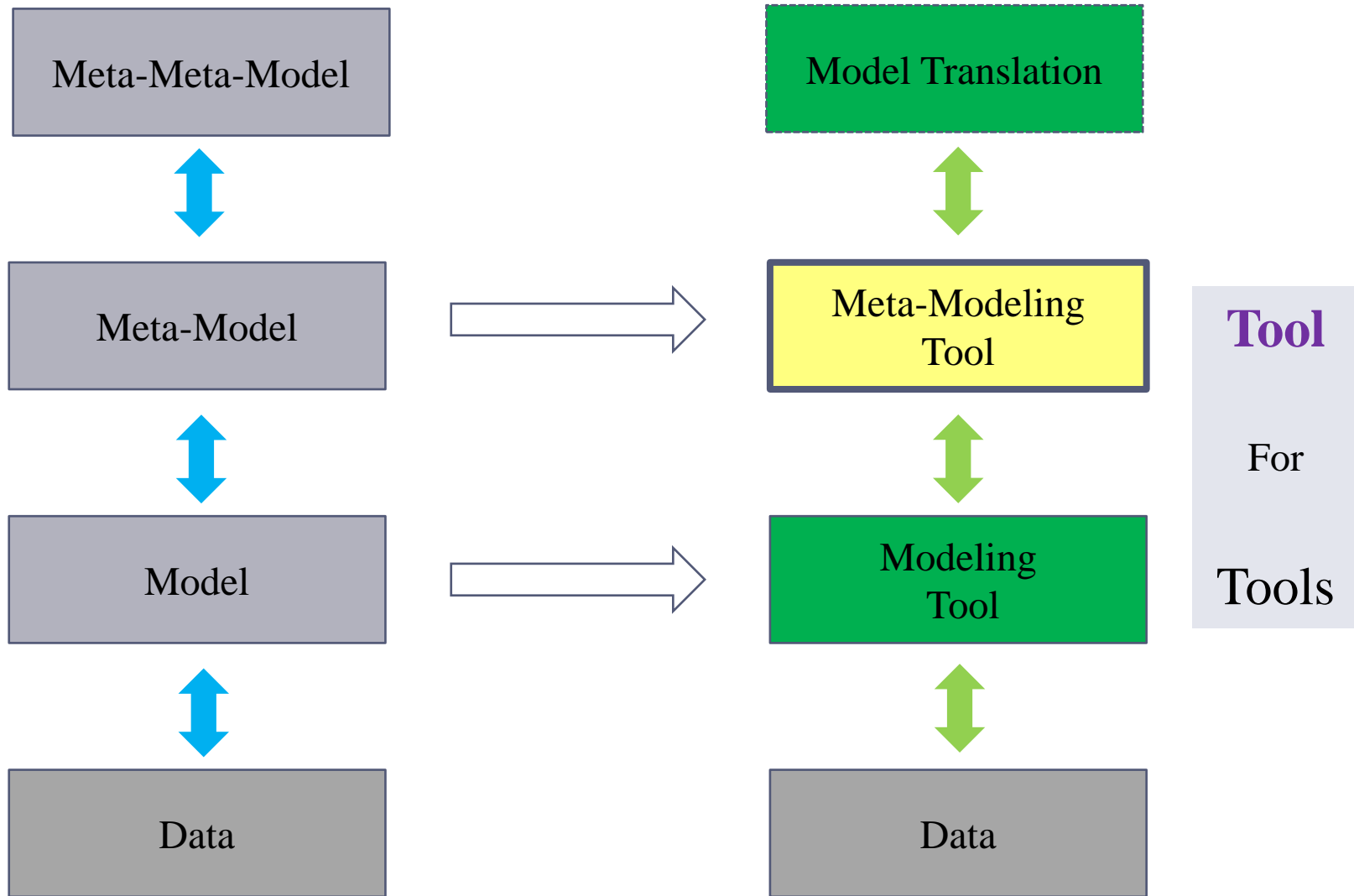


Meta model

► Modeling tool with ADOxx



Modeling Hierarchy



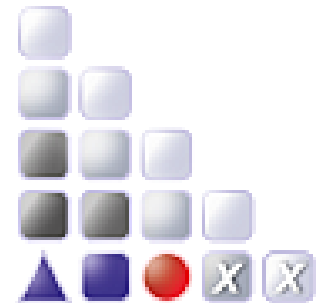
1. Overview
 - 1) Meta Model
 - 2) ADOxx**
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language
 - 3) Core Functions
 - 4) Adoscript
 - 5) Simulation

2)

ADOXX

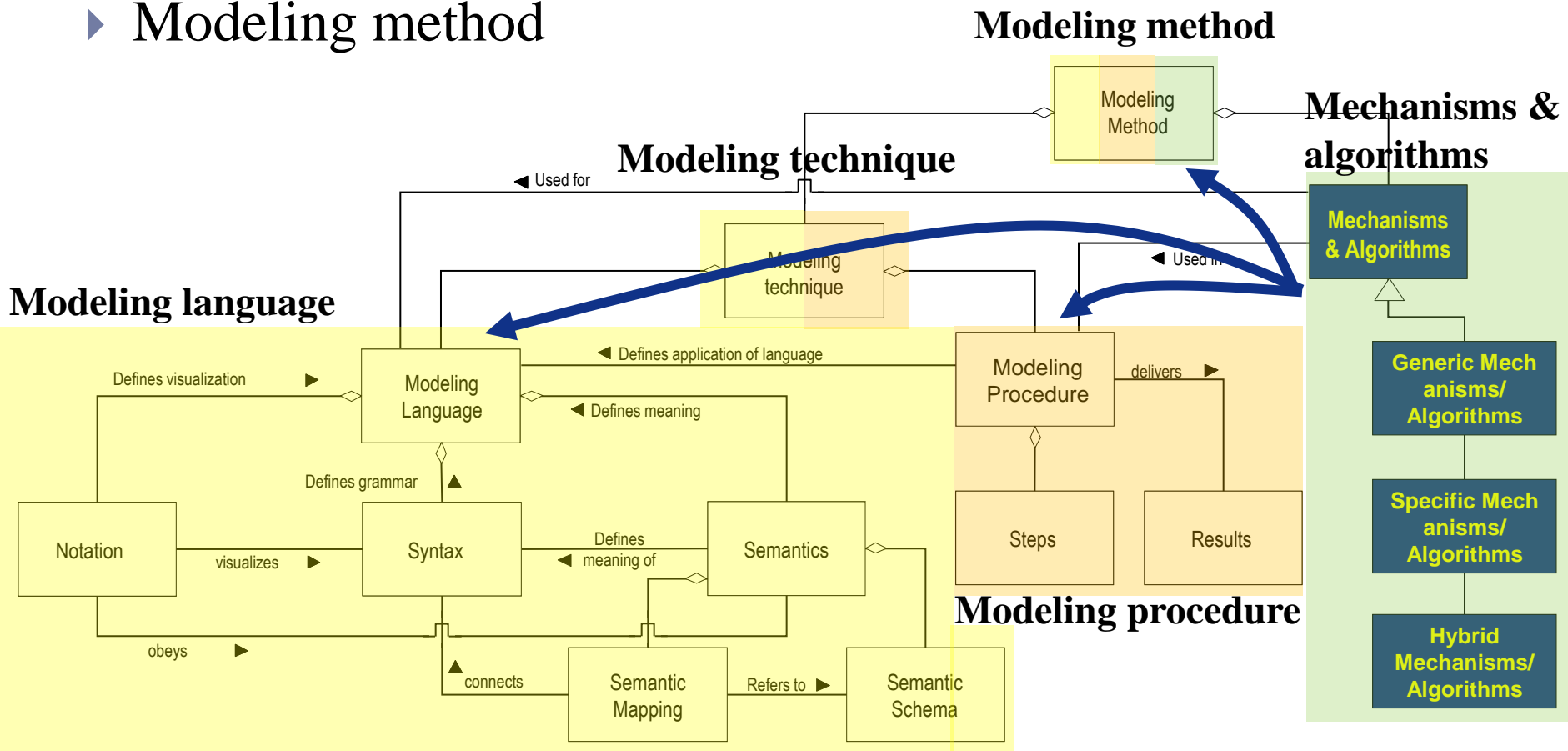
ADOxx

- ▶ Meta Modeling Platform developed by OMiLAB in University of Vienna
- ▶ **Tool to create modeling tool** based on meta model
 - ▶ Tool to define a meta model and to create a model based on the defined meta model
- ▶ Tool to develop a **modeling method**



ADOxx

► Modeling method

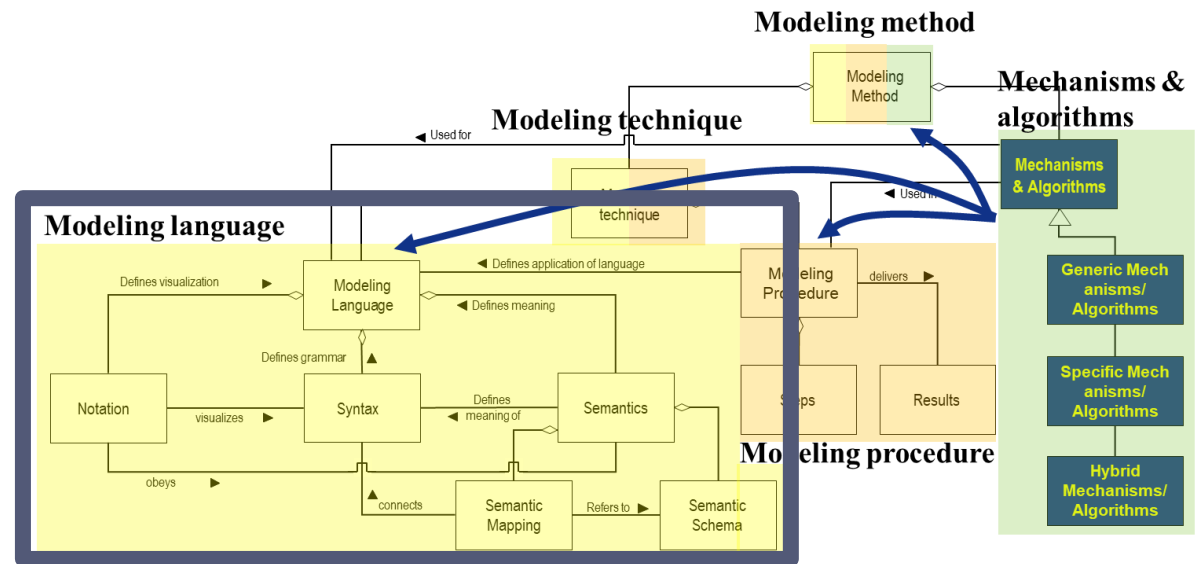


Karagiannis, D., Kühn, H.: „Metamodelling Platforms“. In Bauknecht, K., Min Tjoa, A., Quirmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2002, LNCS 2455, Springer, Berlin/Heidelberg, p. 182.

ADOxx

► Modeling language

- Modeling constructs and their relations to each other to declare a model
- Consist of 3 entities
 - Notation
 - Syntax
 - Semantics



Karagiannis, D., Kühn, H.: „Metamodelling Platforms“. In Bauknecht, K., Min Tjoa, A., Quirmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2002, LNCS 2455, Springer, Berlin/Heidelberg, p. 182.

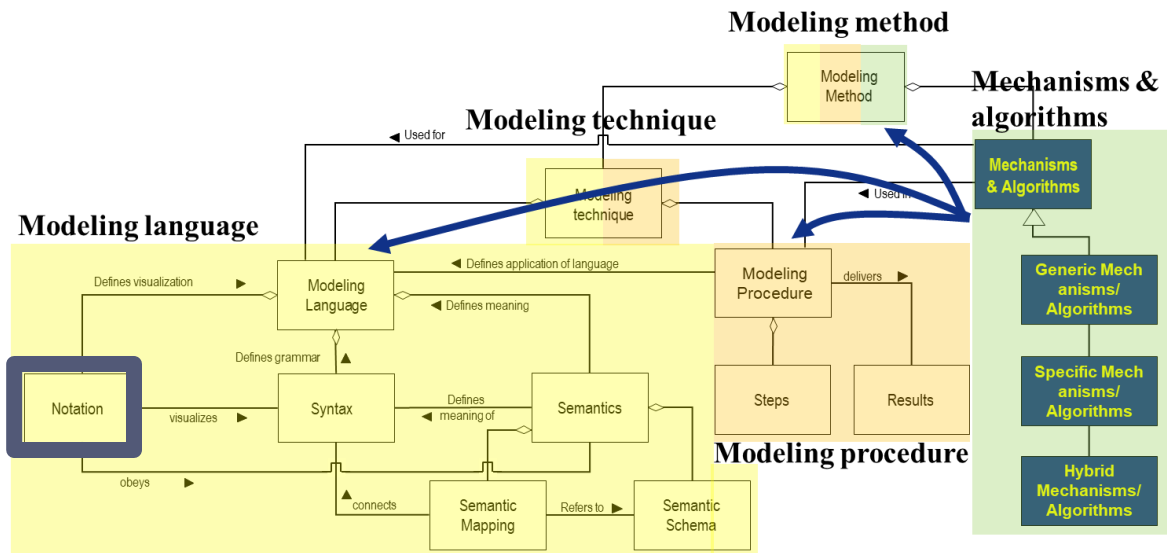
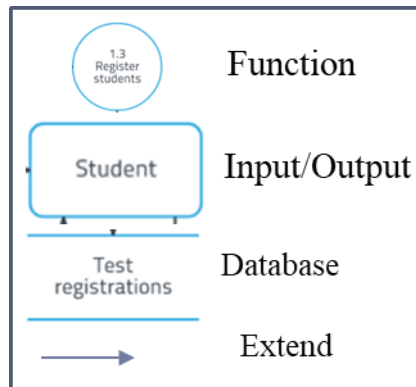
ADOxx

► Modeling language

► Notation

► Describe the visualization of a modelling language.

► ex)

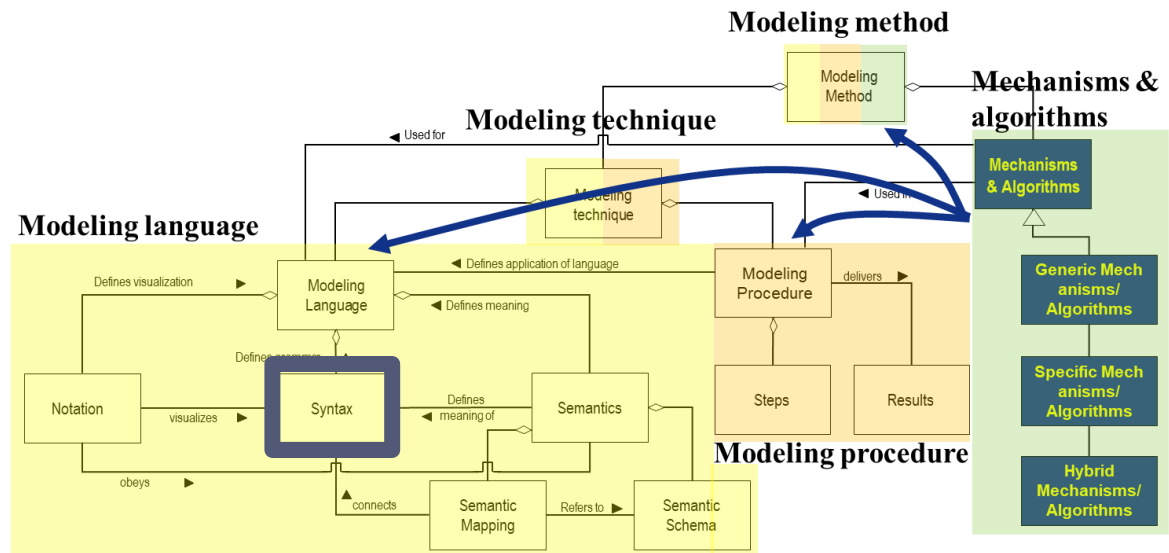


ADOxx

► Modeling language

► Syntax

- Describe the elements and rules for creating models and is described by a grammar

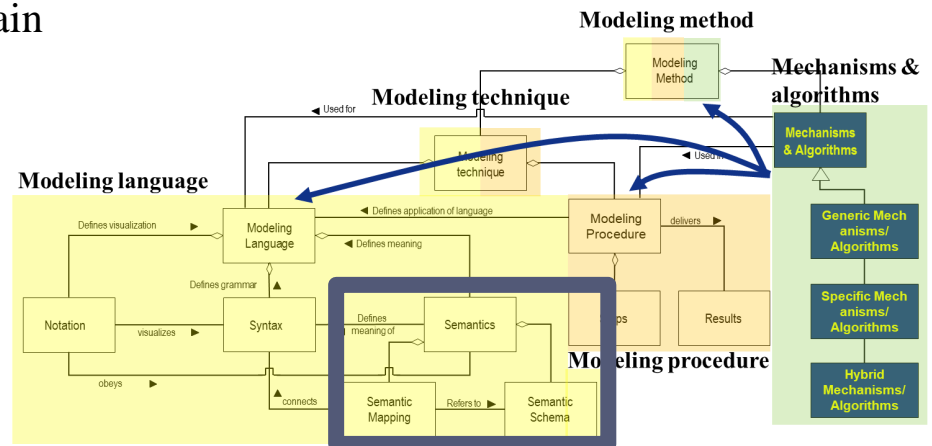


ADOxx

► Modeling language

► Semantics

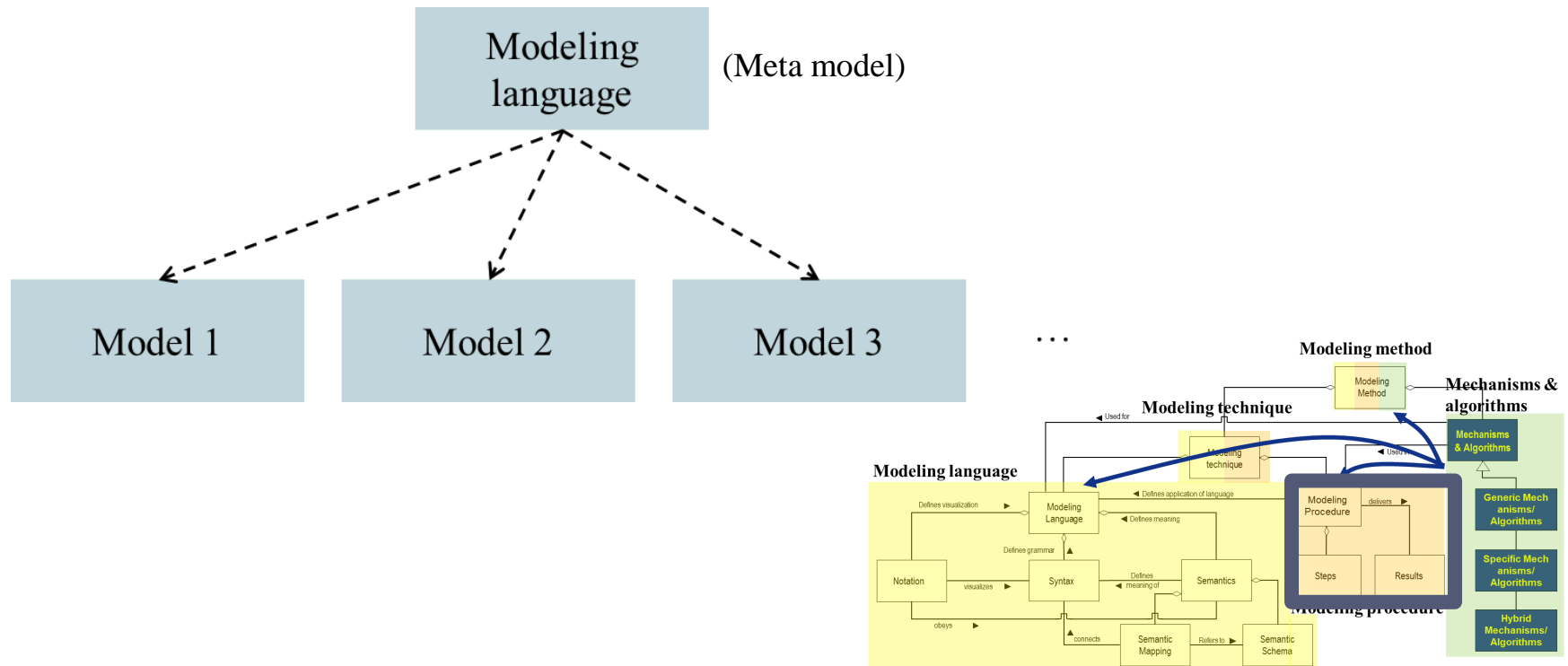
- Describe the meaning of a modelling language
- Consists of a **semantic domain** and a **semantic mapping**.
 - Semantic domain
 - describe the meaning by using ontologies, mathematical expressions
 - Semantic mapping
 - The semantic mapping connects the syntactical constructs with their meaning defined in the semantic domain



ADOxx

► Modeling Procedure

- describe the **steps** applying the modelling language to create **results**

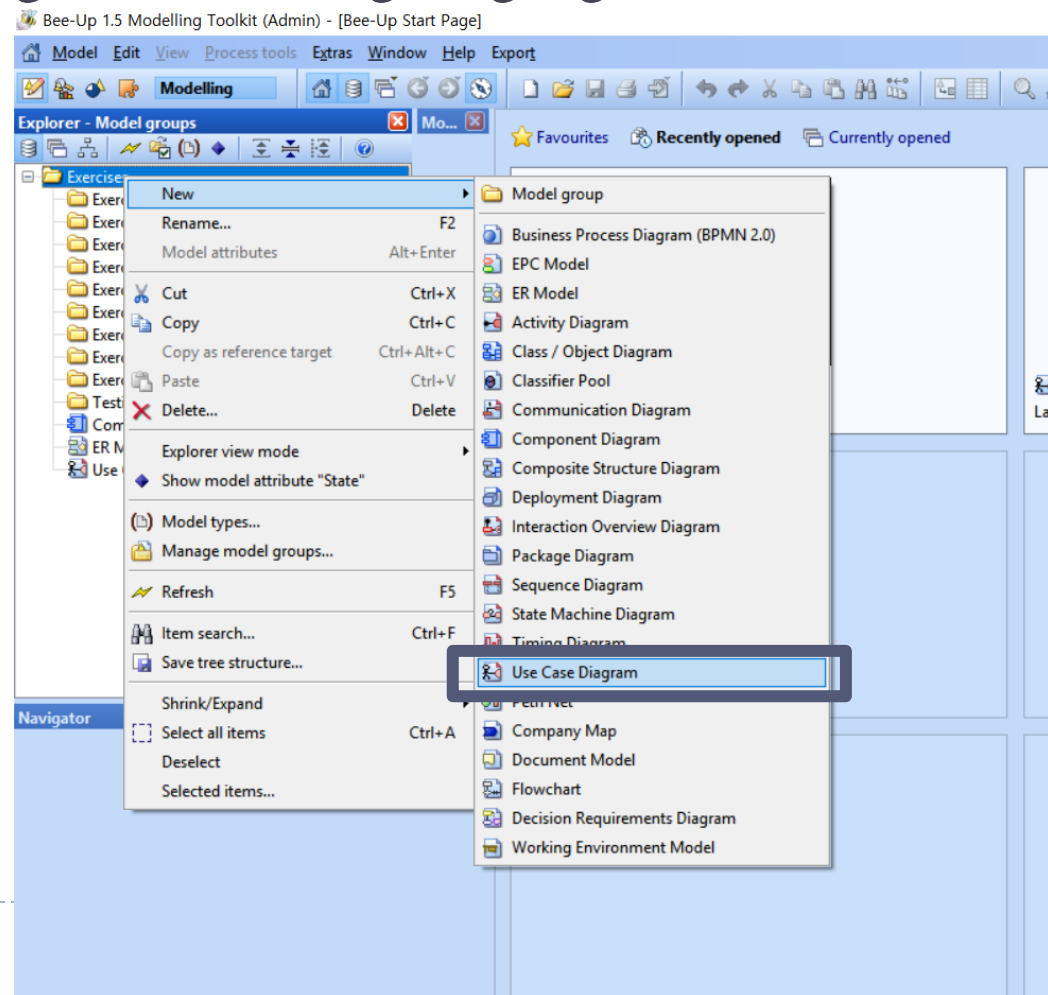


ADOxx

► Modeling Procedure

- describe the **steps** applying the modelling language to create **results**

- Create new model

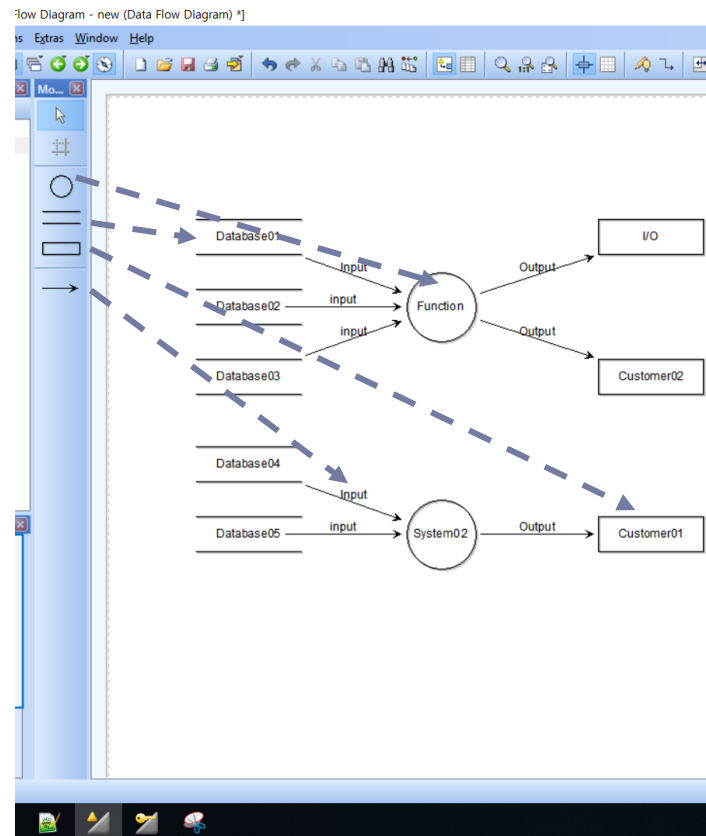


ADOxx

► Modeling Procedure

- describe the **steps** applying the modelling language to create **results**

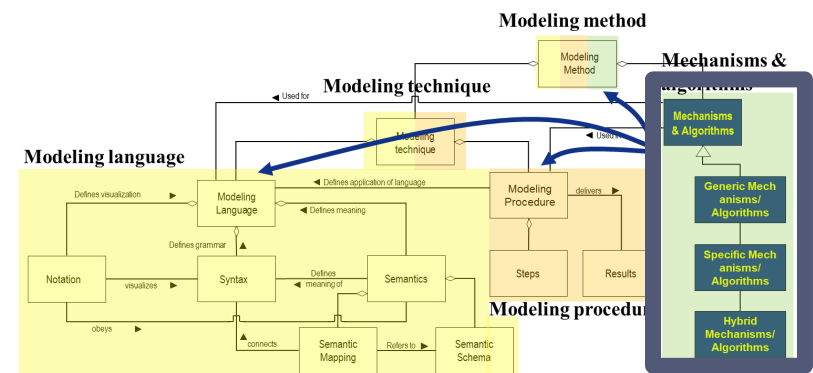
- Modeling
 - ☐ Function
 - ☐ I/O
 - ☐ Database
 - ☐ Flow



ADOxx

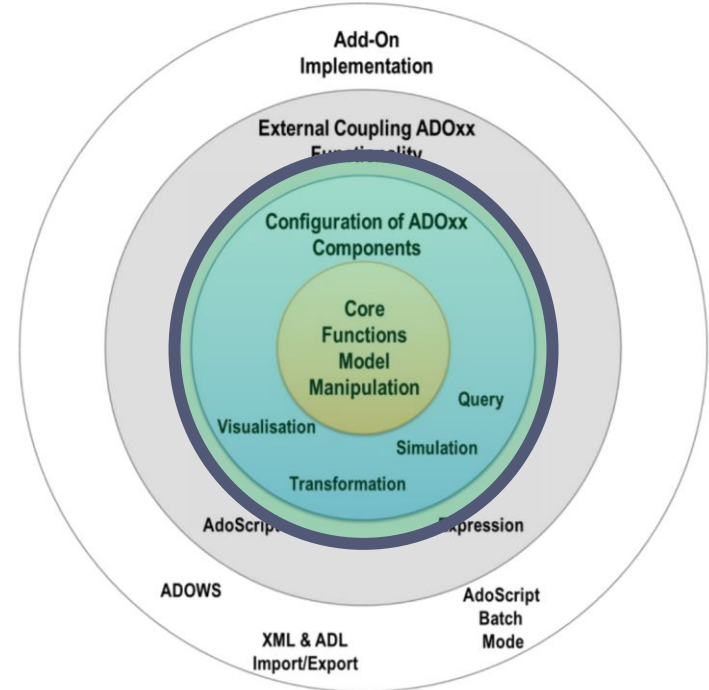
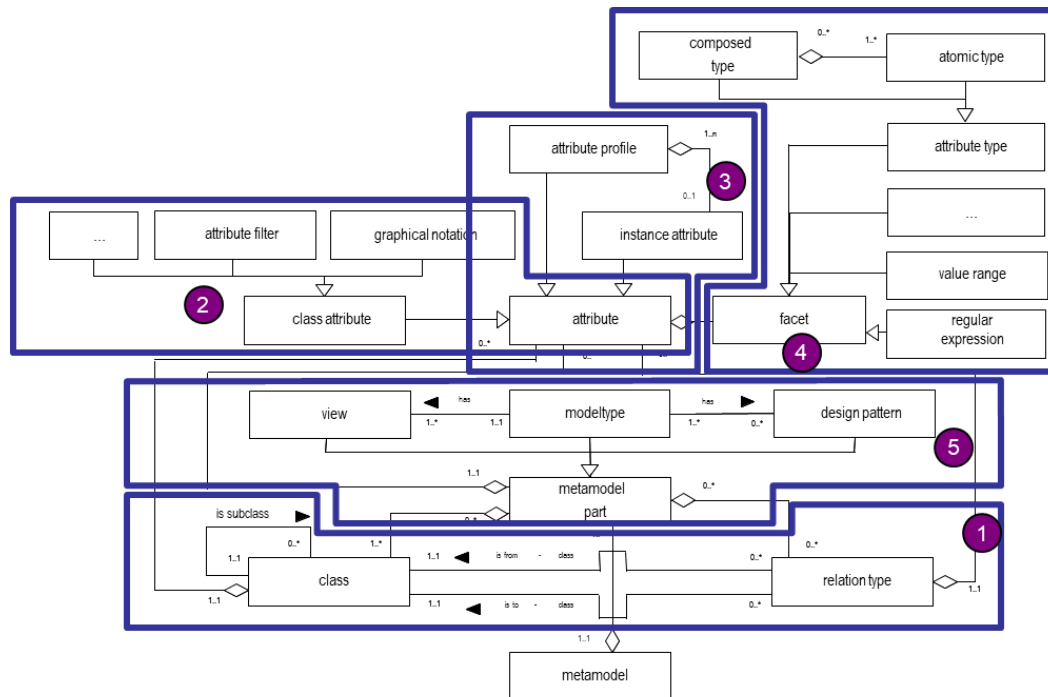
► Mechanisms & algorithms

- ▶ provide the functionality to use and evaluate the models built by using the modelling language.
 - ▶ Mechanisms can be classified into generic, specific, and hybrid
 - Generic Mechanisms & algorithms
 - Meta2-model
 - ▶ Model of abstract syntax of a language to describe meta models.
 - Specific Mechanisms & algorithms
 - Particular meta model
 - Hybrid Mechanisms & algorithms
 - Generic + Specific



ADOxx

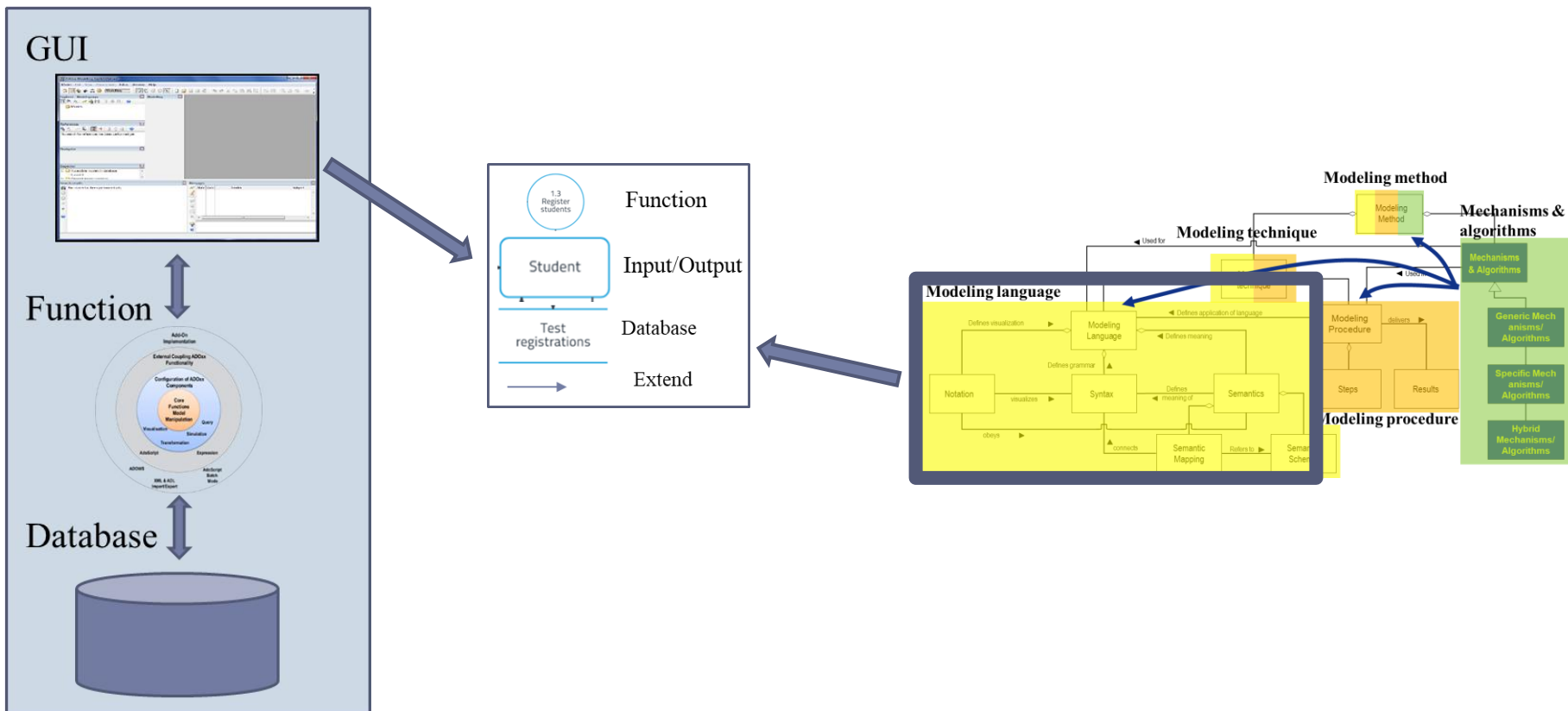
- ▶ Mechanisms & algorithms
 - ▶ Meta2 model
 - ▶ ADOxx functionality on meta level



ADOxx

- ▶ Modeling language
 - ▶ Modeling constructs and their relations to each other to declare a model

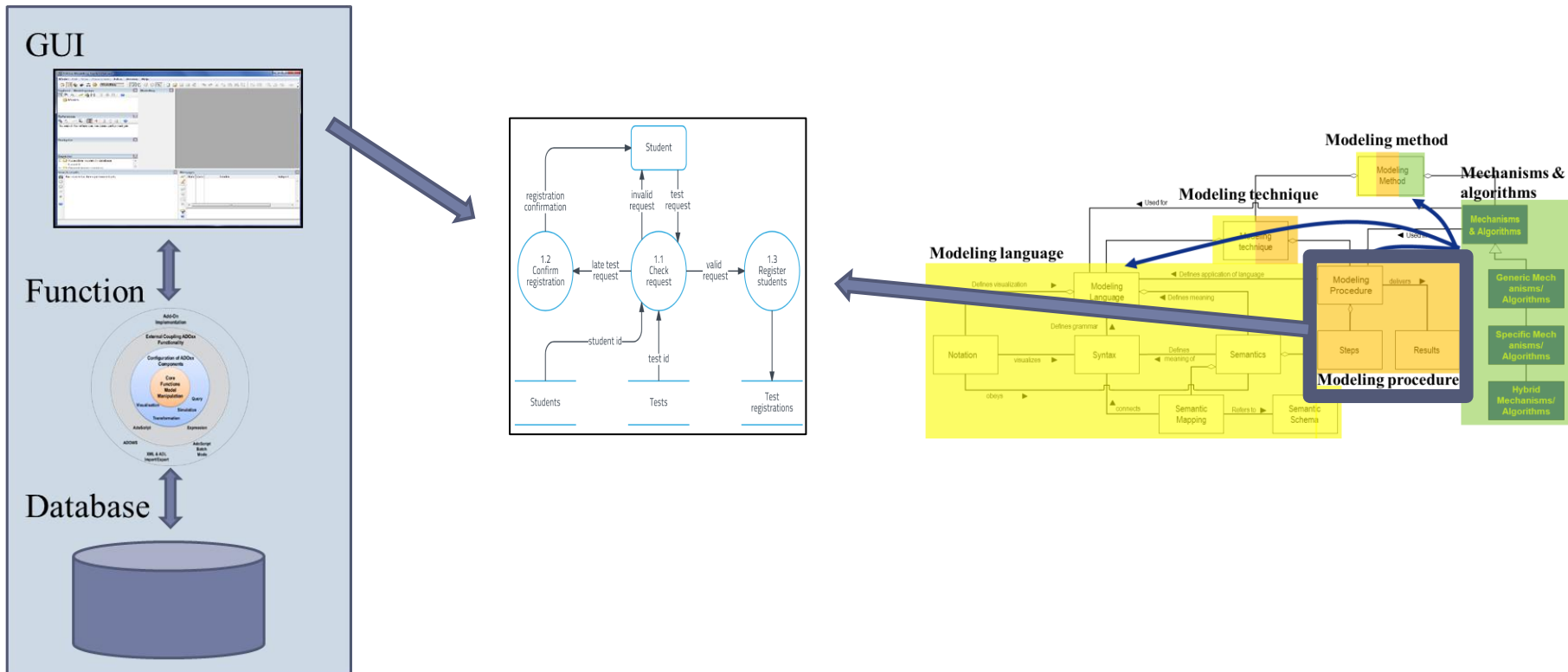
Tool



ADOxx

- ▶ Modeling procedure
 - ▶ Describe the steps applying the modelling language to create results
 - ▶ i.e. models

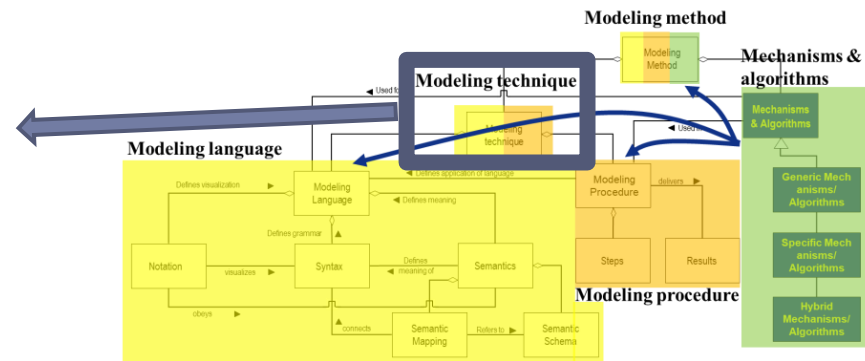
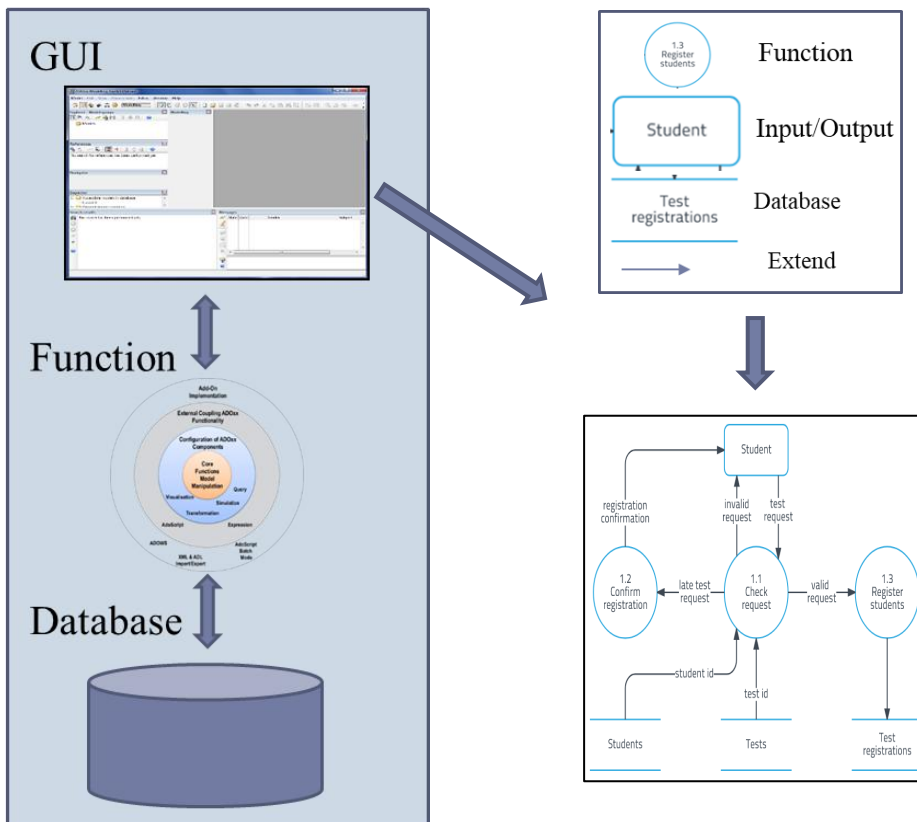
Tool



ADOxx

- Modeling technique
 - Modeling language + Modeling Procedure

Tool

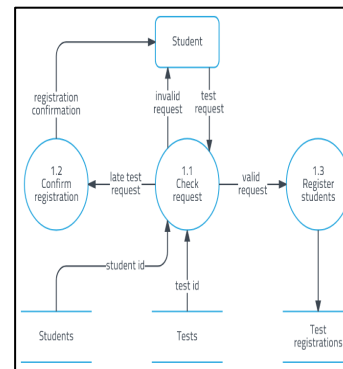
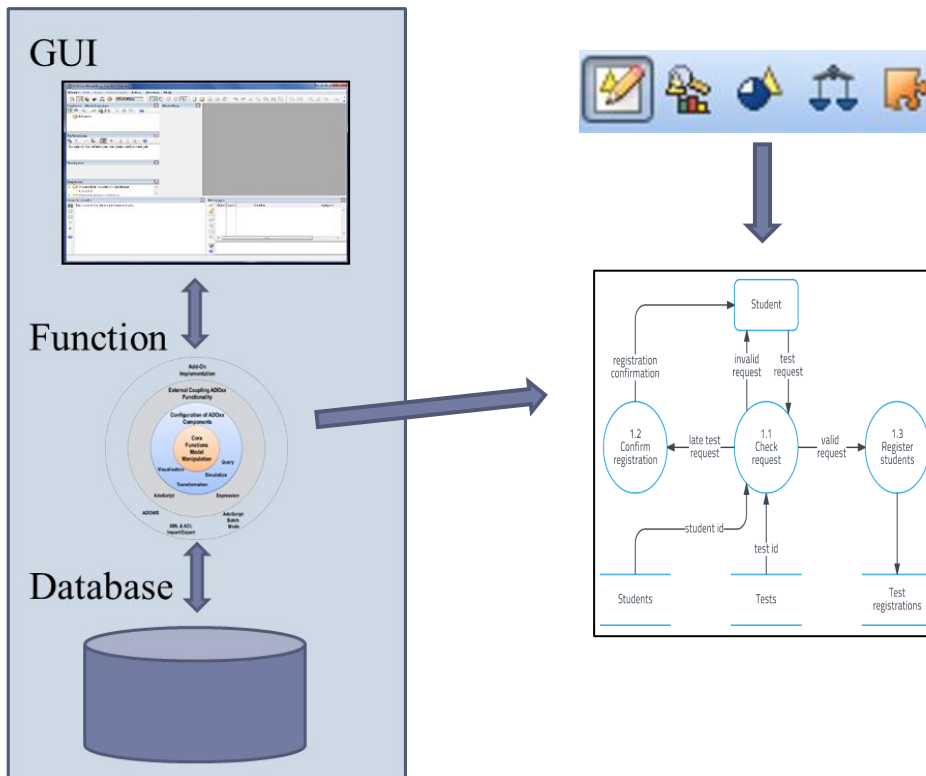


ADOxx

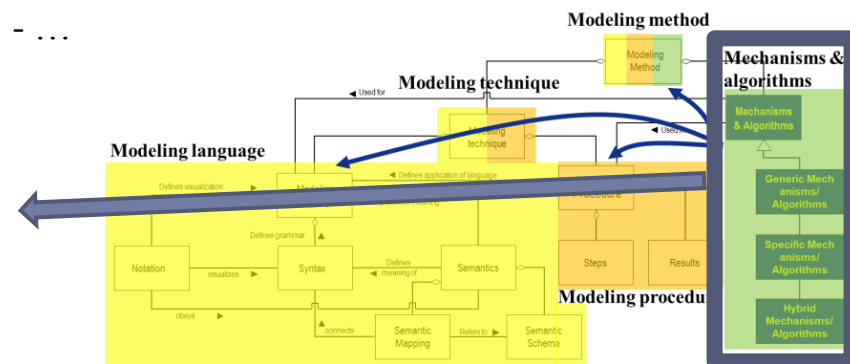
► Mechanisms & algorithms

- Provide the functionality to use and evaluate the models built by using the modelling language

Tool



- Modeling
- Analysis
- Simulation
- Additional Function
- ...



1. Overview

- 1) Meta Model
- 2) ADOxx

2. Details

- 1) ADOxx Toolkit
- 2) Modeling Language
- 3) Core Functions
- 4) Adoscript
- 5) Simulation

2. DETAILS

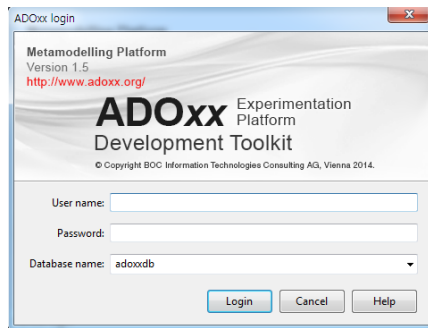
1. Overview
 - 1) Meta Model
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit**
 - 2) Modeling Language
 - 3) Core Functions
 - 4) Adoscript
 - 5) Simulation

1) ADOXX TOOLKIT

ADOxx

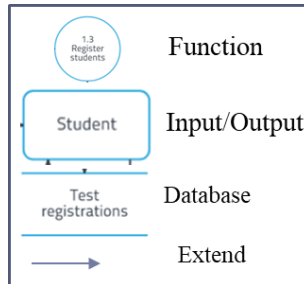
- ▶ ADOxx Toolkits
 - ▶ Development Toolkit
 - ▶ Modeling Toolkit

BEE-UP

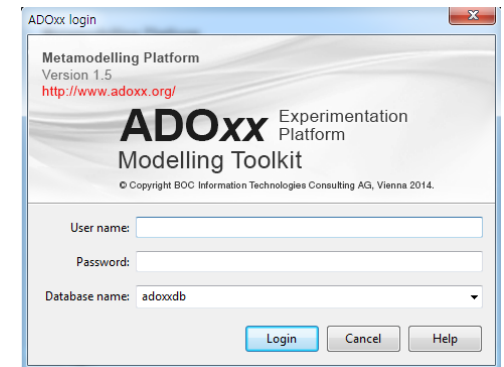
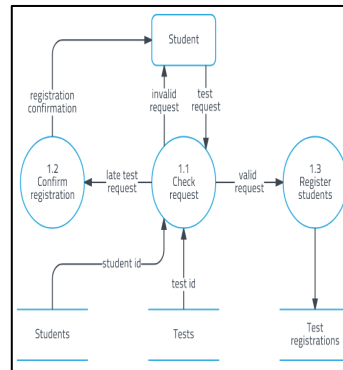


Development Toolkit

Define libraries



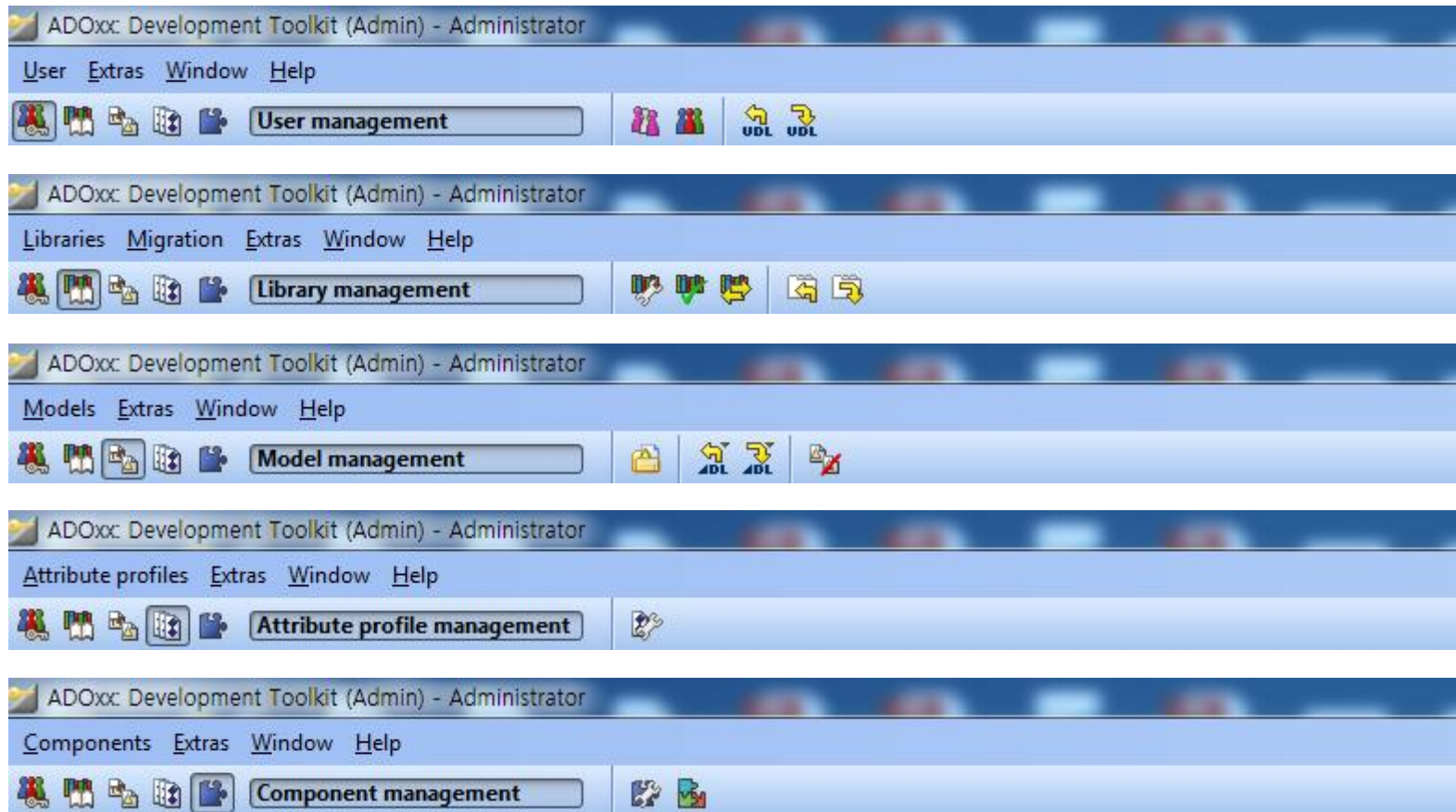
Use libraries



Modeling Toolkit

ADOxx Development Toolkit

► Components



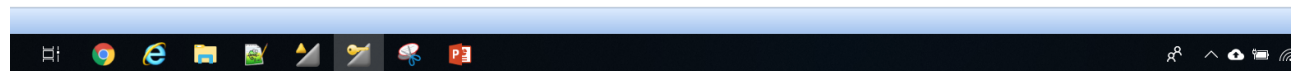
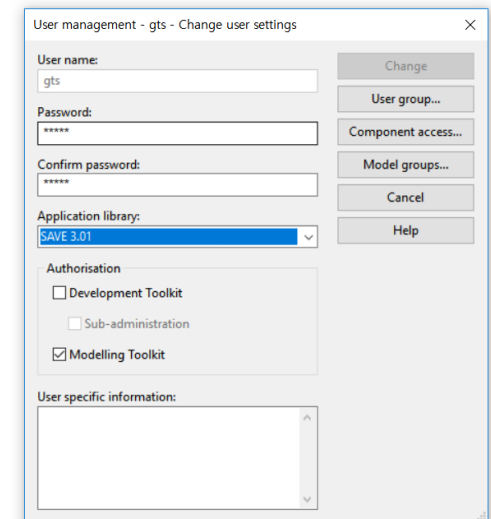
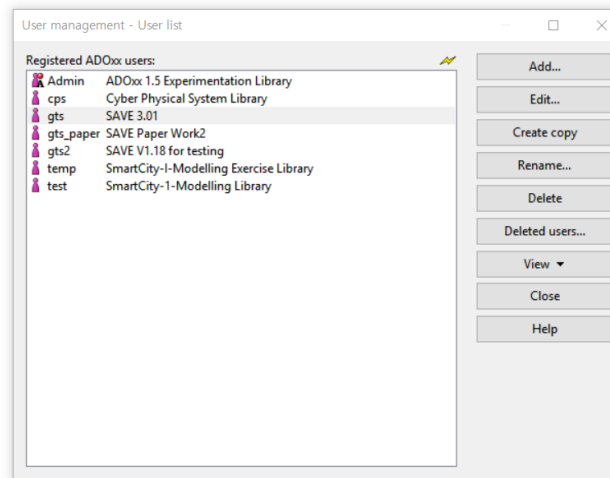
ADOxx Development Toolkit

► User management

► Create and edit user

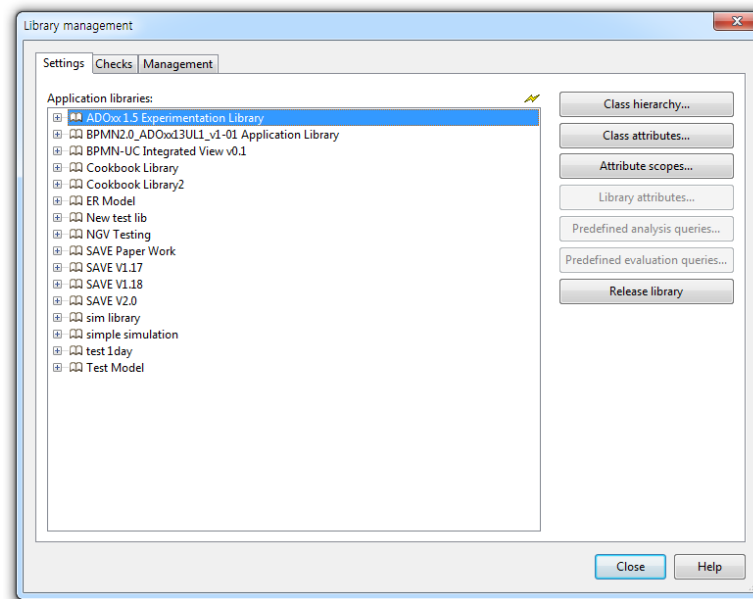
- ID
- Password
- Library
- Right
- ...

velopment Toolkit (Admin) - Administrator



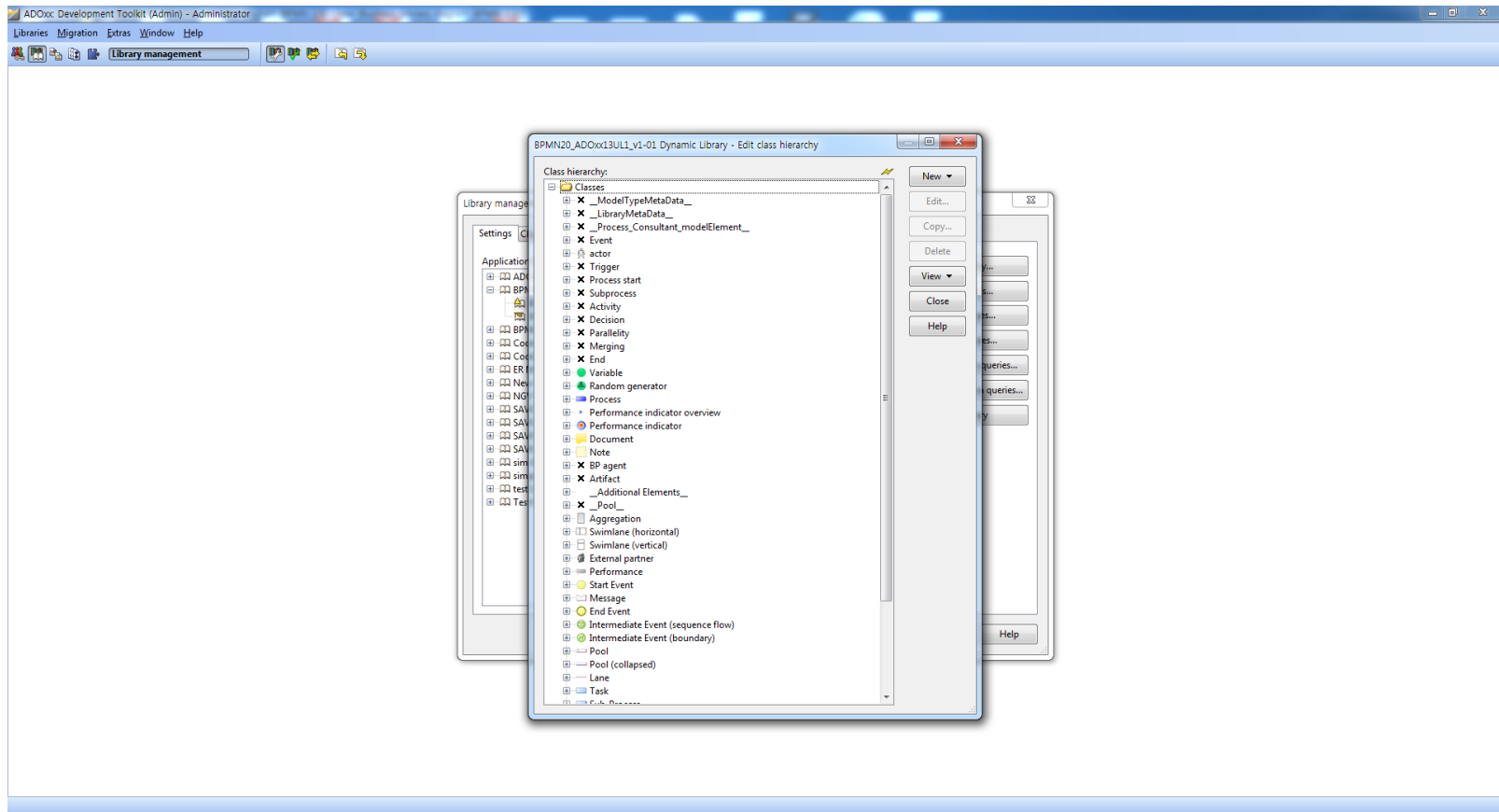
ADOxx Development Toolkit

- ▶ Library management
 - ▶ Manage tools (library)



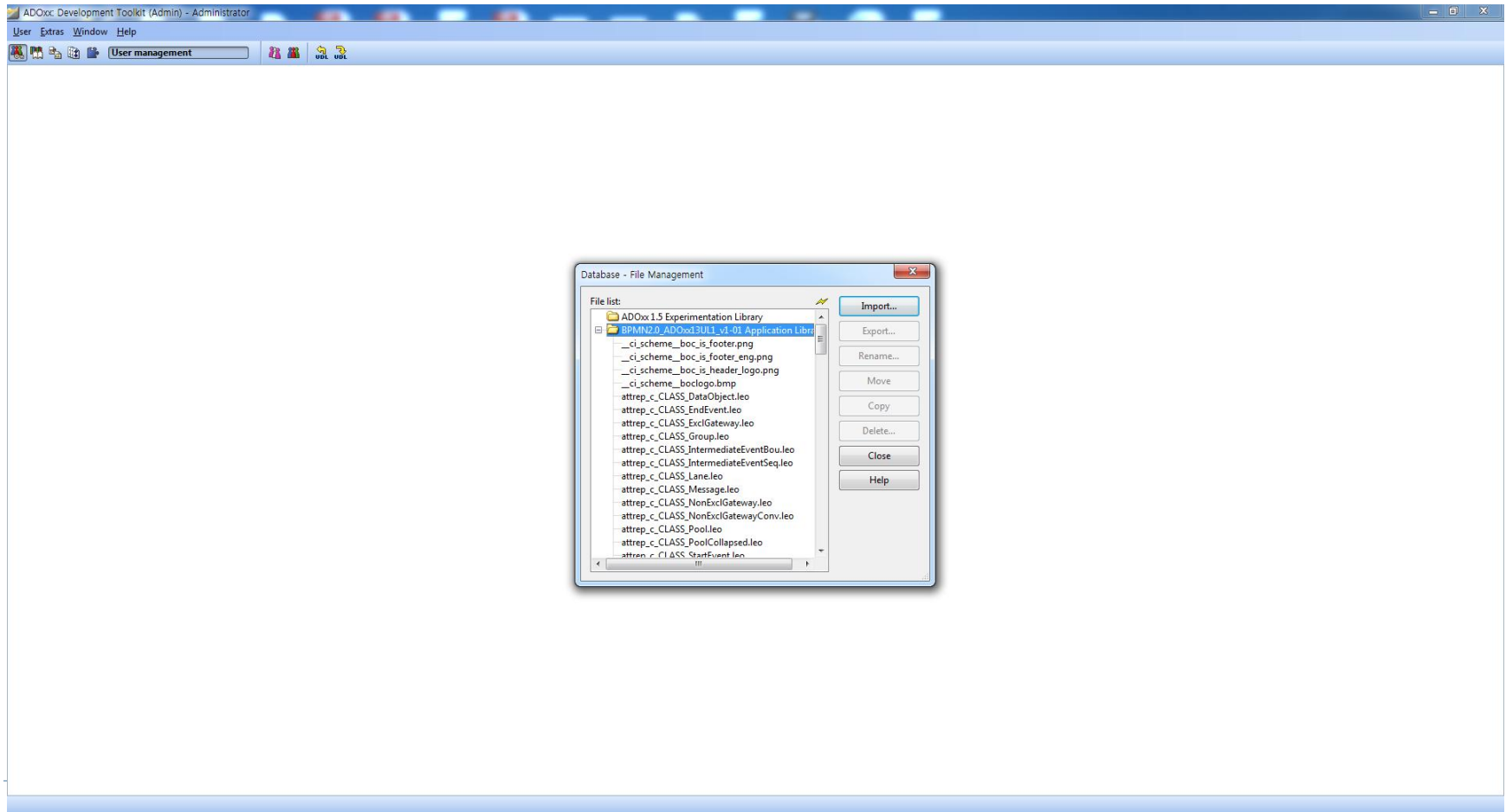
ADOxx Development Toolkit

- ▶ Library management
 - ▶ Edit classes



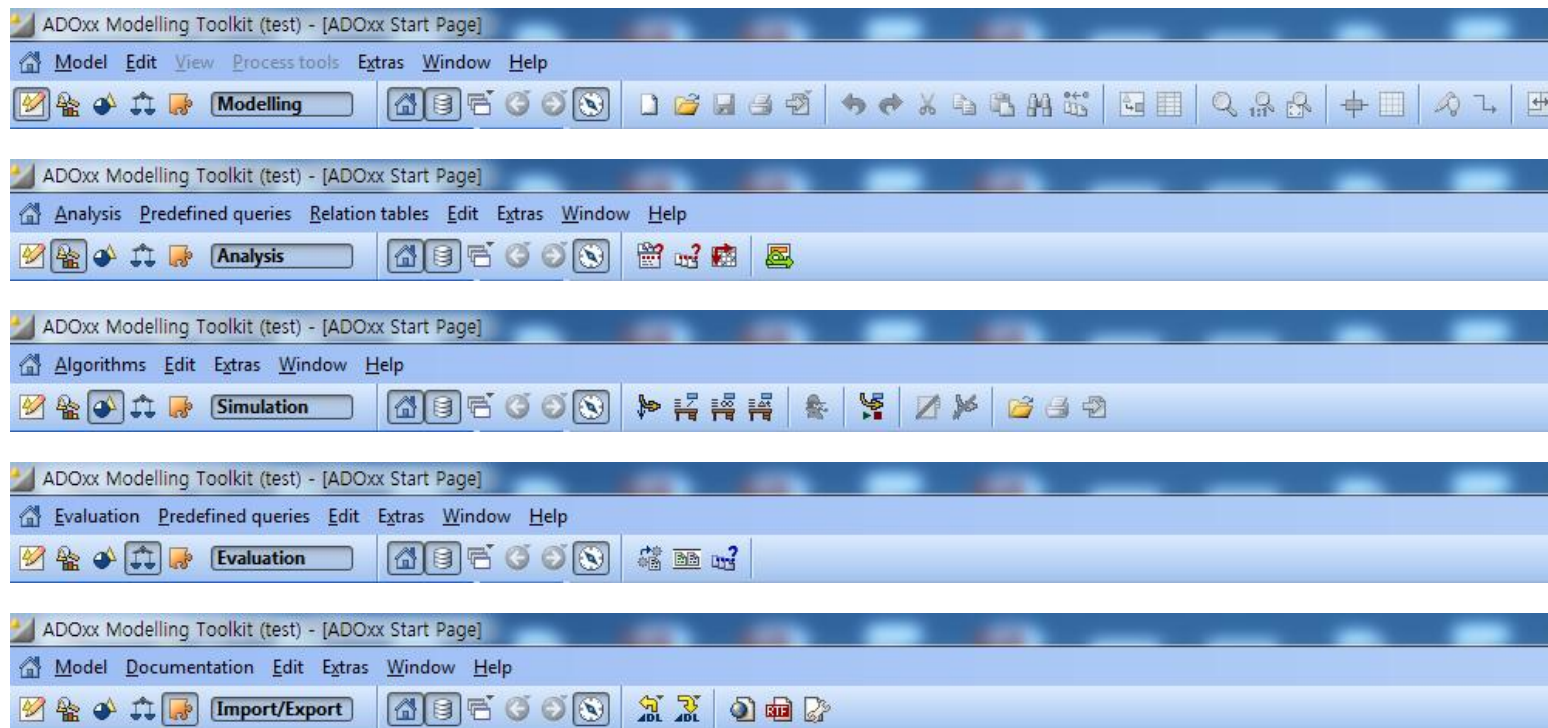
ADOxx Development Toolkit

- ▶ Library management
 - ▶ File management



ADOxx Modeling Toolkit

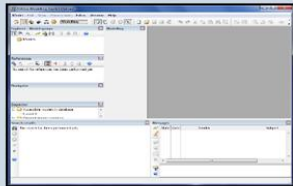
► Components



ADOxx Modeling Toolkit

Tool

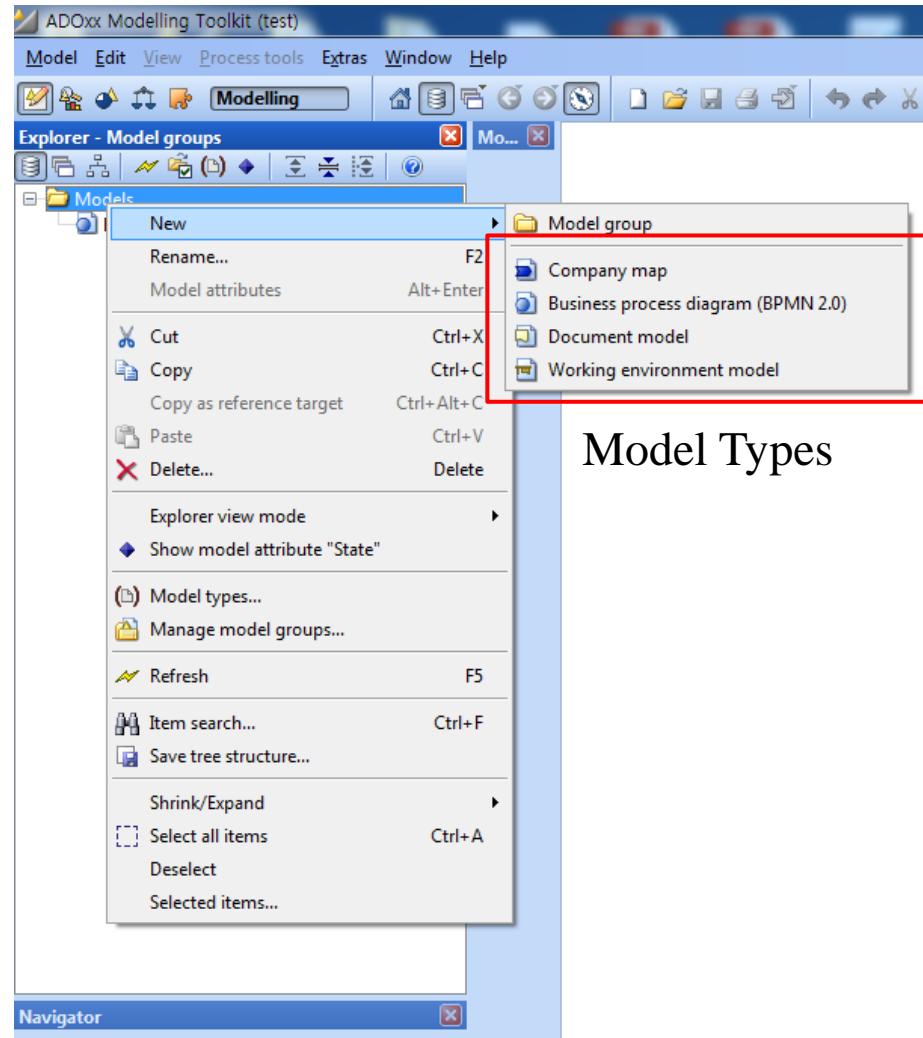
GUI



Function



Database



Model Types

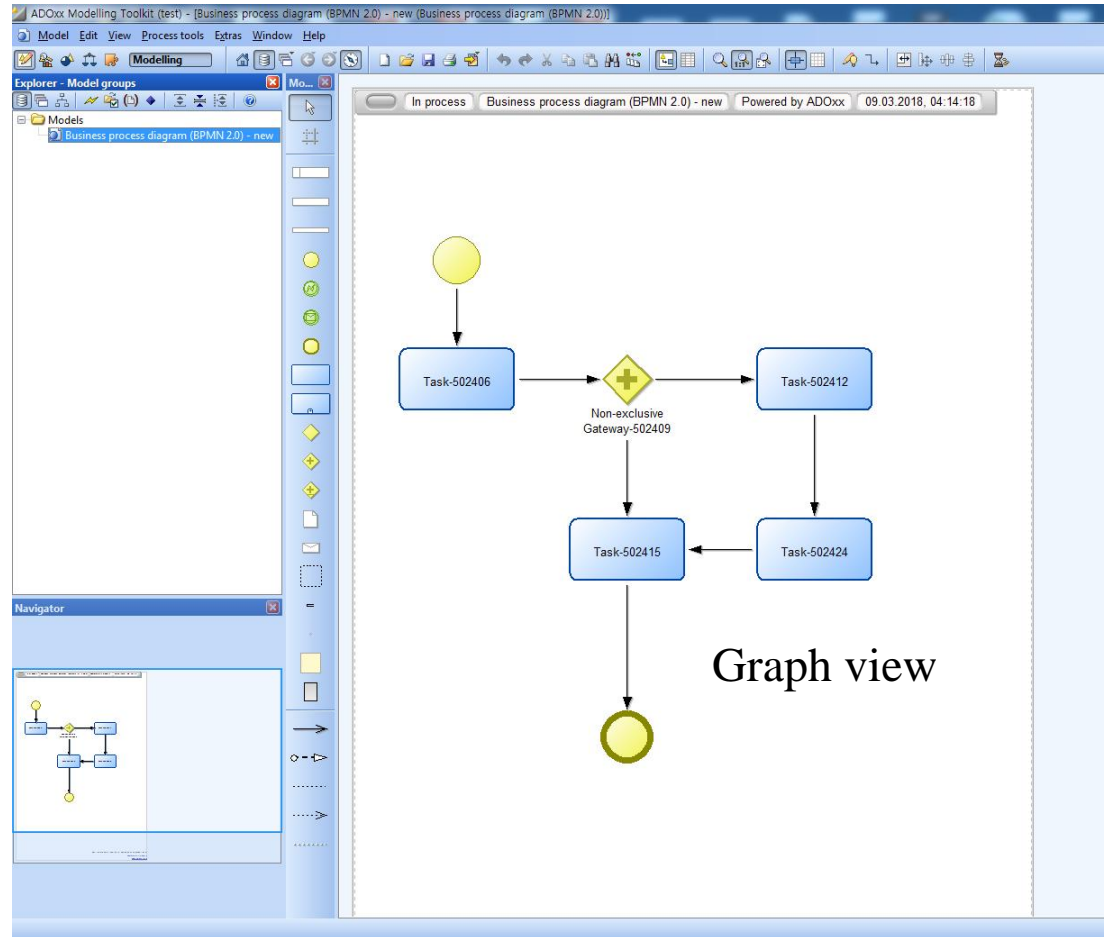
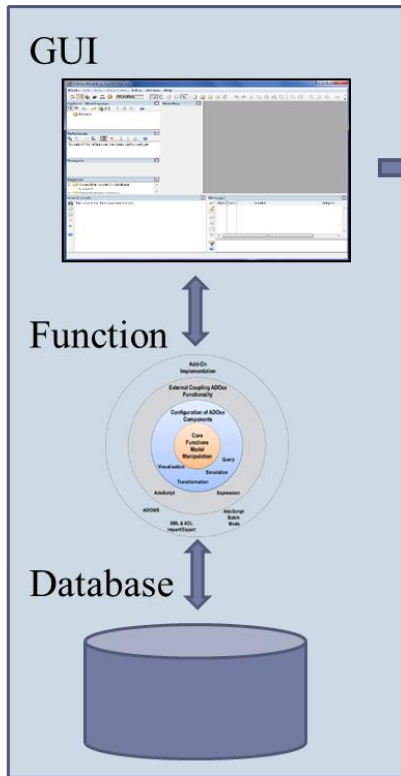
ADOxx Modeling Toolkit

Tool

GUI

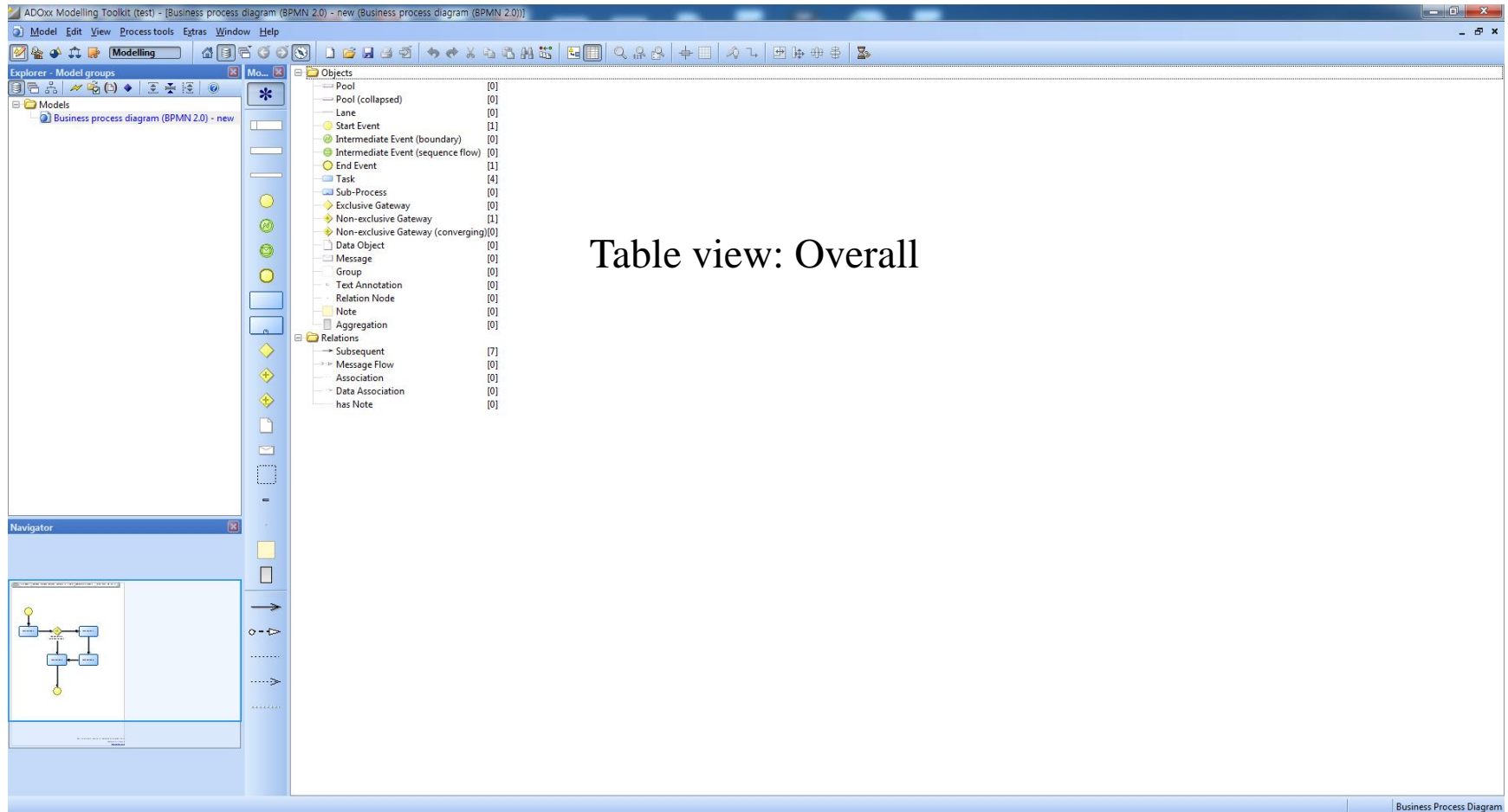
Function

Database



Graph view

ADOxx Modeling Toolkit



ADOxx Modeling Toolkit

The screenshot displays the ADOxx Modeling Toolkit interface. The main window shows a table view for one class, with the text "Table view for one class" overlaid in the center. The table has the following columns: Name, Show name, Task type, Global task, Order, Description, Comment, Open questions, Id (-), Auditing, Monitoring, For compensati..., Loop type, Loop condition ..., and Sequential exec. The table contains four rows of data, each representing a task.

Name	Show name	Task type	Global task	Order	Description	Comment	Open questions	Id (-)	Auditing	Monitoring	For compensati...	Loop type	Loop condition ...	Sequential exec
Task-502406	Task-502406	center	Not specified	No	0			502406	No	No	No	Not specified		No
Task-502412	Task-502412	center	Not specified	No	0			502412	No	No	No	Not specified		No
Task-502415	Task-502415	center	Not specified	No	0			502415	No	No	No	Not specified		No
Task-502424	Task-502424	center	Not specified	No	0			502424	No	No	No	Not specified		No

The interface also includes a left sidebar with a "Navigator" pane showing a small diagram of the business process, and a top menu bar with options like Model, Edit, View, Process tools, Extras, Window, and Help.

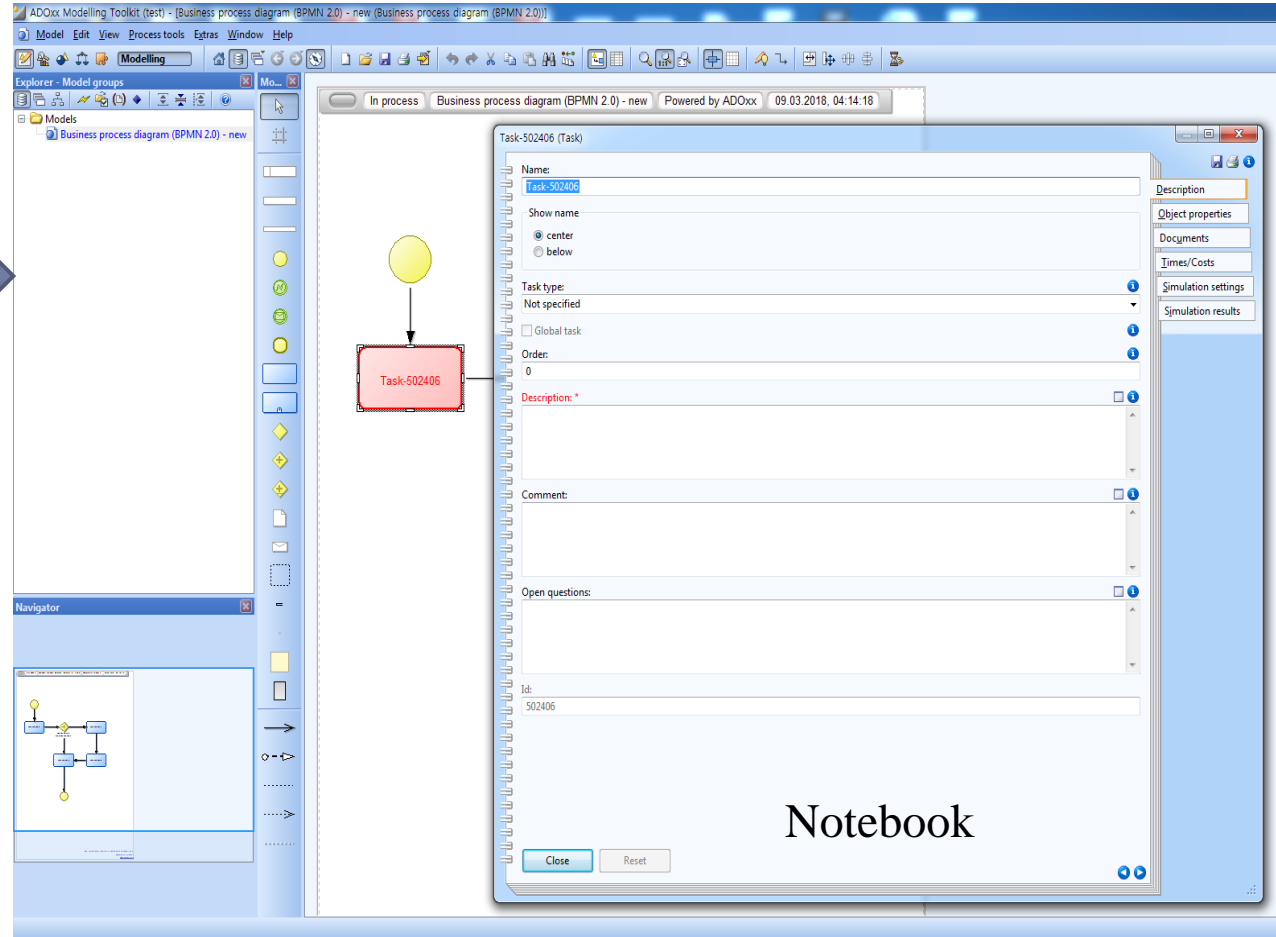
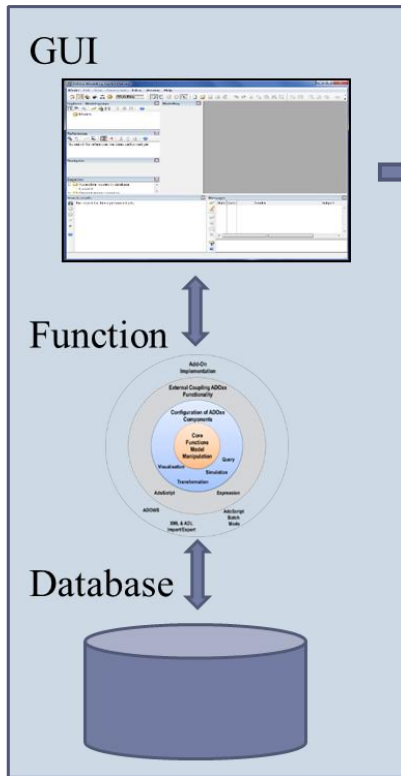
ADOxx Modeling Toolkit

Tool

GUI

Function

Database



Notebook

ADOxx Modeling Toolkit

Task-502406 (Task)

Task relevance

☐ Auditing

☐ Monitoring

For compensation

☐ For compensation

Loop details

Loop type

☒ Not specified

☐ Standard

☐ Multi-instance

Loop condition (standard):

Multi-instance loop details

☐ Sequential execution

Cardinality:

Referenced data input: + X i

Referenced data output: + X i

Completion condition: i

Send/Receive task details

Referenced message: + X i

Receive task details

☐ Instantiate i

Chapter

Close Reset 1/2

ADOxx Modeling Toolkit

Task-502406 (Task)

Task relevance

☐ Auditing i

☐ Monitoring i

☒ For compensation i

Loop details

Loop type i

☒ Not specified

☐ Standard

☐ Multi-instance

Loop condition (standard): i

Multi-instance loop details

☐ Sequential execution i

Cardinality: i

Referenced data input: + X i

Referenced data output: + X i

Completion condition: i

Send/Receive task details

Referenced message: + X i

Receive task details

☐ Instantiate i

Group

Close Reset

1/2

Description

Object properties

Documents

Times/Costs

Simulation settings

Simulation results

ADOxx Modeling Toolkit

Task-502406 (Task)

Task relevance

☐ Auditing

☐ Monitoring

For compensation

☐ For compensation

Loop details

Loop type

☒ Not specified

☐ Standard

☐ Multi-instance

Loop condition (standard):

Multi-instance loop details

☐ Sequential execution

Cardinality:

Referenced data input:

Referenced data output:

Completion condition:

Send/Receive task details

Referenced message:

Receive task details

☐ Instantiate

Close Reset

1/2

Attribute

Description

Object properties

Documents

Times/Costs

Simulation settings

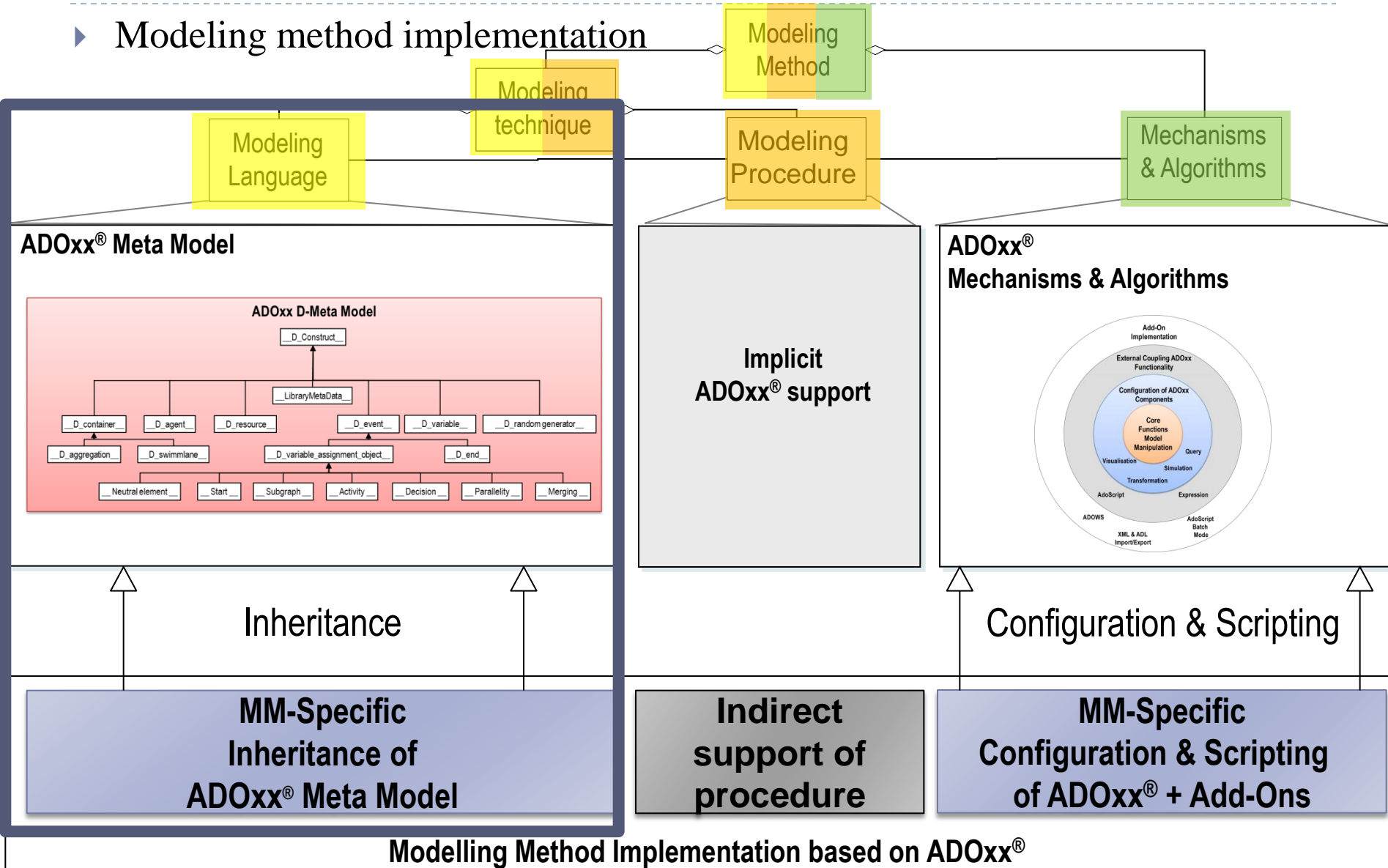
Simulation results

1. Overview
 - 1) Meta Model
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language**
 - 3) Core Functions
 - 4) Adoscript
 - 5) Simulation

2) MODELING LANGUAGE

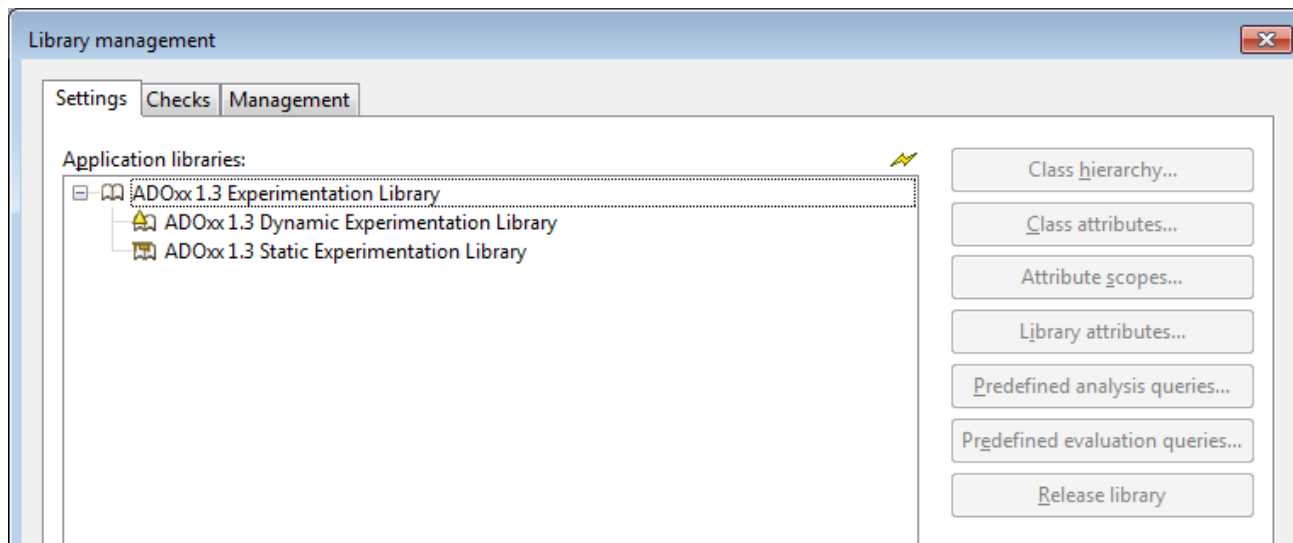
Modeling Language

► Modeling method implementation



Modeling Language

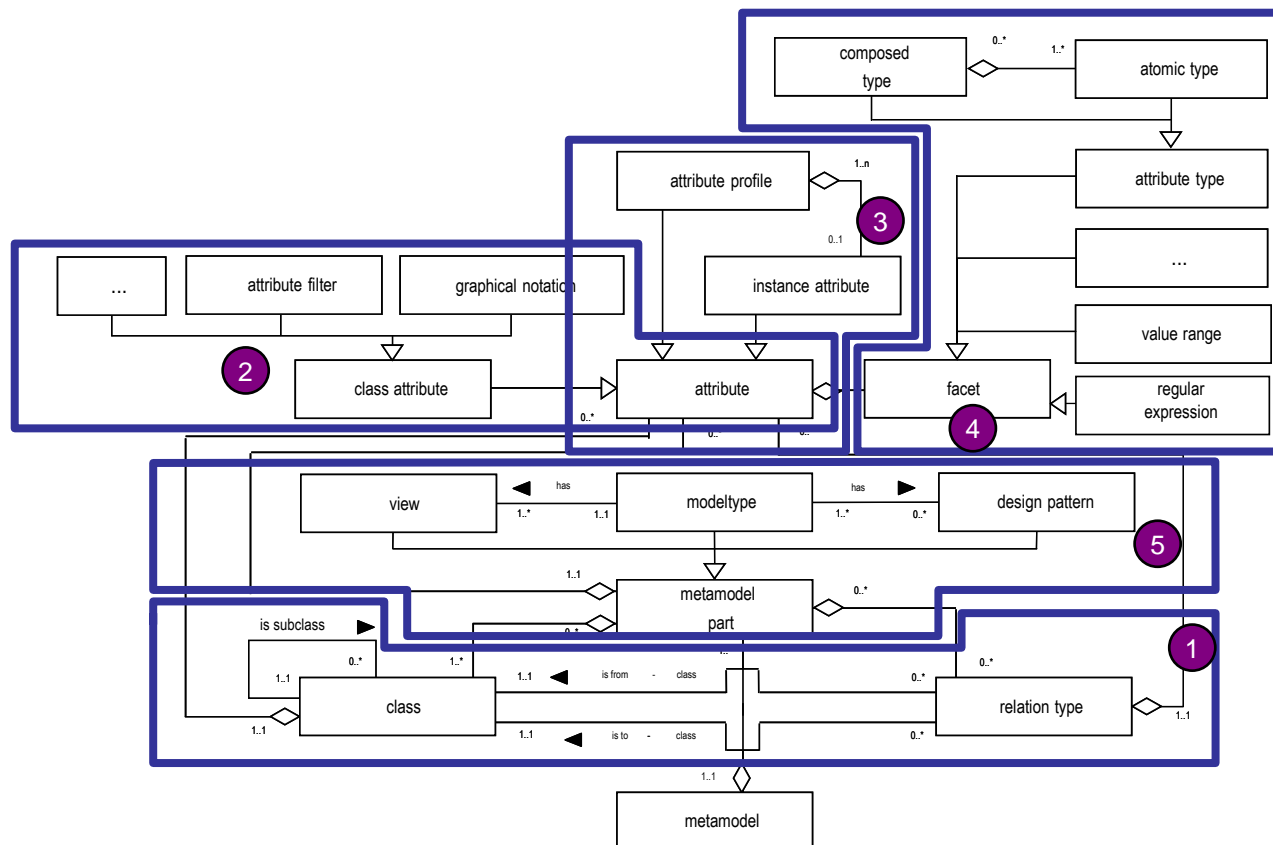
- ▶ Library
 - ▶ Dynamic : Graph-based environment
 - ▶ Static : Tree-based environment



Modeling Language

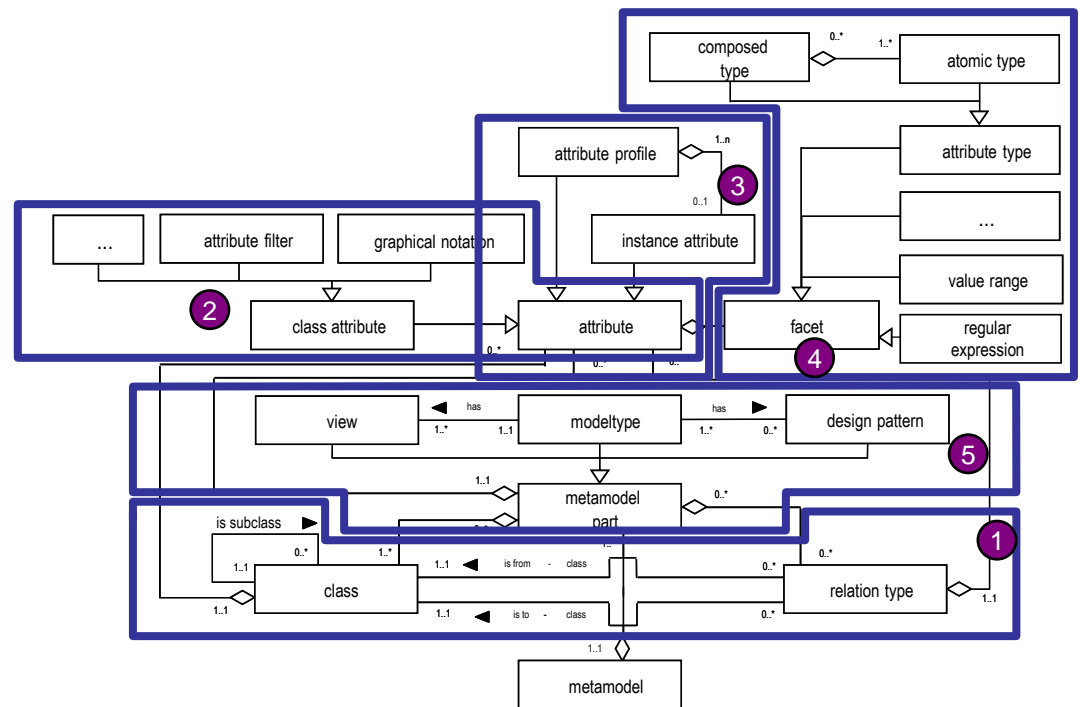
► Meta2 Model

- Model of abstract syntax of a language to describe meta models.



Modeling Language

- ① Classes & Relations
- ② Class Attribute & Attribute
- ③ Special Class Attribute & Attribute
- ④ Attribute Facets
- ⑤ Model Types

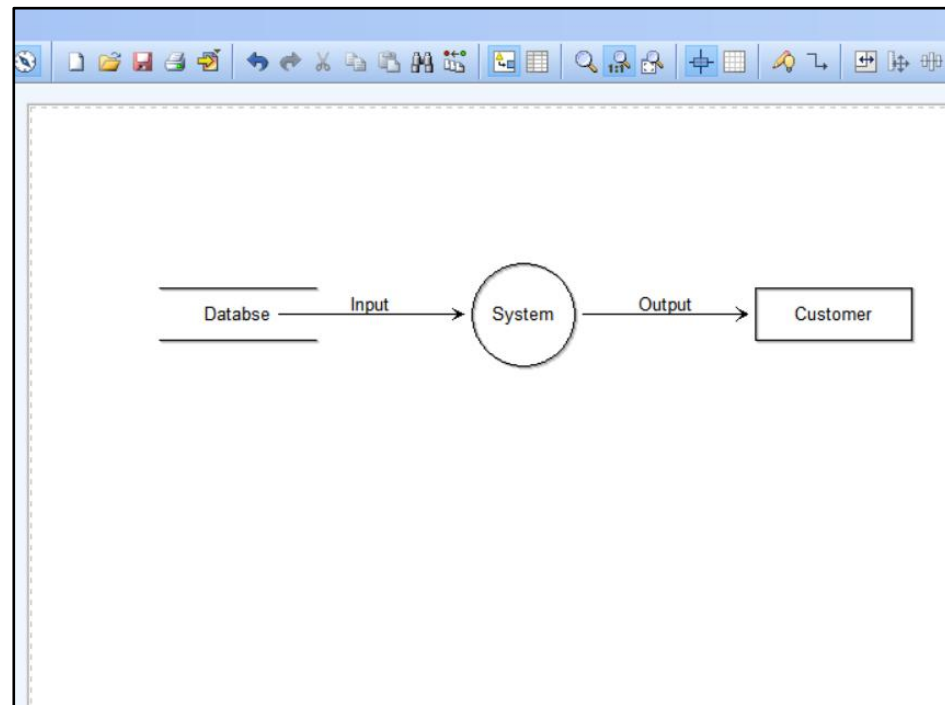
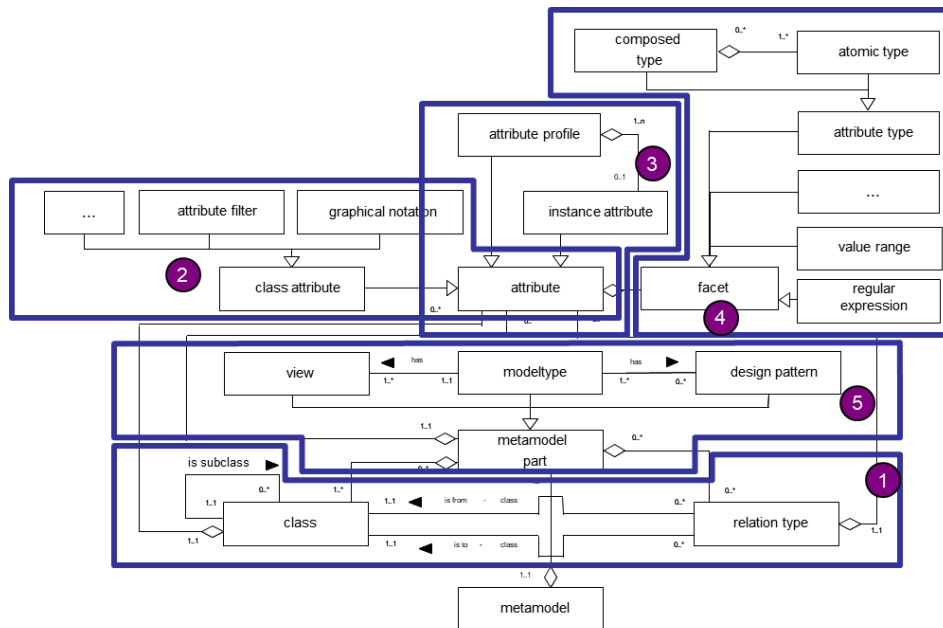


Modeling Language

► Demo

► Data Flow Diagram tool

► How to make and use Data Flow Diagram tool using Meta2 Model



① Classes & Relations



Modeling Language

- ▶ Class
 - ▶ Pre-defined Abstract Class
 - ▶ Provide by ADOxx
 - ▶ Nomenclature: __Class Name__
 - ▶ Abstract Class
 - ▶ Self-defined abstract class
 - ▶ Nomenclature: _Class Name_
 - ▶ Class
 - ▶ Self-defined concrete class
 - ▶ Nomenclature: Class Name

Modeling Language

► Class

The screenshot displays two windows from the ADOxx 1.3 Dynamic Tutorial Library (Experimentation Environment).

Library management window: This window shows a list of application libraries under the 'Management' tab. The libraries listed are:

- ADOxx 1.3 Tutorial Library (Experimentation Environment)
- ADOxx 1.3 Dynamic Tutorial Library (Experimentation Environment) (selected)
- ADOxx 1.3 Static Tutorial Library (Experimentation Environment)

Buttons on the right side of this window include: Class hierarchy..., Class attributes..., Attribute scopes..., Library attributes..., Predefined analysis queries..., Predefined evaluation queries..., and Release library. At the bottom are 'Close' and 'Help' buttons.

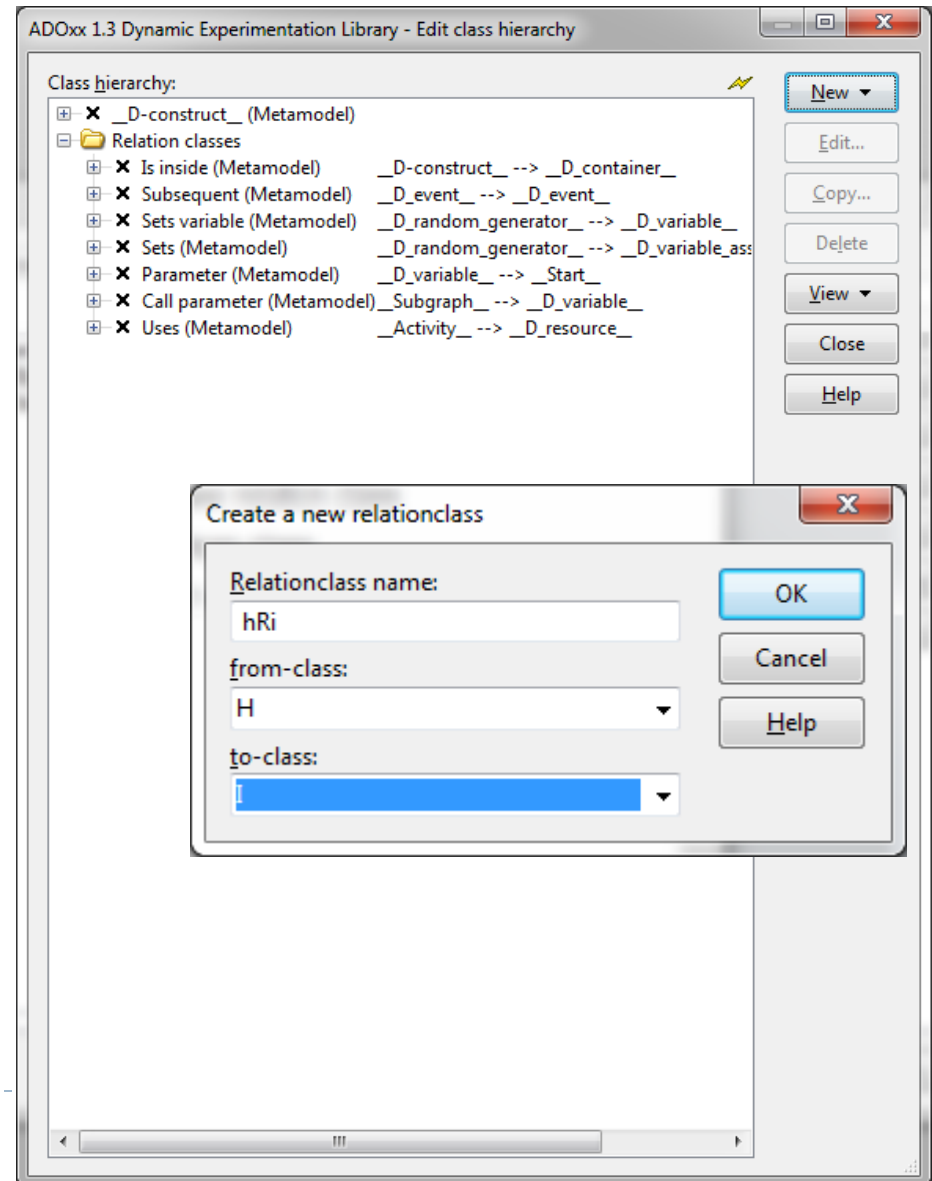
Edit class hierarchy window: This window shows a class hierarchy tree. The root node is 'D-construct_ (Metamodel)'. It contains several sub-nodes, including:

- _D_event_ (Metamodel)
- _D_variable_ (Metamodel)
- _D_random_generator_ (Metamodel)
- _D_resource_ (Metamodel)
- _D_container_ (Metamodel)
- _D_agent_ (Metamodel)
- _LibraryMetaData_
- A
- B
- W
- AnimRep (Metamodel)
- AttrRep (Metamodel)
- Class cardinality (Metamodel)
- ClassAbstract
- ClassName
- ClassVisible
- External tool coupling (Metamodel)
- GraphRep (Metamodel)
- HlpTxt (Metamodel)
- Model pointer (Metamodel)
- Position (Metamodel)
- VisibleAttrs (Metamodel)
- WF_Trans (Metamodel)
- Relation classes
- Is inside (Metamodel)
- Subsequent (Metamodel)
- Sets variable (Metamodel)
- Sets (Metamodel)
- Parameter (Metamodel)
- Call parameter (Metamodel)
- Uses (Metamodel)
- aRb

Each node has associated attributes listed on the right side of the window. For example, 'String (STRING)', 'Longstring (LONGSTRING)', 'Integer (INTEGER)', and 'String (STRING)' are listed for various nodes. The 'Relation classes' section shows relationships like '_D-construct_ --> _D_container_', '_D_event_ --> _D_event_', etc.

Modeling Language

► Relation Class



Modeling Language

▶ Demo (Classes & Relations)

▶ Class

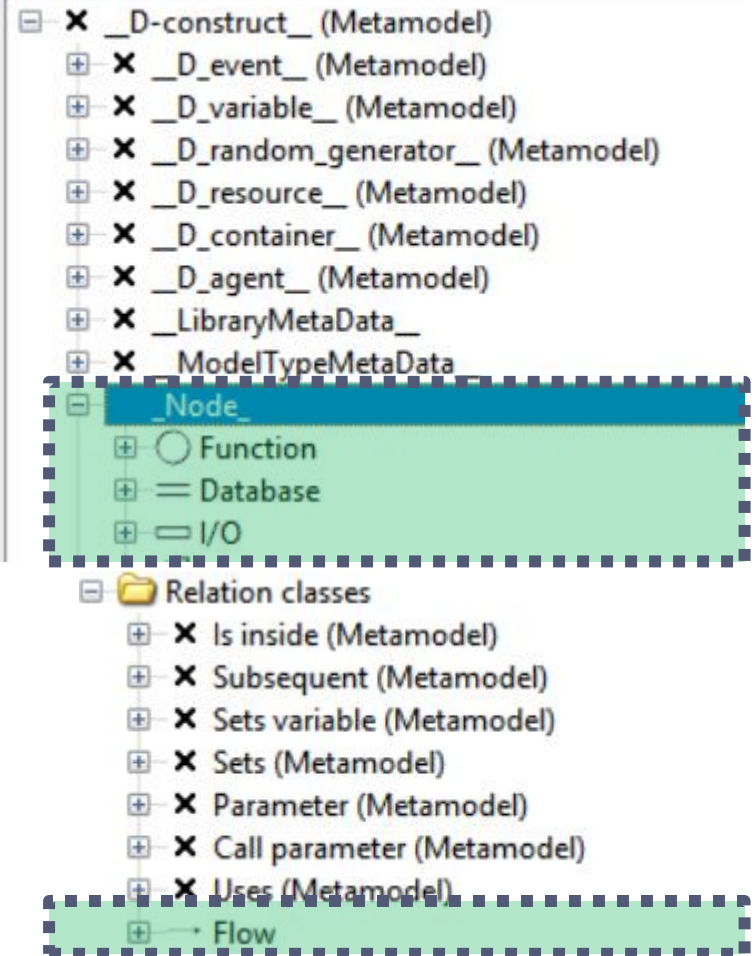
▶ _Node_

- Function
- I/O
- Database

▶ Relation Class

▶ Flow

Class hierarchy:



⑤ Model Types



Modeling Language

- ▶ Model type

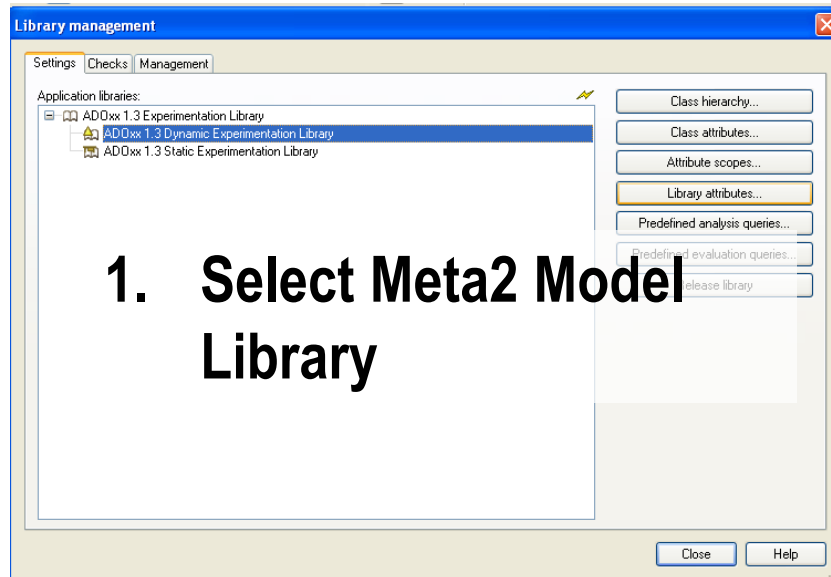
- ▶ Subset of all instanciable classes and relations

- ▶ example

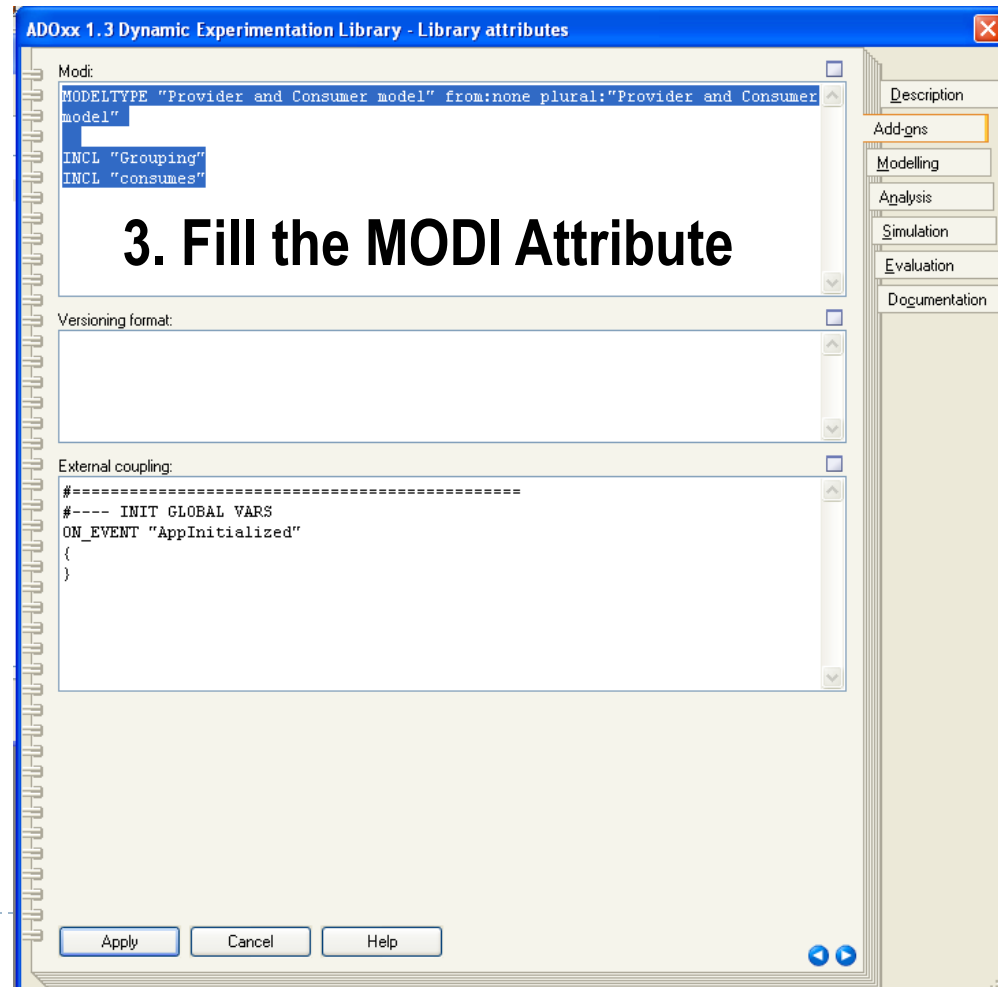
```
MODELTYPE "My First Model Type"  
  from:none  
  plural:"My First Model Types"  
  pos:1  
  not-simulateable  
  bitmap:"db:\\MyFirstModelType.bmp "  
  attrrep:"Notebook for My First Model Type"  
INCL "My Class 1"  
INCL "My Class 2"  
INCL "My Class 3"  
INCL "has relationship 1"  
INCL "has relationship 2"
```

Modeling Language

► Model Types



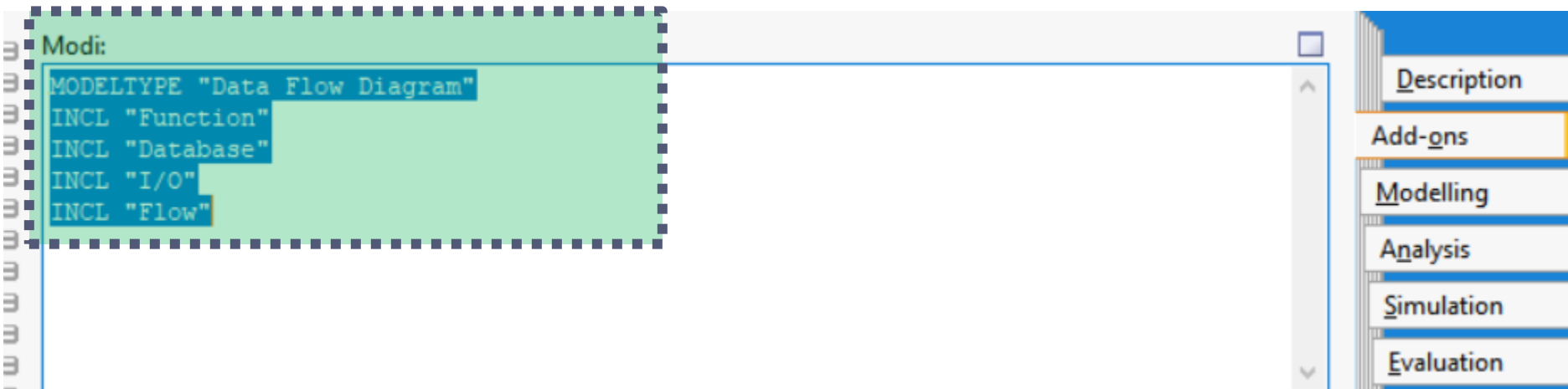
2. Select the Tab Add-Ons



Modeling Language

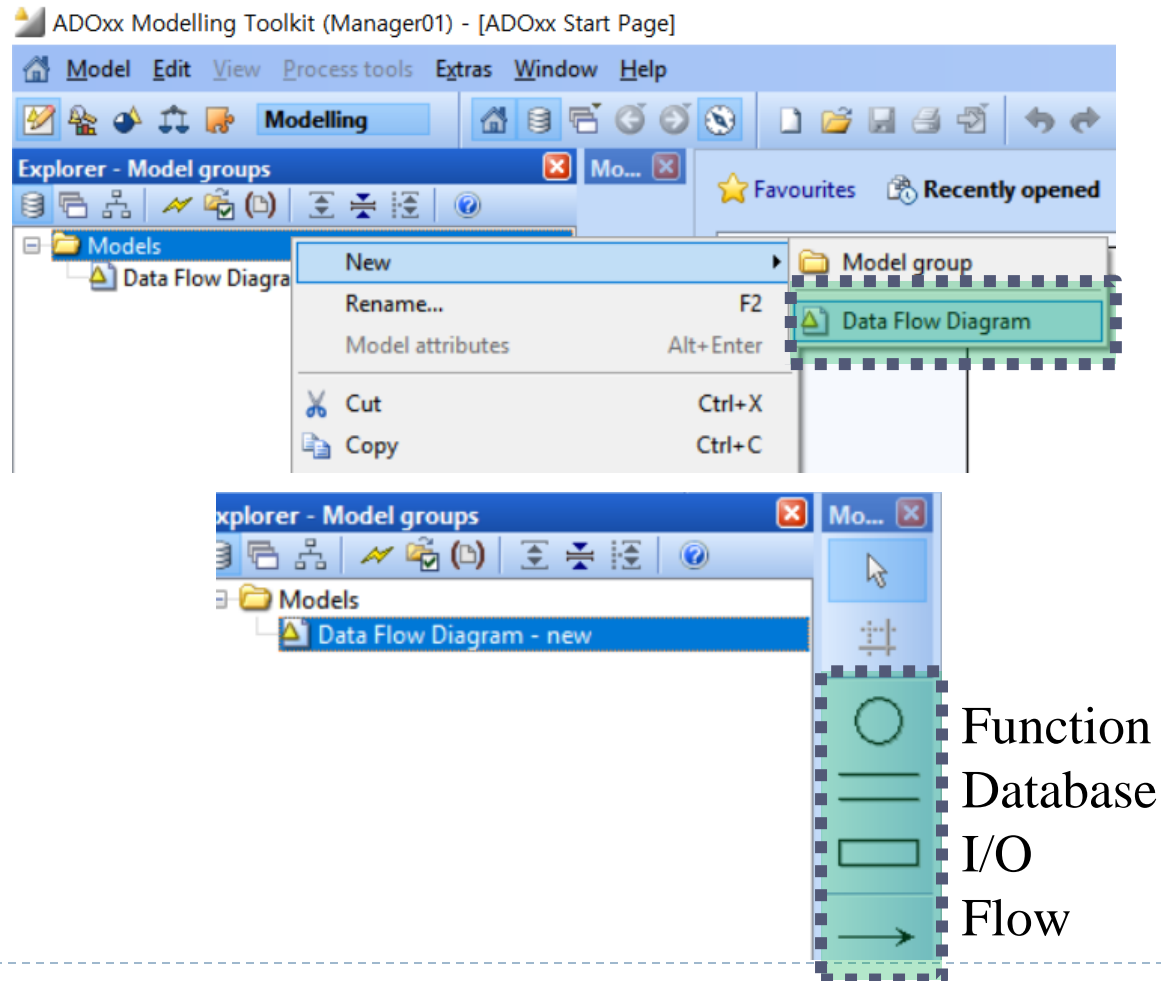
► Demo (Model Types)

```
MODELTYPE "Data Flow Diagram"  
INCL "Function"  
INCL "Database"  
INCL "I/O"  
INCL "Flow"
```



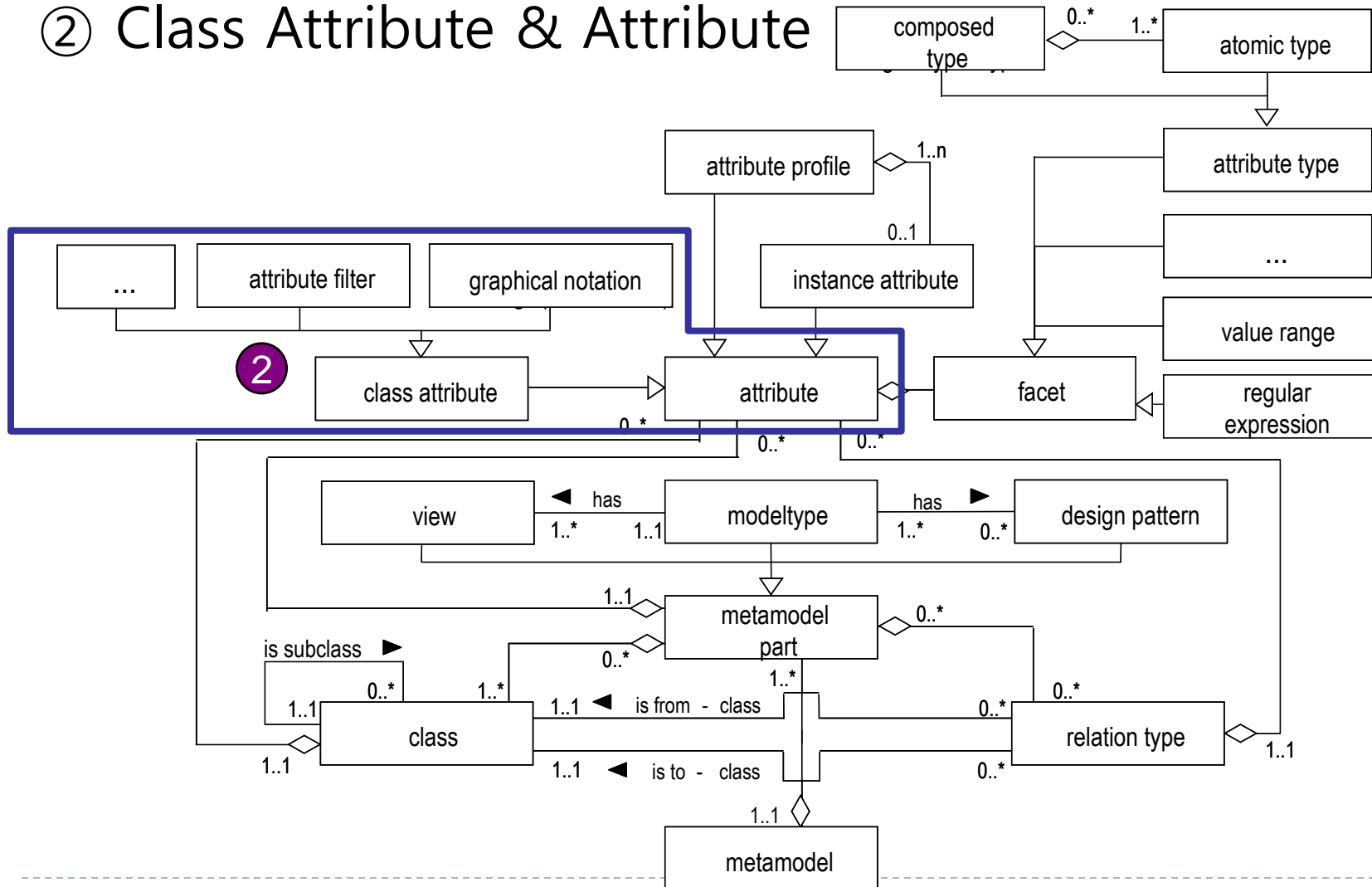
Modeling Language

► Demo (Model Types)



Modeling Language

② Class Attribute & Attribute



Modeling Language

▶ Attributes

▶ Type

- ▶ INTEGER integer
- ▶ DOUBLE floating number
- ▶ STRING string – max. 3699 symbols
- ▶ LONGSTRING string – max. 32000 symbols
- ▶ TIME time
- ▶ DATE date
- ▶ DATETIME date and time
- ▶ ENUMERATION enumeration for selecting a characteristic
- ▶ ENUMERATIONLIST enumeration for selecting one or several characteristics
- ▶ PROGRAMCALL enumeration for selecting a program
- ▶ RECORD a table of attributes
- ▶ EXPRESSION a formula
- ▶ INTERREF reference on a model or an instance
- ▶ ATTRPROFREF a preset set of attribute values

Modeling Language

The screenshot displays a software interface for modeling. On the left, a tree view titled 'My First Class' lists various metamodel elements and their associated data types. A context menu is open over this tree, showing options like 'New class...', 'New attribute...', and 'New class attribute...'. An 'Add new attribute' dialog is also open, allowing the user to define a new attribute with a name, type, and source.

Metamodel Element	Data Type
AnimRep (Metamodel)	String (STRING)
AttrRep (Metamodel)	Longstring (LONGSTRING)
Class cardinality (Metamodel)	String (STRING)
ClassAbstract	Integer (INTEGER)
ClassName	String (STRING)
ClassVisible	Integer (INTEGER)
External tool coupling (Metamodel)	String (STRING)
GraphRep (Metamodel)	Longstring (LONGSTRING)
HlpTxt (Metamodel)	String (STRING)
Model pointer (Metamodel)	String (STRING)
Position (Metamodel)	String (STRING)
Selection (Metamodel)	Expression (EXPRESSION)
VisibleAttrs (Metamodel)	String (STRING)
WF_Trans (Metamodel)	String (STRING)

Context Menu Options:

- New class...
- New attribute...
- New class attribute...
- Copy...
- Delete
- Classes
- Relationclasses
- Metamodel
- Class hierarchy
- Attribute
- Attribute
- Source a
- Refresh
- Item sea
- Save tre
- Shrink/E
- Select al
- Deselect
- Selected

Add new attribute Dialog:

- Attribute name: my first attribute
- Type: [Dropdown menu]
- Buttons: OK, Edit, Cancel

Attribute profile reference (ATTRIBUTEPROFILEREERENCE) List:

- Date
- Datetime
- Enumeration (ENUMERATION)
- Enumeration list (ENUMERATIONLIST)
- Expression (EXPRESSION)
- Floating number (DOUBLE)
- Integer (INTEGER)
- Intermodel reference (INTERREF)
- Longstring (LONGSTRING)
- Programcall (PROGRAMCALL)
- String (STRING)
- Table (RECORD)
- Time (TIME)

Modeling Language

▶ Demo (Class Attribute & Attribute)

▶ Function

- ▶ Node Name: String

▶ Database

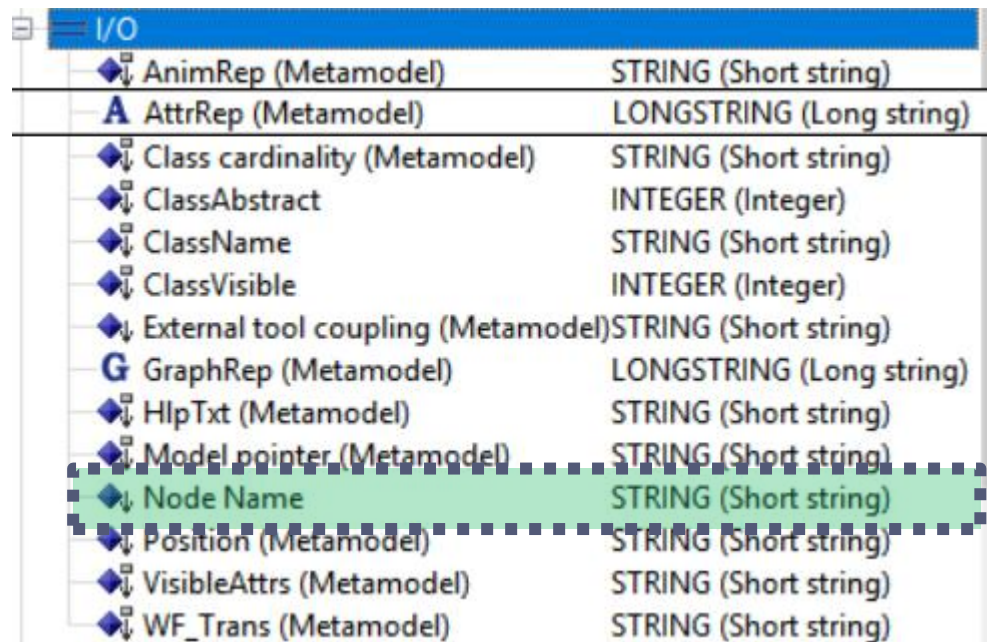
- ▶ Node Name: String

▶ I/O

- ▶ Node Name: String

▶ Flow

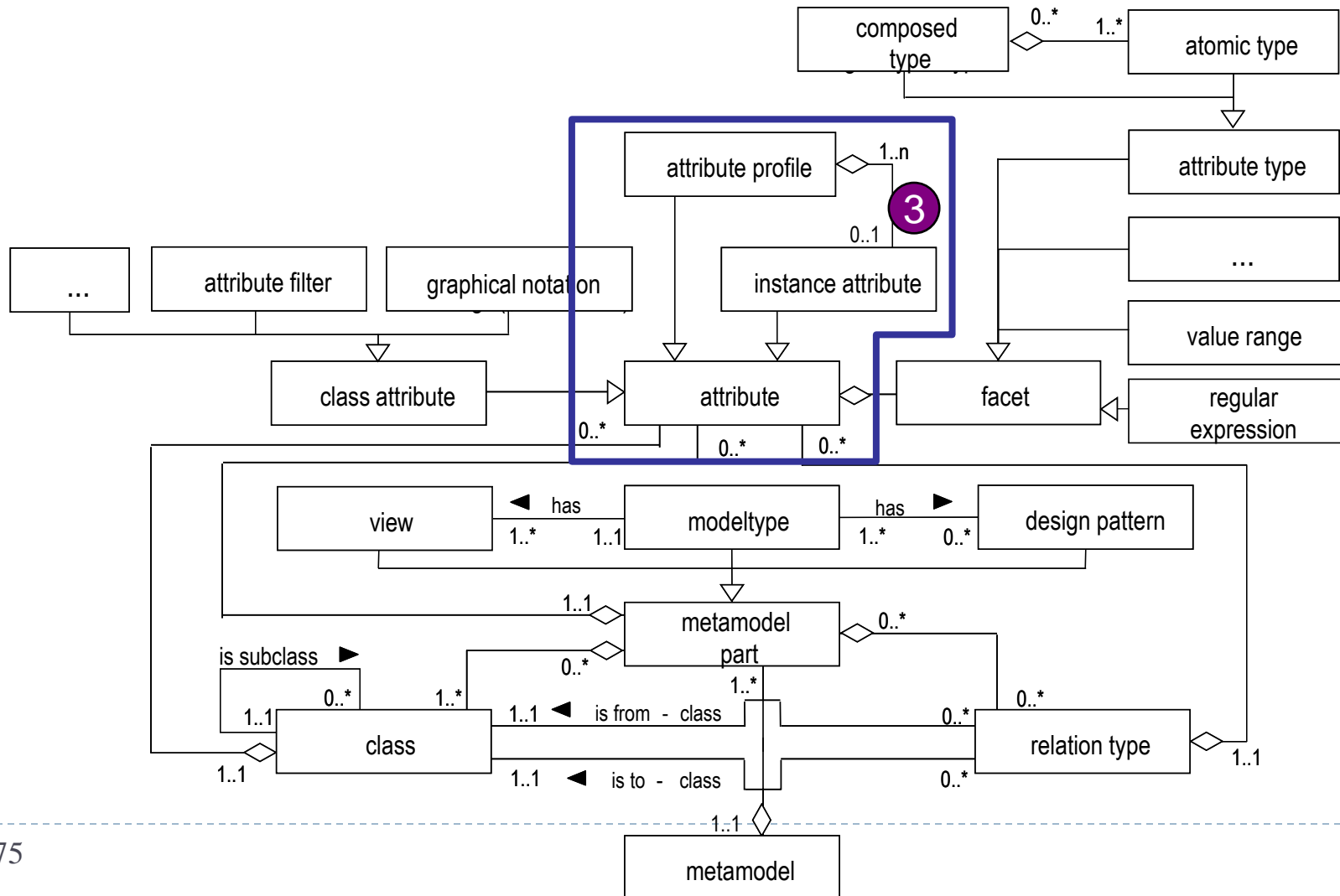
- ▶ Node Name: String



I/O	
AnimRep (Metamodel)	STRING (Short string)
AttrRep (Metamodel)	LONGSTRING (Long string)
Class cardinality (Metamodel)	STRING (Short string)
ClassAbstract	INTEGER (Integer)
ClassName	STRING (Short string)
ClassVisible	INTEGER (Integer)
External tool coupling (Metamodel)	STRING (Short string)
GraphRep (Metamodel)	LONGSTRING (Long string)
HlpTxt (Metamodel)	STRING (Short string)
Model pointer (Metamodel)	STRING (Short string)
Node Name	STRING (Short string)
Position (Metamodel)	STRING (Short string)
VisibleAttrs (Metamodel)	STRING (Short string)
WF_Trans (Metamodel)	STRING (Short string)

Modeling Language

③ Special Class Attribute & Attribute



Modeling Language

► Special Attributes

GraphRep: Graphical representation (object- and relation classes)

AttrRep: Notebook-Definition (all classes)

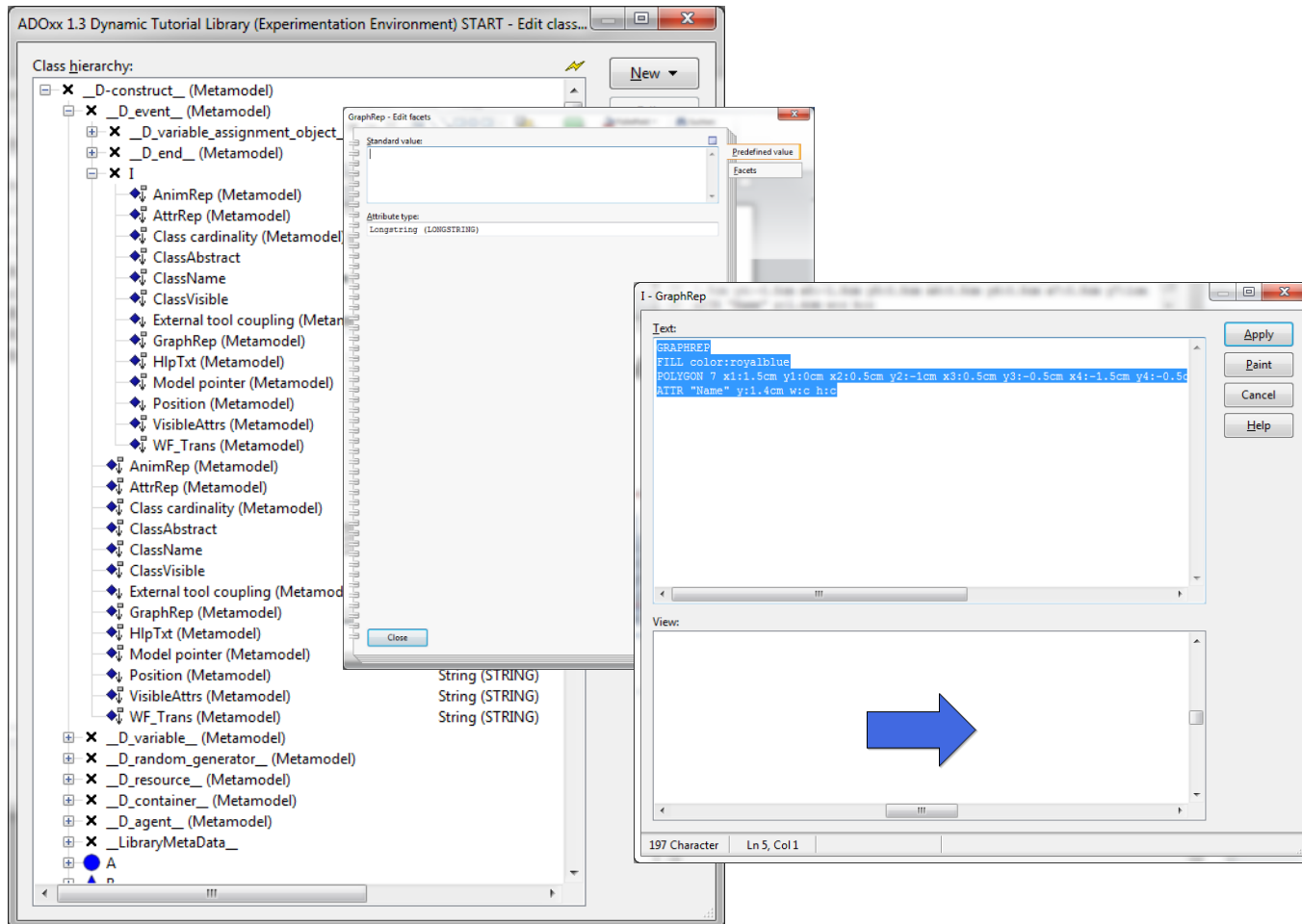
Model pointer: Relations to other models (object classes)

Class cardinality: Relation constraints (object classes)

__Conversion__: Conversion from one object to another

Modeling Language

► GraphRep

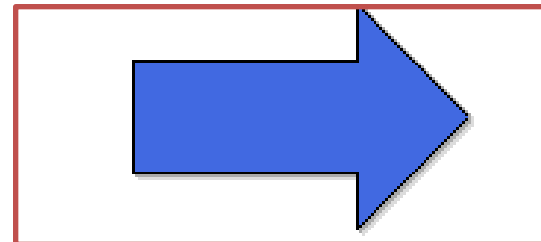


Modeling Language

► GraphRep

```
GRAPHREP  
FILL color:royalblue  
POLYGON 7 x1:1.5cm y1:0cm x2:0.5cm  
y2:-1cm x3:0.5cm y3:-0.5cm x4:-1.5cm  
y4:-0.5cm x5:-1.5cm y5:0.5cm  
x6:0.5cm y6:0.5cm x7:0.5cm y7:1cm
```

```
ATTR "Name" y:1.4cm w:c h:c
```



In case attribute name is available, it is shown here

Modeling Language

► Demo (GraphRep)

Text:

```
GRAPHREP
```

```
ATTR "Node Name" w:c h:c
```

```
RECTANGLE x:-1.5cm y:-0.5cm w:3cm h:1cm
```



Text:

```
GRAPHREP
```

```
ATTR "Node Name" w:c h:c
```

```
LINE x1:-1.5cm y1:-0.5cm x2:1.5cm y2:-0.5cm
```

```
LINE x1:-1.5cm y1:0.5cm x2:1.5cm y2:0.5cm
```



Text:

```
GRAPHREP
```

```
ELLIPSE x:0.00cm y:0cm rx:1cm ry:1cm
```

```
ATTR "Node Name" w:c h:c
```



Text:

```
GRAPHREP
```

```
SHADOW off
```

```
PEN style:solid w:0.05cm
```

```
EDGE
```

```
START
```

```
MIDDLE
```

```
ATTR "Node Name" x:0cm y:-0.2cm w:c h:c
```

```
END
```

```
FILL color:black
```

```
POLYGON 4 x1:0.0cm y1:0cm x2:-0.2cm y2:-0.1cm x3:-0.1cm y3:0cm x4:-0.2cm y4:0.1cm
```



Modeling Language

- ▶ AttrRep
 - ▶ Notebook
 - ▶ Chapter
 - ▶ Group
 - ▶ Attribute

Respond to Customer (Task)

Task relevance

- ☐ Auditing
- ☐ Monitoring

☐ For compensation

Loop details

Loop type

- ☒ Not specified
- ☐ Standard
- ☐ Multi-instance

Loop condition (standard):

Multi-instance loop details

- ☐ Sequential execution

Cardinality:

Referenced data input:

Referenced data output:

Completion condition:

Send/Receive task details

Referenced message:

Receive task details

- ☐ Instantiate

Close Reset

1/2

Object properties

Documents

Times/Costs

Simulation settings

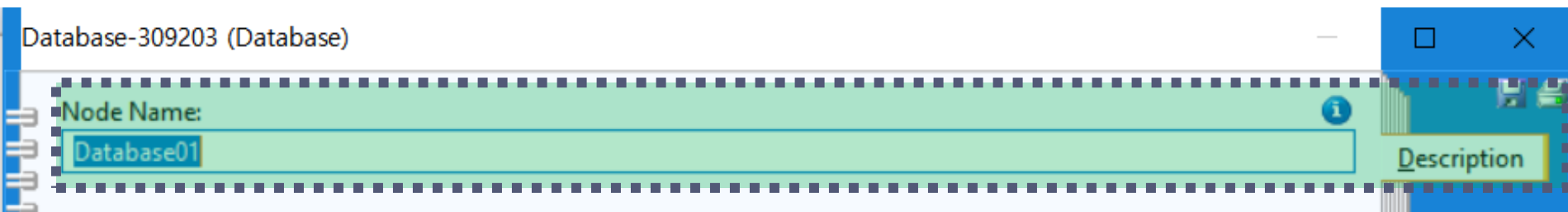
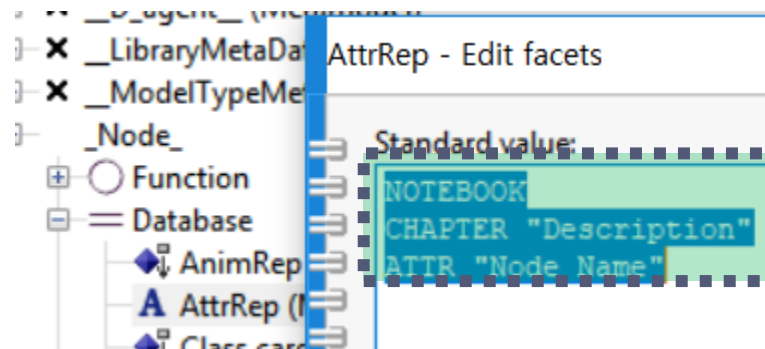
Simulation results

► AttrRep



Modeling Language

► Demo (AttrRep)



Modeling Language

NOTEBOOK

CHAPTER "Definition"

ATTR "Name"

GROUP "Definition"

ATTR "Description"

ATTR "External content"

ENDGROUP

Chapter
Structure

Attributes

Grouping of
attributes on
same chapter

Modeling Language

- ▶ **Class Cardinalities**

- ▶ The minimal/maximal number of objects of this class per model
- ▶ The minimal/maximal number of relations of a specific type

Modeling Language

CARDINALITIES

The cardinality definition must start with this command to be valid. It has no parameters.

RELATION *"RelationName"*

Restricts the following commands to the relation class with the name *<RelationName>*.

FROM_CLASS *"ClassName"* / TO_CLASS *"ClassName"*

Restricts the following commands to relations with the class of *<ClassName>*.

min-objects / max-objects

Specifies how many objects of a class can minimally/maximally be available in the model.

min-relations / max-relations

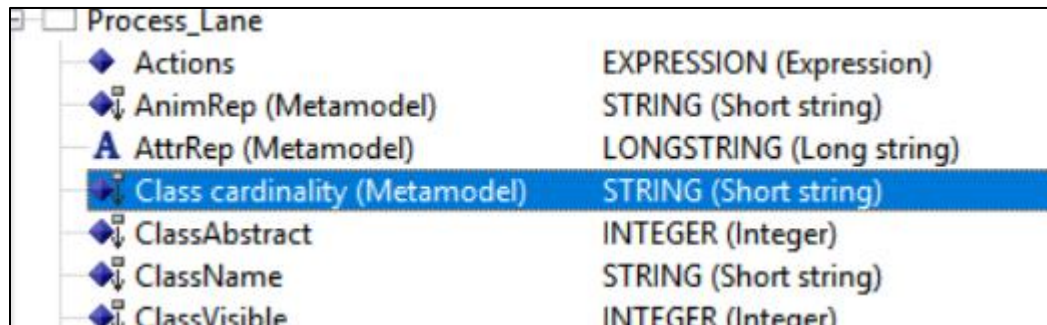
Specifies the minimal/maximal number of relations which can be connected with this object from this class.

max-outgoing / min-outgoing / max-incoming / min-incoming

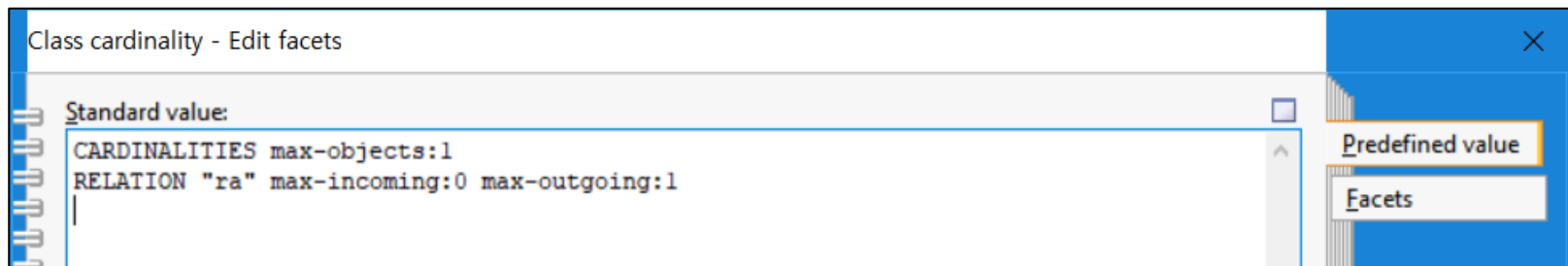
Restricts the number of allowed incoming/outgoing relations; either:

Modeling Language

- ▶ Example
 - ▶ Edit class cardinality attribute
 - ▶ CARDINALITIES max-objects:1



Process_Lane	
Actions	EXPRESSION (Expression)
AnimRep (Metamodel)	STRING (Short string)
AttrRep (Metamodel)	LONGSTRING (Long string)
Class cardinality (Metamodel)	STRING (Short string)
ClassAbstract	INTEGER (Integer)
ClassName	STRING (Short string)
ClassVisible	INTEGER (Integer)



Class cardinality - Edit facets

Standard value:

```
CARDINALITIES max-objects:1
RELATION "ra" max-incoming:0 max-outgoing:1
```

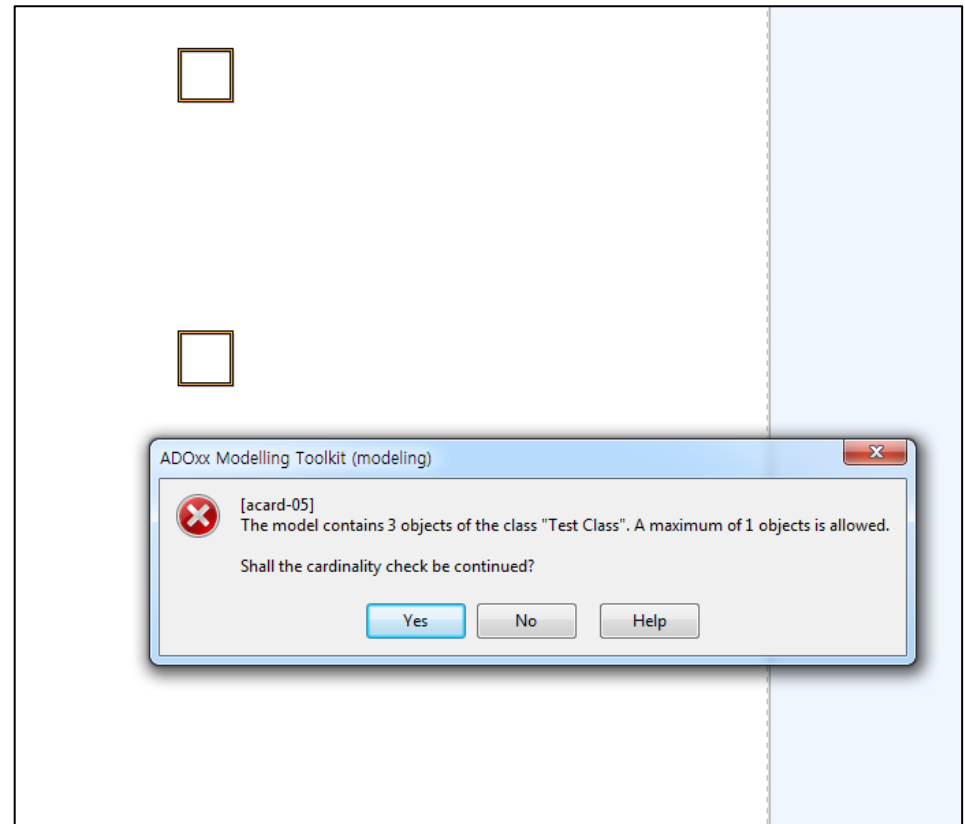
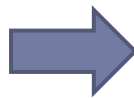
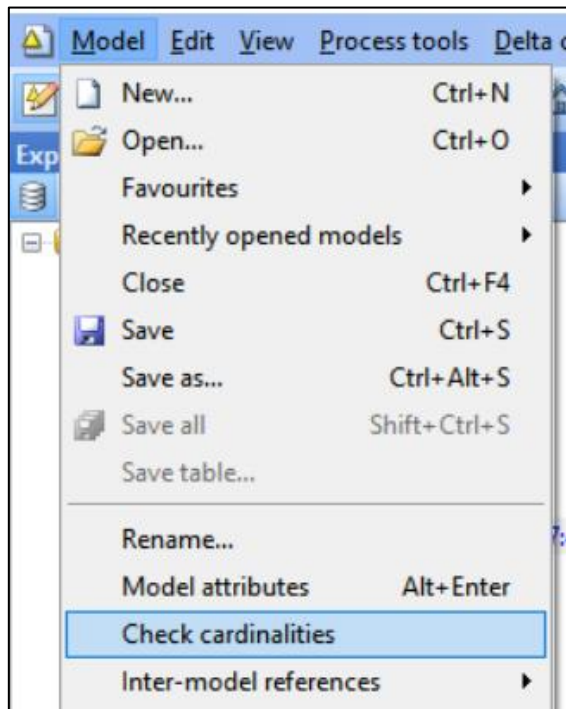
Predefined value

Facets

Modeling Language

► Example

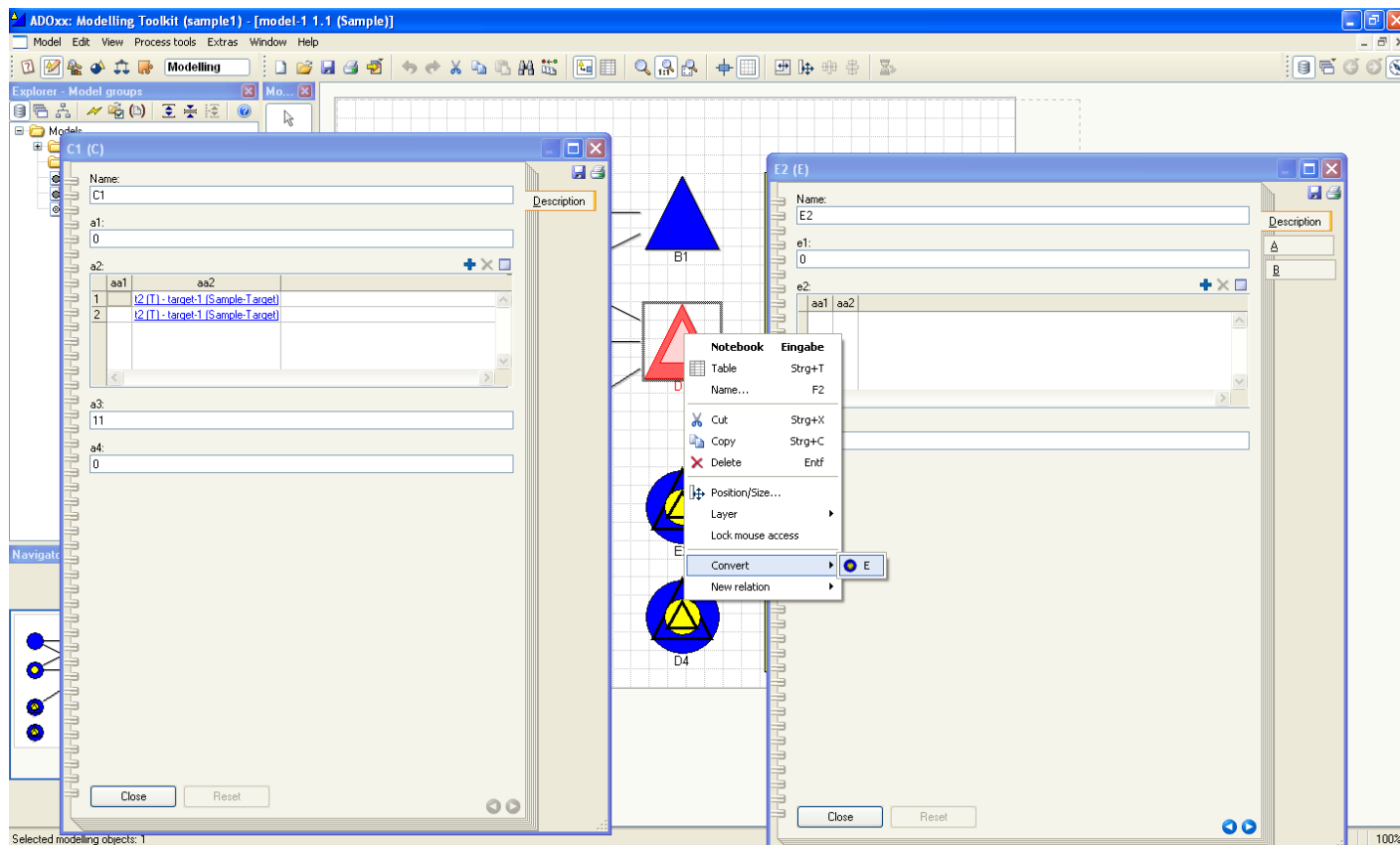
- Click “Check cardinalities” in the Modeling toolkit



Modeling Language

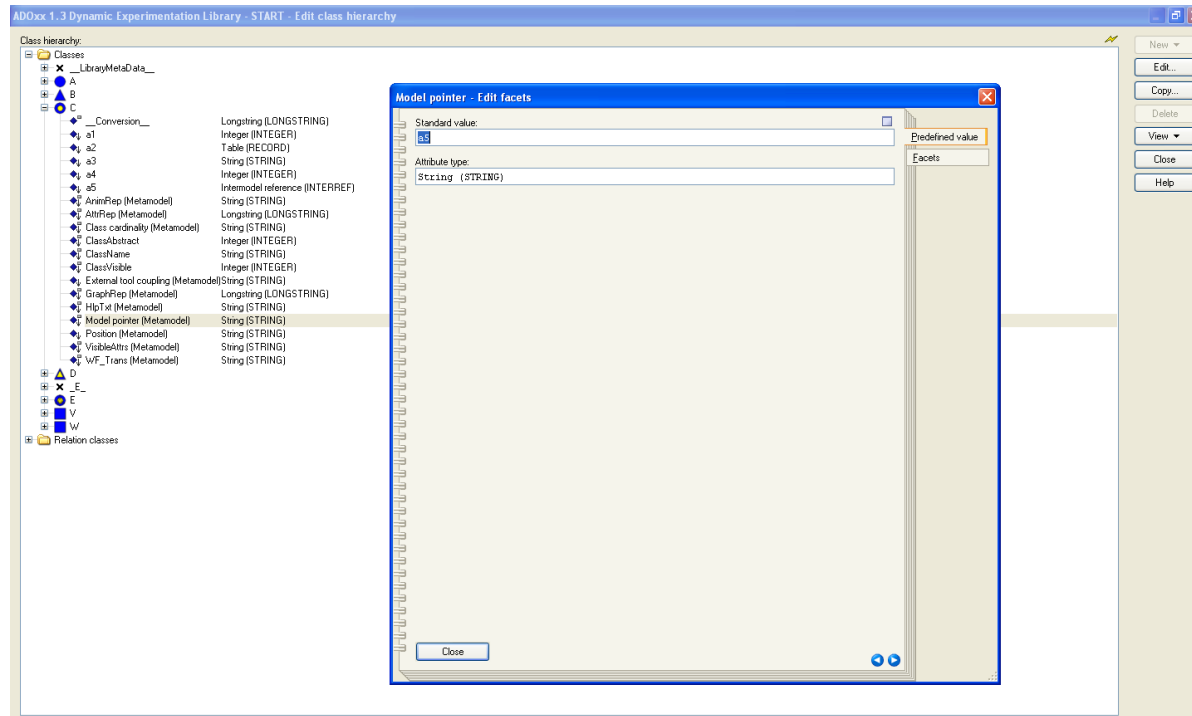
► __Conversion__

- Conversion of a modeling object from one class to another



Modeling Language

- ▶ Model pointer
 - ▶ Directly to another model
 - ▶ Using interref attribute
 - ▶ <Ctrl> + double click



④ Attribute Facets



Modeling Language

▶ Attribute Facets

- ▶ Defined attributes can be further detailed in ADOxx using attribute facets.
- ▶ For different attribute types, attributes facets are defined in platform to specify
 - ▶ e.g. the value domain, restrictions, helptexts etc.
- ▶ 6 types of facet
 - ▶ Attribute Domain Definition
 - ▶ Regular Expression Definition
 - ▶ InterRef Domain Definition
 - ▶ Enumeration Domain Definition
 - ▶ MultiLineString Definition
 - ▶ Attribute Help Text Definition

Modeling Language

► Attribute Facets

	AttributeNumeric Domain	AttributeRegular Expression	AttributeInterref Domain	Enumeration Domain	MultiLineString	AttributeHelp Text	RecordClass Name	RecordClass Multiplicity
INTEGER	X					X		
DOUBLE	X					X		
STRING		X			X	X		
LONGSTRING		X			X	X		
TIME						X		
ENUMERATION		X		X		X		
ENUMERATIONLIST		X		X		X		
PROGRAMCALL				X		X		
RECORD						X	X	X
EXPRESSION					X	X		
INTERREF			X			X		

Modeling Language

► Demo (Attribute Help Text)

The image shows two overlapping windows from a software application. The top window, titled "Node Name - Edit facets", has a tab labeled "MultiLineString" and a text area for "AttributeHelpText" containing the text "This attribute is to visulize a information of object on Drawing area". To the right of this window is a blue sidebar with a close button (X) and two tabs: "Predefined value" and "Facets". The bottom window, titled "Database-309203 (Database)", has a "Node Name:" field with the value "Databse". A green dashed box highlights an information icon (i) in the top right corner of this window. Below it, a smaller window titled "Database-309203 (Database) - Information on Node Name" is open, showing the same text "This attribute is to visulize a information of object on Drawing area" in a green dashed box. A "Close" button is visible in the bottom right corner of this smaller window.

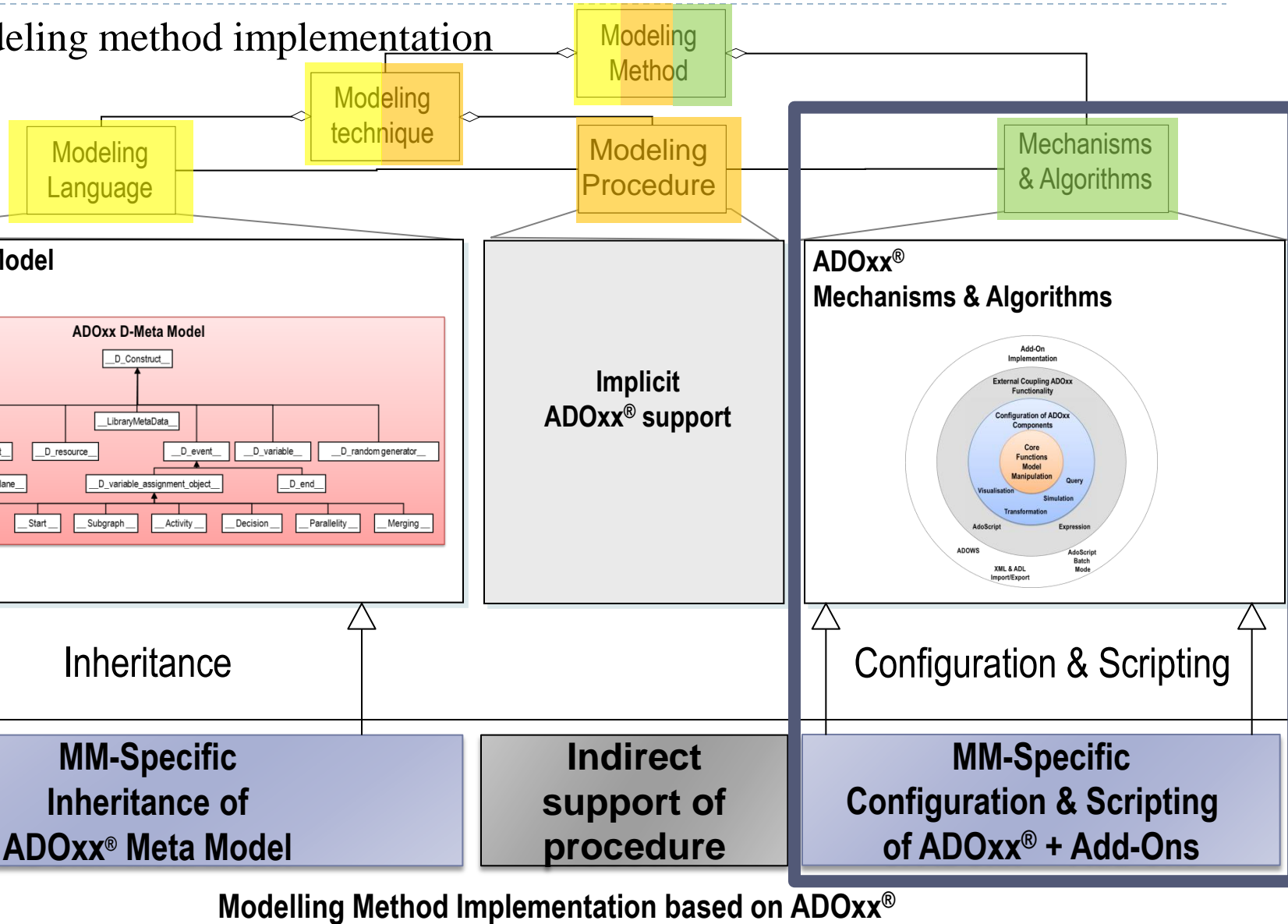
1. Overview
 - 1) Meta Model
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language
 - 3) Core Functions**
 - 4) Adoscript
 - 5) Simulation

3)

CORE FUNCTIONS

Core Functions

► Modeling method implementation

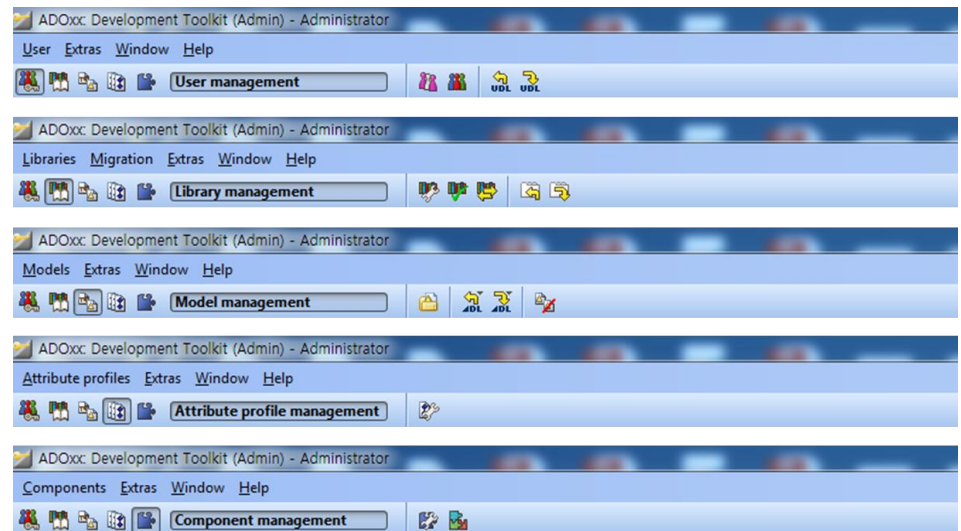


Core Functions

- ▶ Core Functions for Model Manipulation
 - ▶ Database
 - ▶ Visualization
 - ▶ Query
 - ▶ Transformation

Core Functions

- ▶ Database
 - ▶ ADOxx Development Toolkit
 - ▶ User Management
 - ▶ Model Management
 - ▶ Library Management
 - ▶ Component Management



Modeling Language

► Demo (User management)

The screenshot displays two windows from a software application. The background window, titled "User management - User list", shows a table of registered users. The foreground window, titled "User management - Definition of a new user", is a dialog for creating a new user account.

User management - User list

Registered ADOxx users:	
Admin	ADOxx 1.5 Exper...
cps	Cyber Physical Sy...
gts	SAVE 3.01
gts_paper	SAVE Paper Work...
gts2	SAVE V1.18 for te...
Manager01	Mini Data Flow D...
temp	SmartCity-I-Mod...
test	SmartCity-1-Mox...
week1	2019-2 tuto.
zzz	VHDL Experimen...

User management - Definition of a new user

User name:

Password:

Confirm password:

Application library:

Authorisation

☐ Development Toolkit

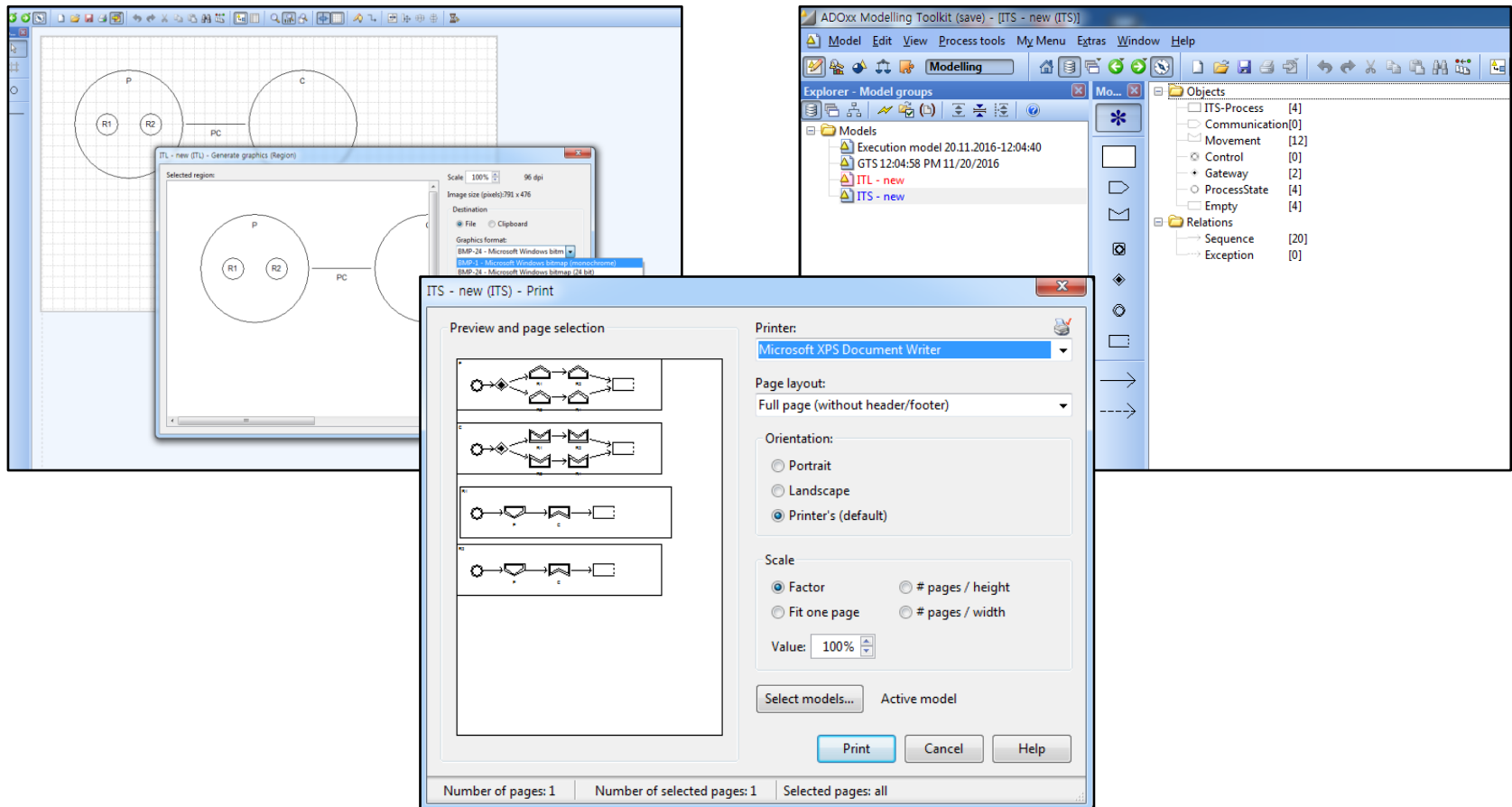
Buttons: Add, User group..., Component access..., Model groups..., Cancel, Help

Core Functions

- ▶ Visualization
 - ▶ Graphical Presentation of models in the user interface
 - ▶ Drag and Drop: Creation and Move, Delete, Edit
 - ▶ Cardinality conformity check
 - ▶ Notebook representation
 - ▶ Zoom Functionality (zoom, world-area, right mouse, etc.)

Core Functions

► Visualization

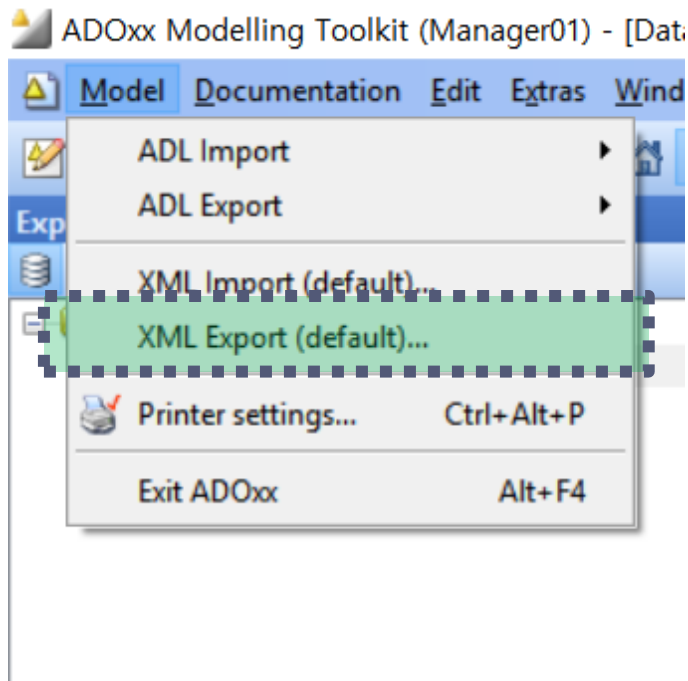


Core Functions

- ▶ Transformation
 - ▶ Generation of ADL
 - ▶ Text file in complimentary ADOxx Definition Language
 - ▶ Generation of XML
 - ▶ Text file in complimentary ADOxx defined XML syntax

Modeling Language

► Demo (Transformation)



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ADOXML SYSTEM "adoxml31.dtd">
<ADOXML version="3.1" date="22.09.2019" time="23:4
<MODELS>
<MODEL id="mod.308402" name="Data Flow Diagram - 1
<MODELATTRIBUTES>
<ATTRIBUTE name="Version number" type="STRING"></A
<ATTRIBUTE name="Author" type="STRING">Manager01<
<ATTRIBUTE name="Creation date" type="STRING">22.0
<ATTRIBUTE name="Date last changed" type="STRING">
<ATTRIBUTE name="Last user" type="STRING">Manager(
<ATTRIBUTE name="Keywords" type="STRING"></ATTRIBU
<ATTRIBUTE name="Comment" type="STRING"></ATTRIBU
<ATTRIBUTE name="Model type" type="ENUMERATION">C
<ATTRIBUTE name="State" type="ENUMERATION">In pro
<ATTRIBUTE name="Reviewed on" type="STRING"></ATT
<ATTRIBUTE name="Reviewed by" type="STRING"></ATT
<ATTRIBUTE name="Description" type="STRING"></ATT
<ATTRIBUTE name="Number of objects and relations"
<ATTRIBUTE name="World area" type="STRING">w:20.8
<ATTRIBUTE name="Grid" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="Zoom" type="INTEGER">100</ATTRIBU
<ATTRIBUTE name="Viewable area" type="STRING">VIEW
GRAPHIC x:-16 y:-16 w:1153 h:717 scale:1
TABLE
</ATTRIBUTE>
```

Core Functions

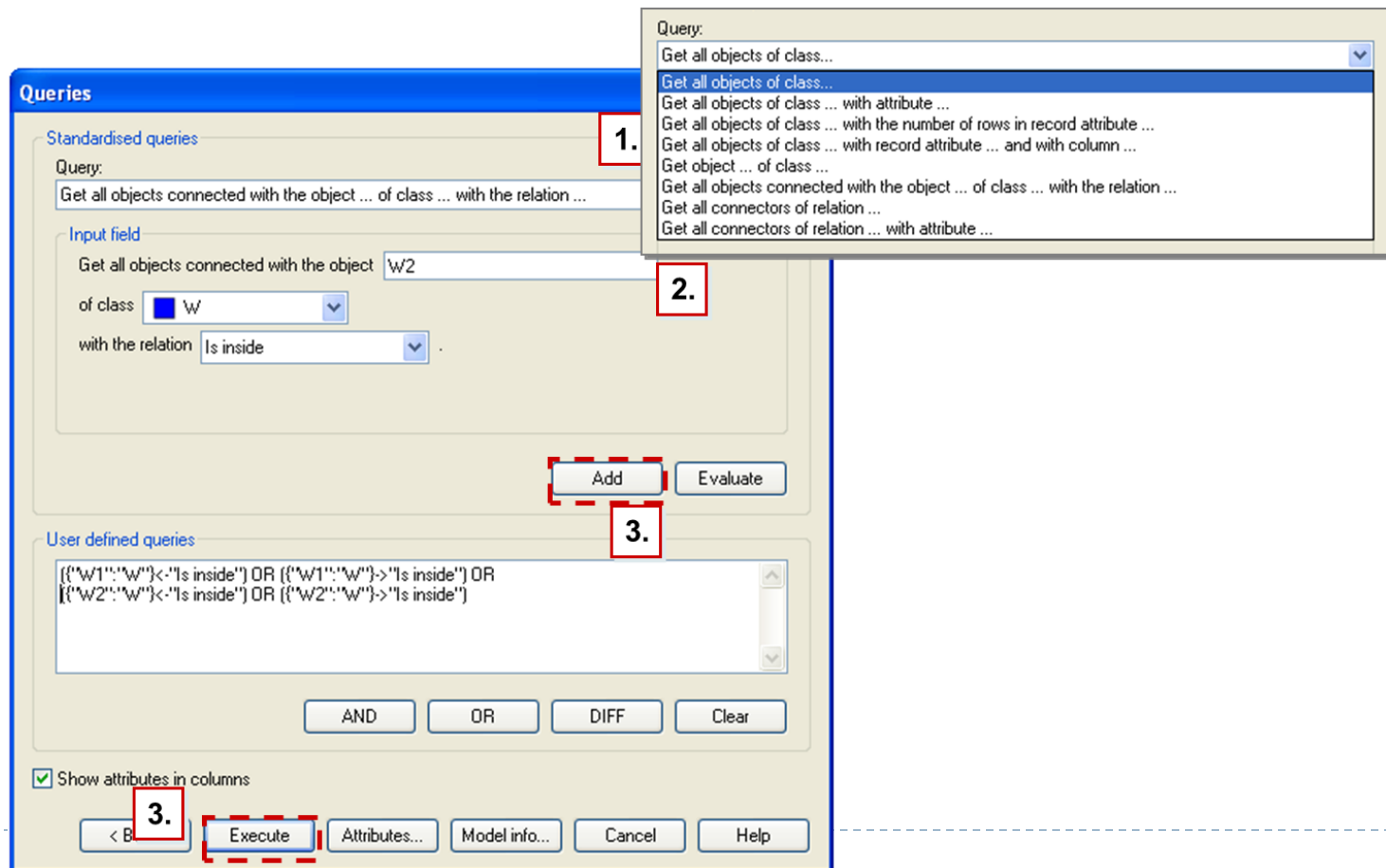
► ADL and XML sample

```
INSTANCE <E1> : <E>
  ATTRIBUTE <Position>
  VALUE "NODE x:4cm y:11cm w:2cm h:2cm index:1"
  ATTRIBUTE <External tool coupling>
  VALUE ""
  ATTRIBUTE <a1>
  VALUE 0
  ATTRIBUTE <a2>
  VALUE
  ATTRIBUTE <a3>
  VALUE ""
  ATTRIBUTE <b1>
  VALUE 0
  ATTRIBUTE <b2>
  VALUE
  ATTRIBUTE <b3>
  VALUE ""
  ATTRIBUTE <e1>
  VALUE 0
  ATTRIBUTE <e2>
  VALUE
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE ADOXML (View Source for full doctype...)>
- <ADOXML version="3.1" date="28.06.2012" time="13:32" database="adoxx13" username="sample1" adoversion="Version 1.0">
- <MODELS>
- <MODEL id="mod.13813" name="model-1" version="1.1" modeltype="Sample" libtype="bp" applib="ADOxx 1.3 Dynamic Experimentation Library - START">
+ <MODELATTRIBUTES>
- <INSTANCE id="obj.13814" class="E" name="E1">
+ <ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:11cm w:2cm h:2cm index:1</ATTRIBUTE>
+ <ATTRIBUTE name="External tool coupling" type="STRING" />
+ <ATTRIBUTE name="a1" type="INTEGER">0</ATTRIBUTE>
+ <RECORD name="a2" />
+ <ATTRIBUTE name="a3" type="STRING" />
+ <ATTRIBUTE name="b1" type="INTEGER">0</ATTRIBUTE>
+ <RECORD name="b2" />
+ <ATTRIBUTE name="b3" type="STRING" />
+ <ATTRIBUTE name="e1" type="INTEGER">0</ATTRIBUTE>
+ <RECORD name="e2" />
+ <ATTRIBUTE name="e3" type="STRING">11</ATTRIBUTE>
+ <ATTRIBUTE name="a4" type="INTEGER">0</ATTRIBUTE>
+ <ATTRIBUTE name="b4" type="STRING" />
</INSTANCE>
+ <INSTANCE id="obj.13817" class="A" name="A1">
+ <INSTANCE id="obj.13826" class="B" name="B1">
+ <INSTANCE id="obj.13832" class="C" name="C-13010">
+ <INSTANCE id="obj.13835" class="D" name="D-13013">
+ <INSTANCE id="obj.16408" class="B" name="B-16408">
+ <INSTANCE id="obj.16604" class="V" name="V1">
+ <INSTANCE id="obj.17004" class="W" name="W1">
+ <INSTANCE id="obj.17007" class="B" name="B-16408-17007">
+ <INSTANCE id="obj.17291" class="E" name="E-17291">
+ <INSTANCE id="obj.17294" class="E" name="E-17294">
+ <INSTANCE id="obj.17297" class="E" name="E-17297">
+ <INSTANCE id="obj.17328" class="E" name="D-13013-17321">
+ <INSTANCE id="obj.17334" class="E" name="C-13010-17318">
+ <CONNECTOR id="con.13841" class="arB">
+ <CONNECTOR id="con.13842" class="arB">
+ <CONNECTOR id="con.13843" class="arB">
+ <CONNECTOR id="con.13844" class="arB">
+ <CONNECTOR id="con.13845" class="arB">
+ <CONNECTOR id="con.16607" class="Is inside">
</MODEL>
</MODELS>
</ADOXML>
```

Core Functions

- ▶ Query
 - ▶ AQL (Adoxx Query Language)



Modeling Language

► Demo (AQL)

Queries

Query scope

☐ Queries on models (model attributes)

☒ Queries on model contents

Standardised queries

Query:

Get all objects of class...

Input field

Get all objects of class Database

Add Evaluate

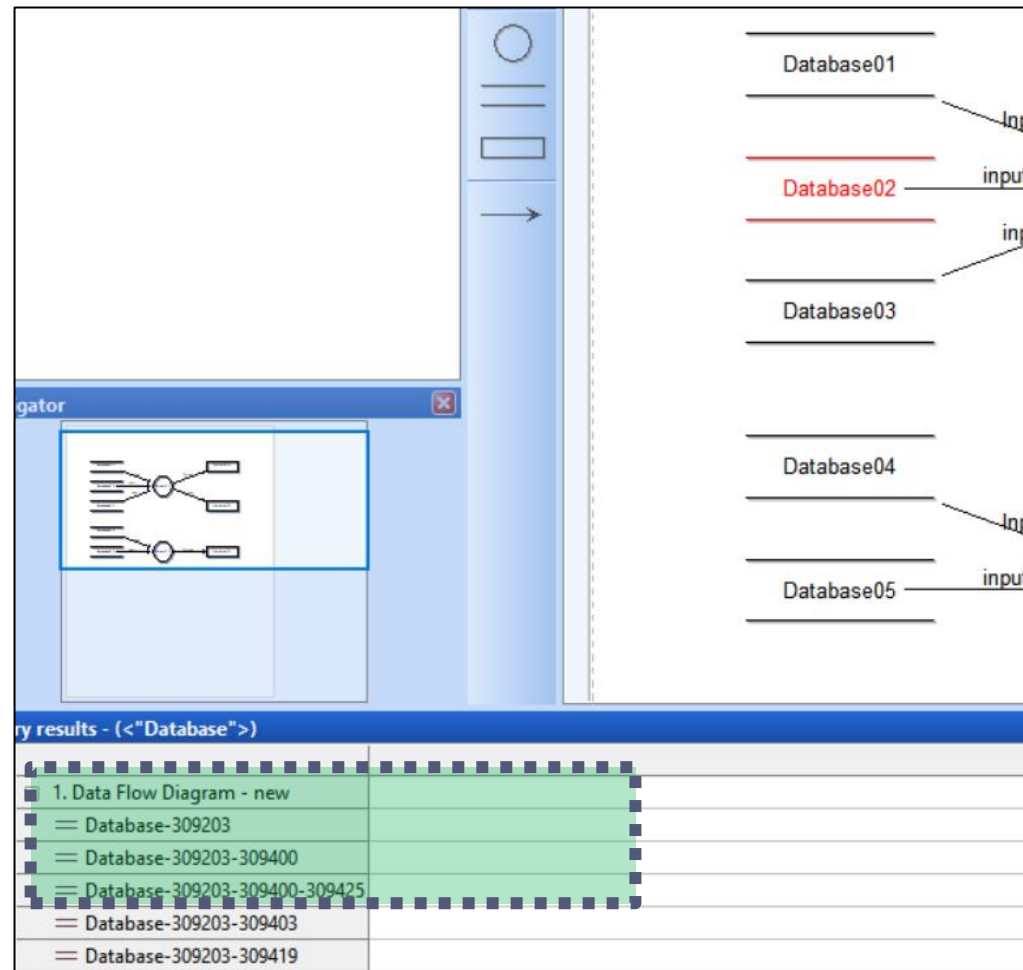
User defined queries

(<"Database">)

AND OR DIFF Clear

☒ Show attributes in columns

< Back Execute Attributes... Model info... Cancel Help



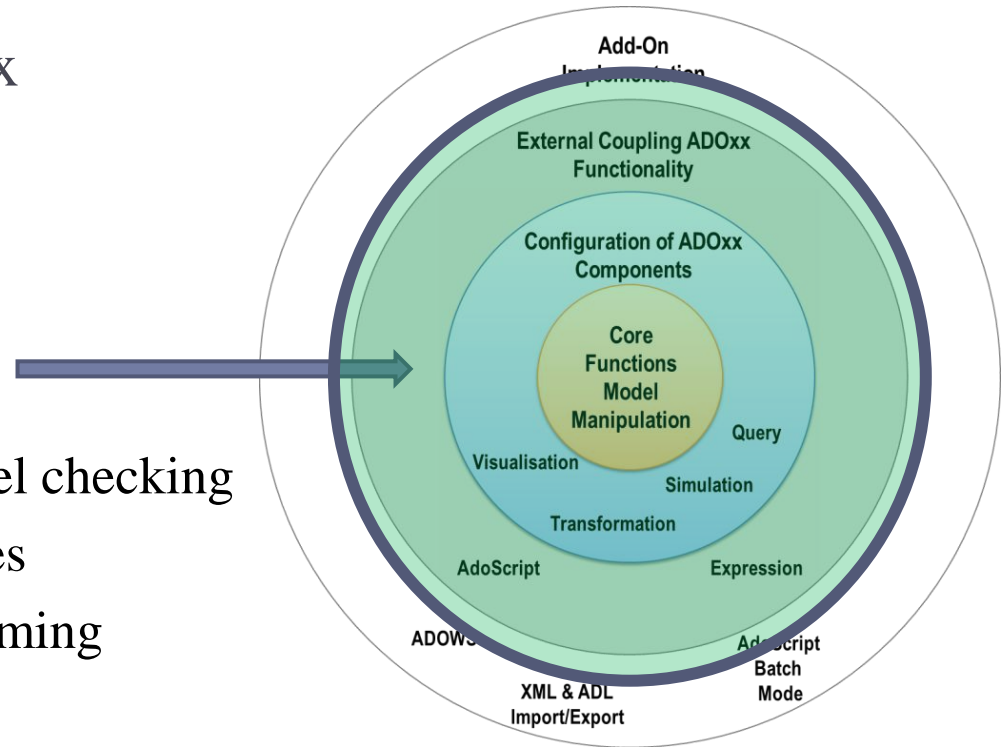
1. Overview
 - 1) Meta Model
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language
 - 3) Core Functions
 - 4) Adoscript**
 - 5) Simulation

4)

ADOSCRIPT

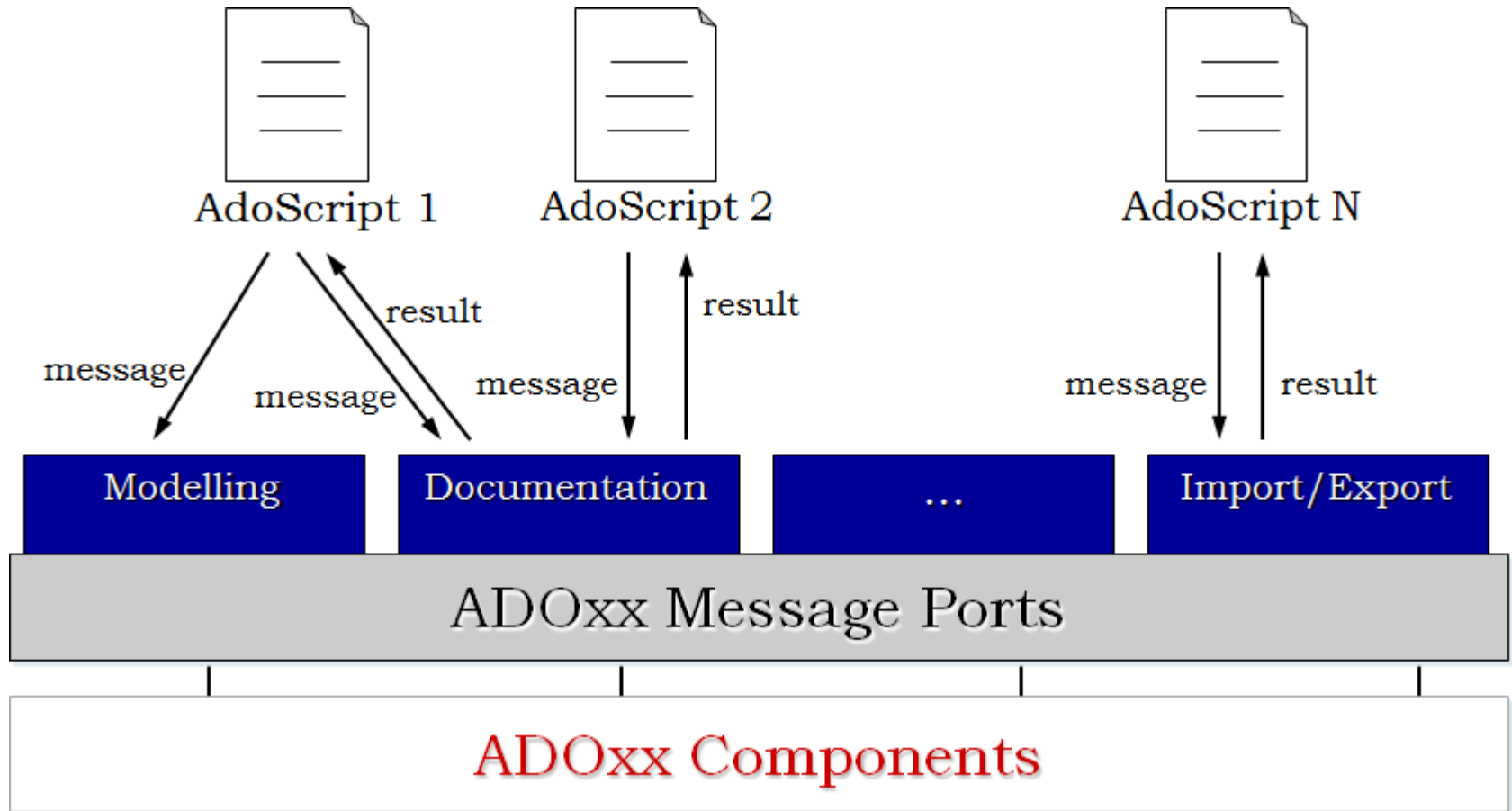
AdoScript

- ▶ AdoScript
 - ▶ Macro language of ADOxx
- ▶ Examples:
 - ▶ New menu entries
 - ▶ Integration of new tools
 - ▶ Realisation of specific model checking
 - ▶ Realisation of new interfaces
 - ▶ Additional add-on-programming



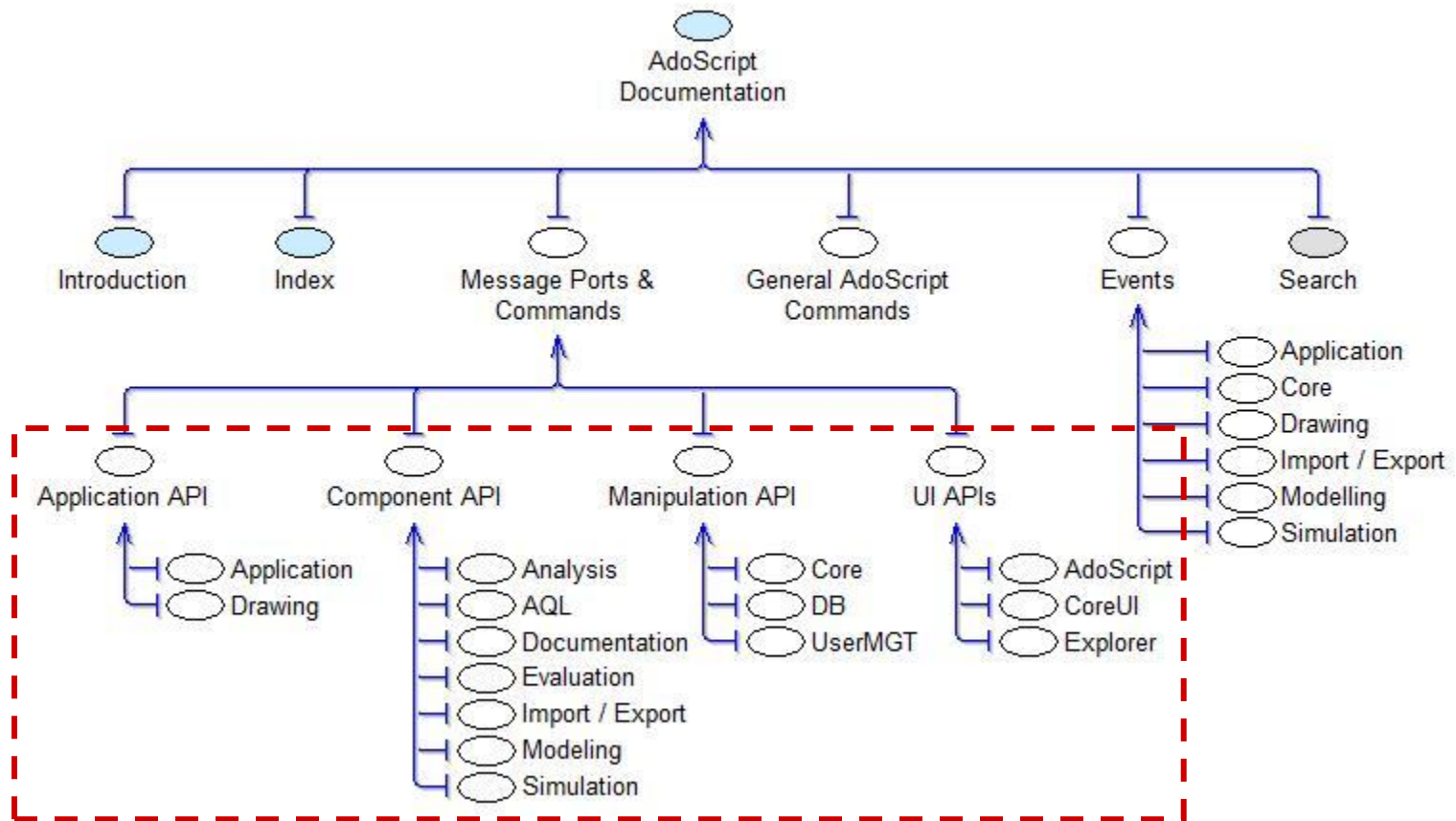
AdoScript

▶ AdoScript



AdoScript

► AdoScript



AdoScript

▶ ADOxx Homepage

▶ AdoScript Documentation

▶ <https://www.adoxx.org/AdoScriptDoc/index.html>

INTRODUCTION

INDEX

MESSAGE PORTS & COMMANDS

Introduction

APPLICATION APIs

APPLICATION

DRAWING

COMPONENT APIs

ANALYSIS

AQL

DOCUMENTATION

EVALUATION

IMPORT/EXPORT

MODELING

SIMULATION

MANIPULATION APIs

CORE

DB

USRMGt

UI APIs

AdoSCRIPT

INTRODUCTION

Summary

INTRODUCTION

WHAT IS AdoSCRIPT

USAGE OF AdoSCRIPT

INTEGRATION OF AdoSCRIPT

PROGRAMMABLE THROUGH SCRIPTING APIs

USEFUL TIPS

IMPROVEMENTS

AdoScript is the scripting language of ADOxx.

AdoScript can be executed on different ways, so it can be used where it AdoScript builds on the so-called "Message Port-Concept".

Method-specific development of functionalities is possible through scripti

Please provide your feedback and improvement suggestions for the AdoS

WHAT IS AdoSCRIPT

AdoScript is the scripting language of ADOxx. It is based on LEO, is build procedural, and allows extension possibilities with low programm

USAGE OF AdoSCRIPT

AdoScript can be executed on different ways, so it can be used where it is needed.

As menu entry

For manual execution (e.g. transformation procedures, evaluation scenarios)

In events

Modeling Language

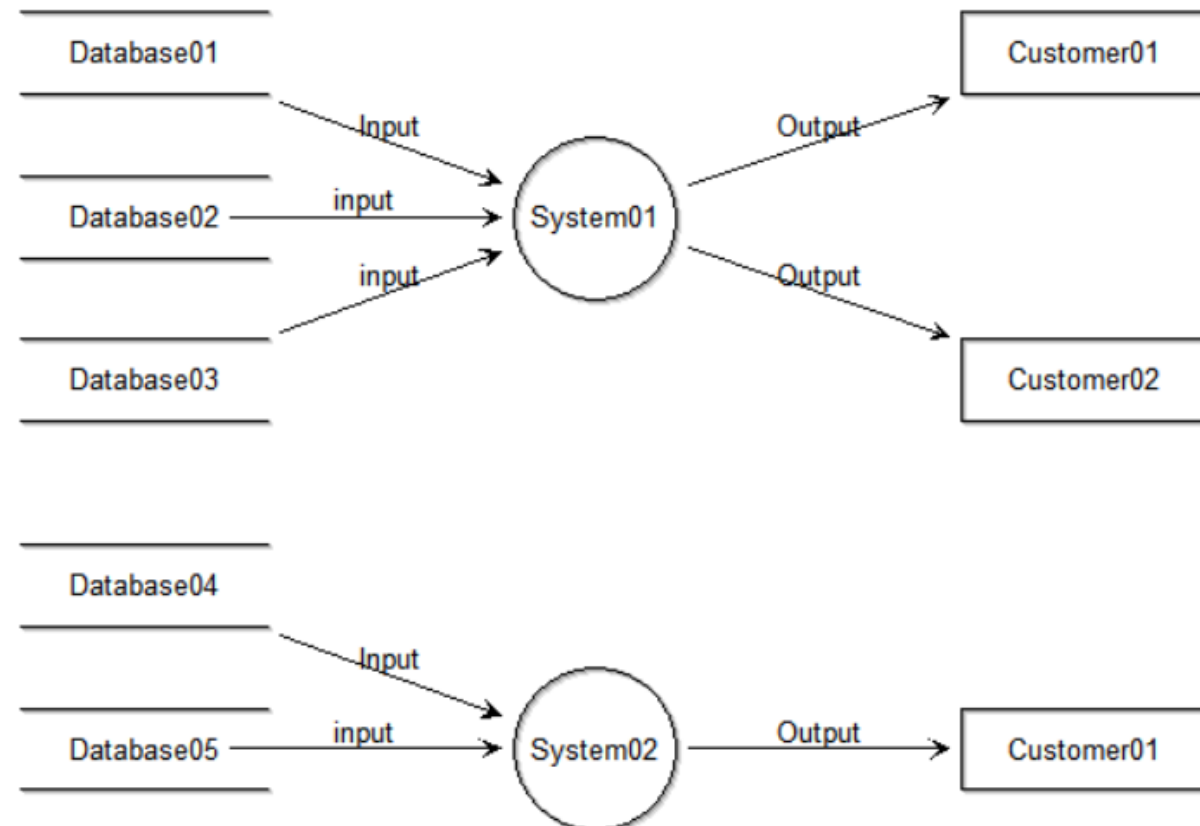
- ▶ Demo (AdoScript)
 - ▶ Get weight of all function objects

```
CC "Modeling" GET_ACT_MODEL
SETL nModelID:(modelid)

CC "Core" GET_ALL_OBJS_OF_CLASSNAME modelid:(nModelID) classname:("Function")
SET function_list:("")
FOR sObjId in:(objids)
{
    CC "Core" GET_CONNECTORS objid:(VAL sObjId) in
    SETL in_cnt:(tokcnt ( objids , " "))
    CC "Core" GET_CONNECTORS objid:(VAL sObjId) out
    SETL out_cnt:(tokcnt ( objids , " "))
    CC "Core" GET_ATTR_VAL objid:(VAL sObjId) attrname:("Node Name")
    SETL node_name:(val)
    SET function_list:(function_list+node_name+": "+"["+"input: "+STR in_cnt+"]"+"
"+"["+"output: "+STR out_cnt+"]"+"\\n")
}
CC "AdoScript" VIEWBOX text:(function_list) title:"result:" fontheight:20
```

Modeling Language

- ▶ Demo (AdoScript)
 - ▶ Get weight of all function objects

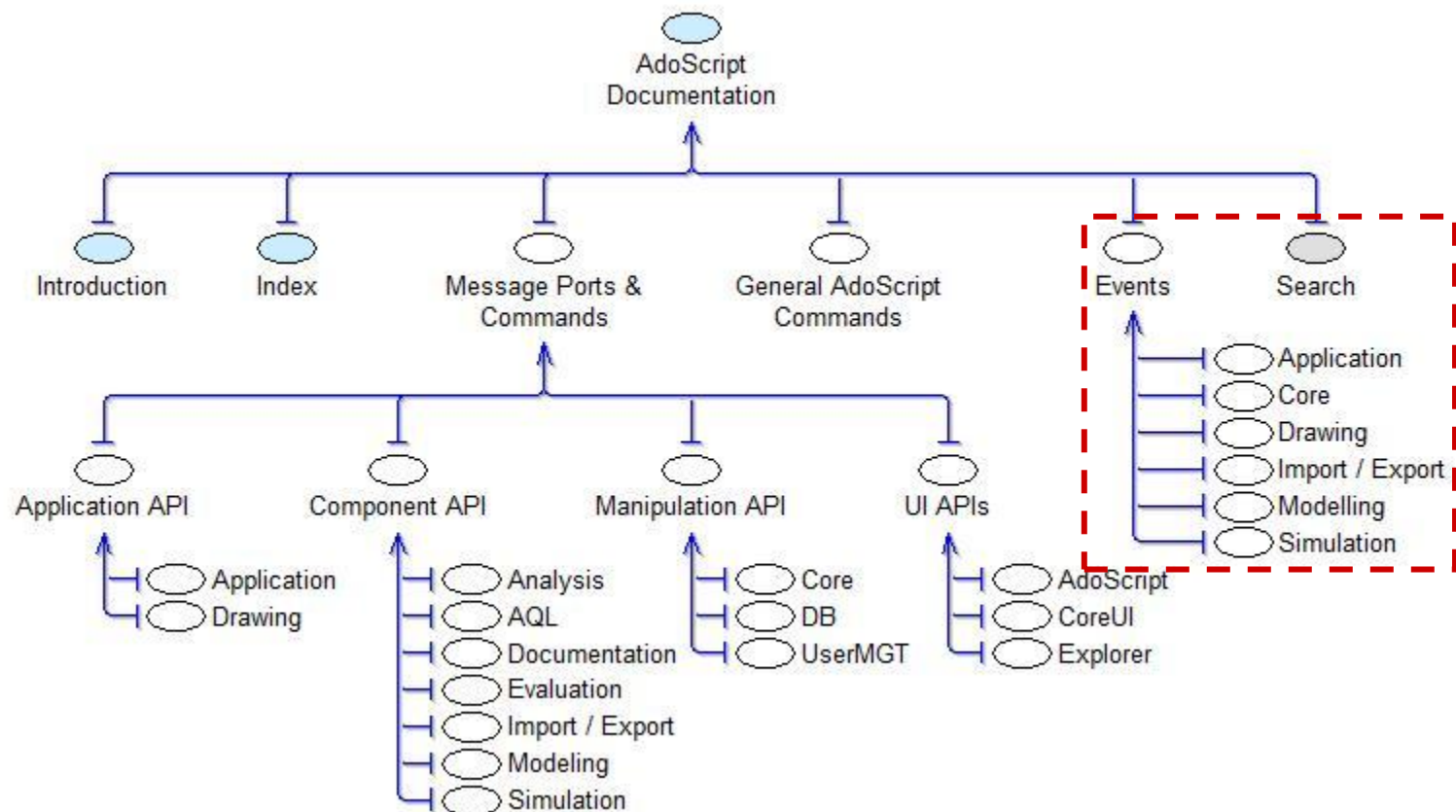


result:

```
System01: [input: 3] [output: 2]  
System02: [input: 2] [output: 1]
```


AdoScript

- ▶ Code execution
 - ▶ Event

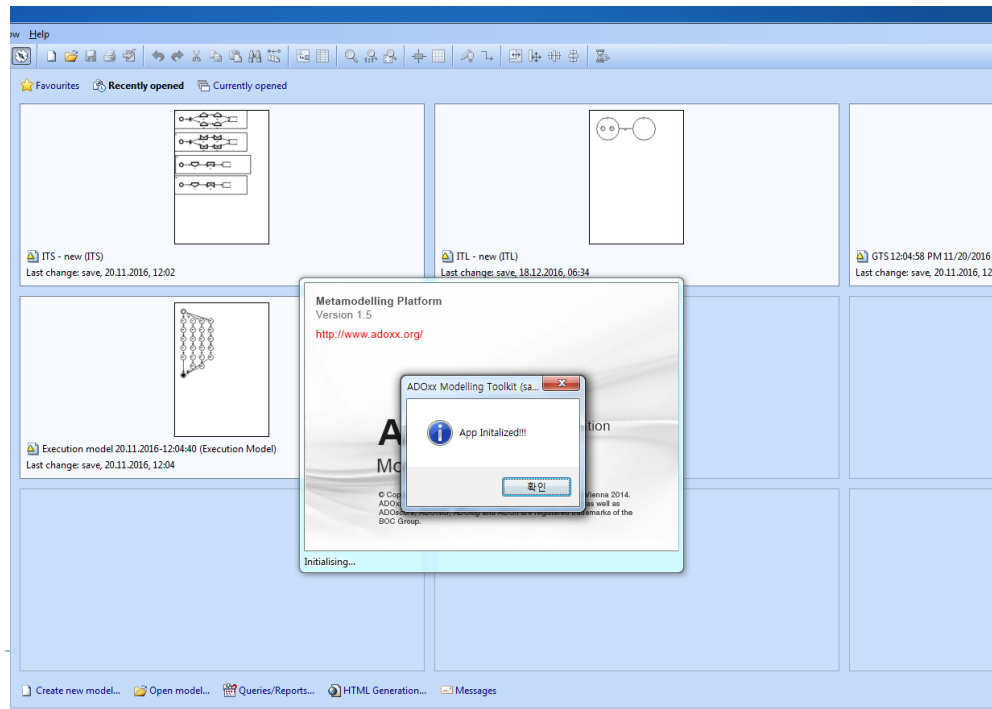


AdoScript

► Code execution

► Event

```
ON_EVENT "AppInitialized"  
{  
    CC "AdoScript" INFOBOX ("App Initialized!!!")  
}
```



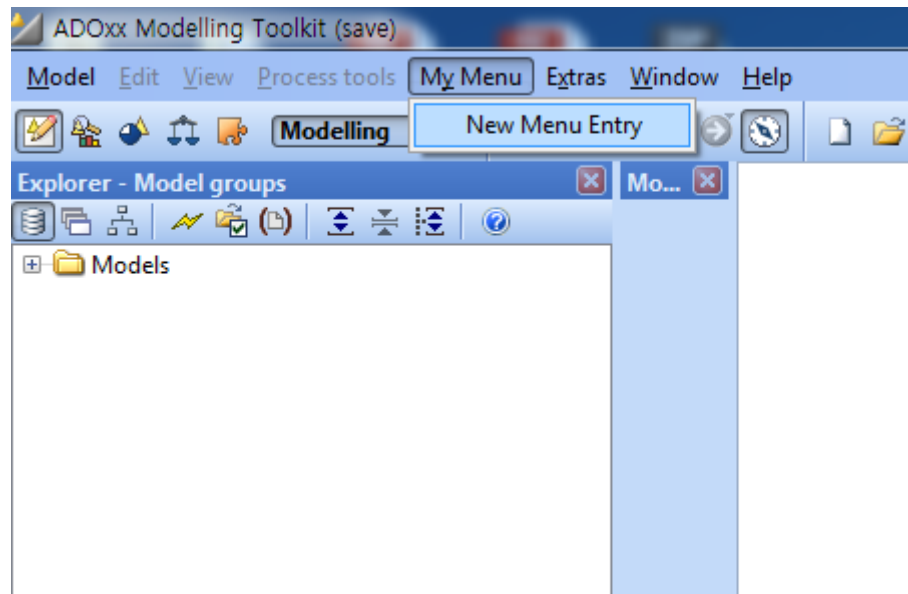
AdoScript

- ▶ Code execution
 - ▶ Event
 - ▶ AppInitialized
 - ▶ BeforeCreateRelationInstance
 - ▶ CreateInstance
 - ▶ DelateInstance
 - ▶ BeforeDeleteInstance
 - ▶ SetAttrivuteValue
 - ▶ ...

AdoScript

- ▶ Code execution
 - ▶ Menu entry

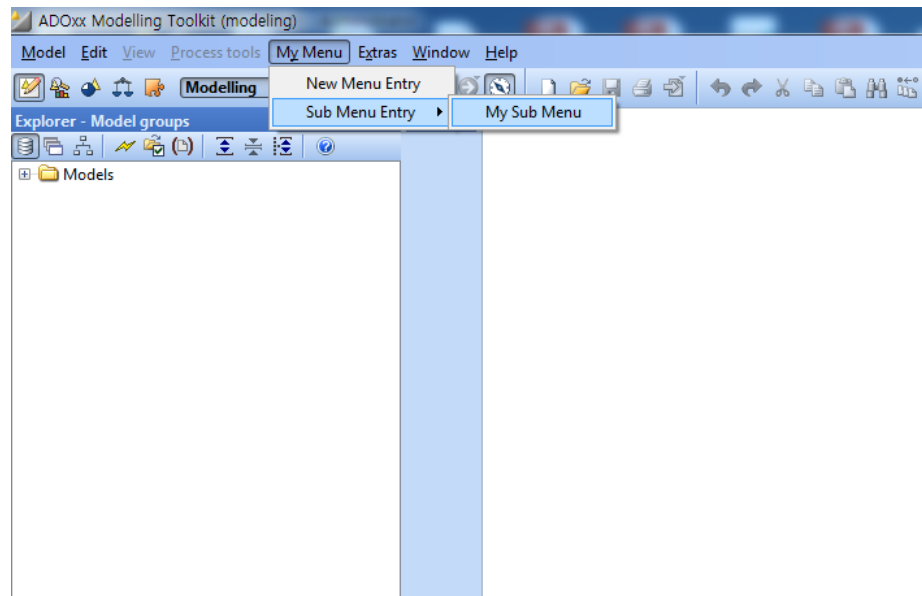
```
ITEM "New Menu Entry" modeling:"My Menu"  
EXECUTE file:("C:\\adoscript.asc")
```



AdoScript

- ▶ Code execution
 - ▶ Sub Menu entry

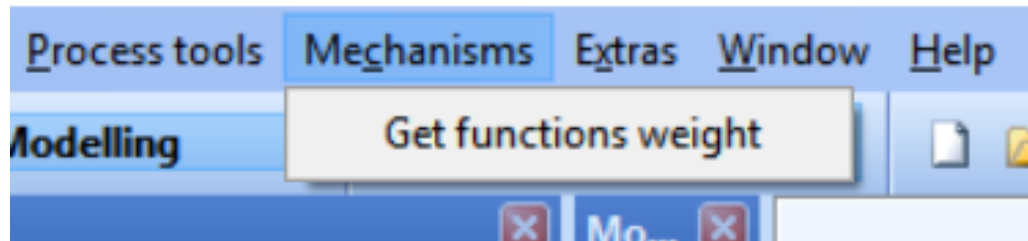
```
ITEM "My Sub Menu" modeling:"My Menu" sub-of:"Sub Menu Entry"  
EXECUTE file:("C:\\adoscript.asc")
```



Modeling Language

► Demo (Menu entry)

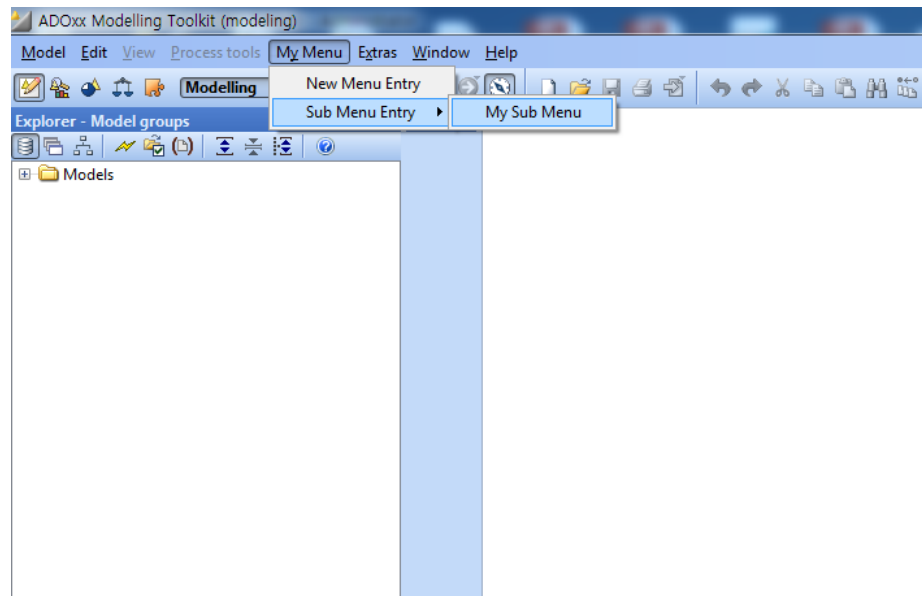
```
External coupling:
{
  SETG adoscript:(text)
  EXECUTE (text)
}
}
ITEM "Get functions weight" modeling:"Mechanisms"
EXECUTE file:("db:\\AdoScript_weight_example.asc")
```



AdoScript

- ▶ Code execution
 - ▶ Sub Menu entry

```
ITEM "My Sub Menu" modeling:"My Menu" sub-of:"Sub Menu Entry"  
EXECUTE file:("C:\\adoscript.asc")
```



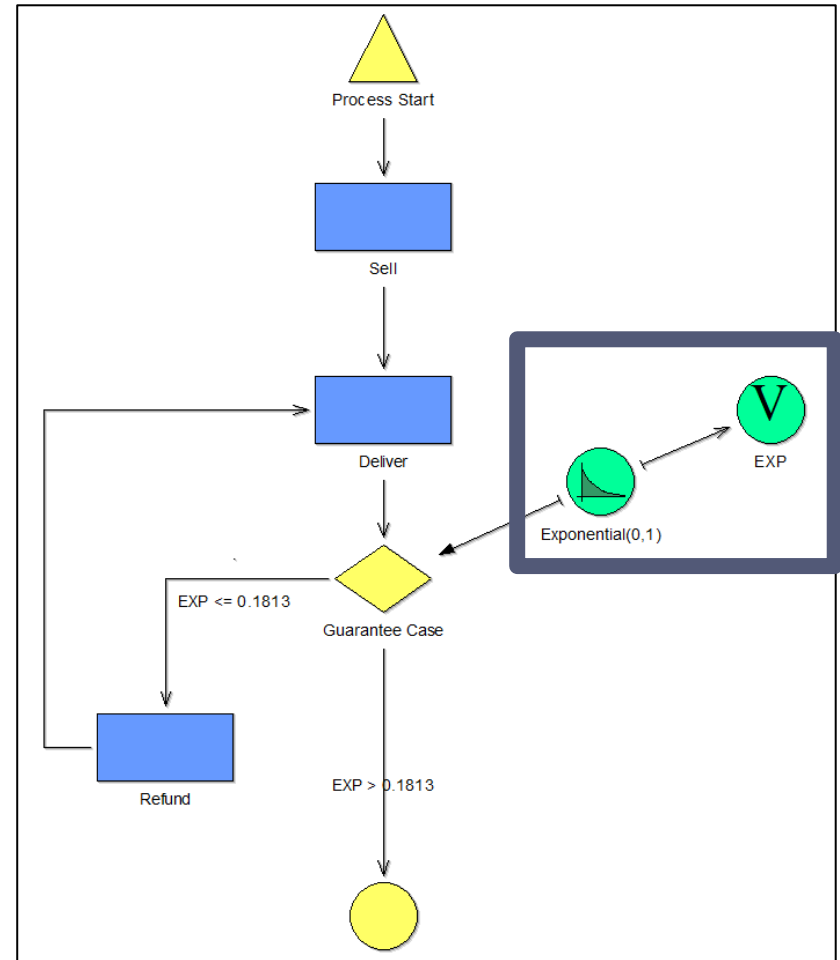
1. Overview
 - 1) Meta Model
 - 2) ADOxx
2. Details
 - 1) ADOxx Toolkit
 - 2) Modeling Language
 - 3) Core Functions
 - 4) Adoscript
 - 5) Simulation**

5) SIMULATION

Simulation

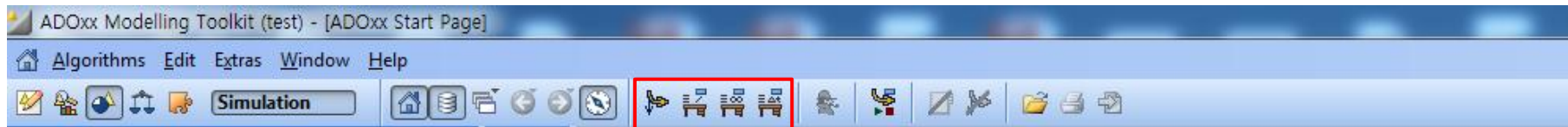
► Premise

- Have to be used in flow oriented model
 - Business process Diagram
 - Etc.



Simulation

▶ Type of simulation



▶ Path Analysis

- ▶ Simulation without working environment conditions
 - ☐ Expected values of cost and time
 - ☐ Critical Paths

▶ Capacity Analysis

- ▶ Simulation with the assignment of activities to “processors”
 - ☐ Evaluation of human requirement
 - ☐ Activity and process costs under personal cost condition

▶ Workload Analysis

- ▶ Simulation on a time axis by daily calendar and time
 - ☐ Activity and process costs under personnel cost condition
 - ☐ Capacity plan by using of process and personnel calendar

Simulation

▶ Type of simulation

▶ Path Analysis

- ▶ Input: Process time and waiting time
- ▶ Output: Weighted path results, mean values

▶ Capacity Analysis

- ▶ Input: Quantity (global/time cycle), processor assignment
- ▶ Output: Global capacity calculation, process costs

▶ Workload Analysis

- ▶ Input: Amounts per day, attendance time
- ▶ Output: Dynamically evaluated capacity curve

Simulation

▶ Matching Condition & Variable Assignment

▶ Discrete

- ▶ Variable name
- ▶ Probability



Discrete(x 0.5; y
0.5)

▶ Normal

- ▶ Expectation
- ▶ Standard deviation



Normal(50; 5)

▶ Exponential

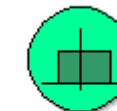
- ▶ Expectation



Exponential(0,01)

▶ Uniform

- ▶ Lower bound
- ▶ Upper bound



Uniform(1; 5)

Simulation

- ▶ Example
 - ▶ Path analysis

Path Analysis



Simulation

► Example

► Path analysis

► Numbers

- ☐ Number of simulations
- ☐ Working days per year
- ☐ Hours per working day

► Setting

- ☐ Input parameter

► Passive components

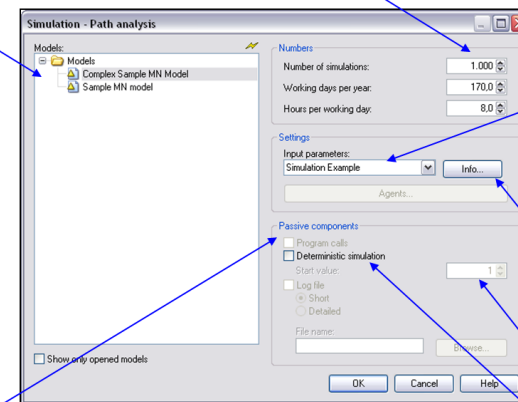
- ☐ Deterministic simulation

Select/enter the values you need for the path analysis simulation.

Select the model you want to simulate

Indicate how many processes are to be "run through". The number of simulations selected affects the accuracy - the higher the number, the more exact the simulation results will be.

Select the input parameter combination you want to work with. The input parameters are defined in the „Simmapping“ library attribute of the dynamic library in your metamodel, using the „SIMOPTION“ keyword. (see above slides of Simmapping)

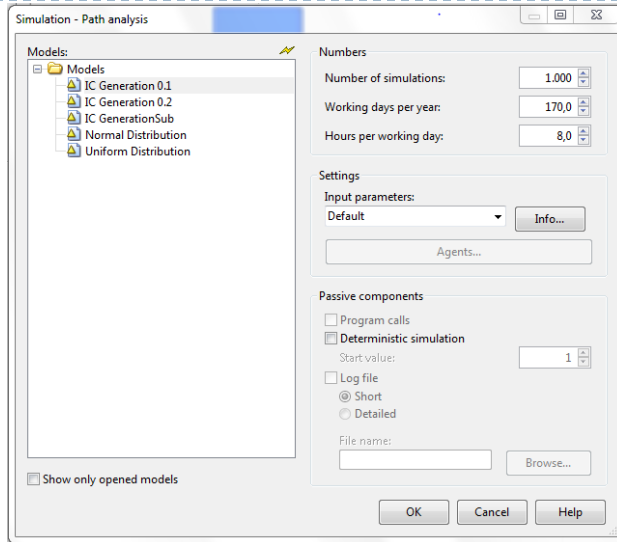


Display information about the selected input parameter combination (SIMOPTION).

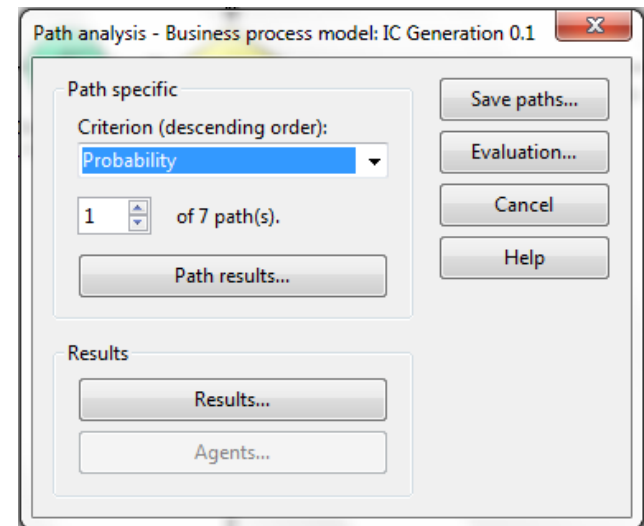
"Program calls" (default setting: deactivated)
If this option is activated, program calls specific to the activity will be carried out during the simulation of each activity. The selected input parameter combination will determine which program calls will be concerned by this .

"Deterministic simulation," (default setting: disabled)
When enabling this option the simulation is initialised with the same start value. This ensures that with the same start value independent simulation runs will determine the same simulation results.

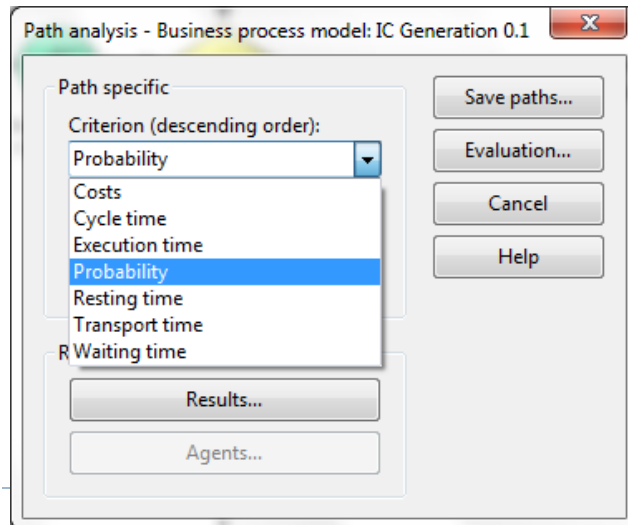
Simulation



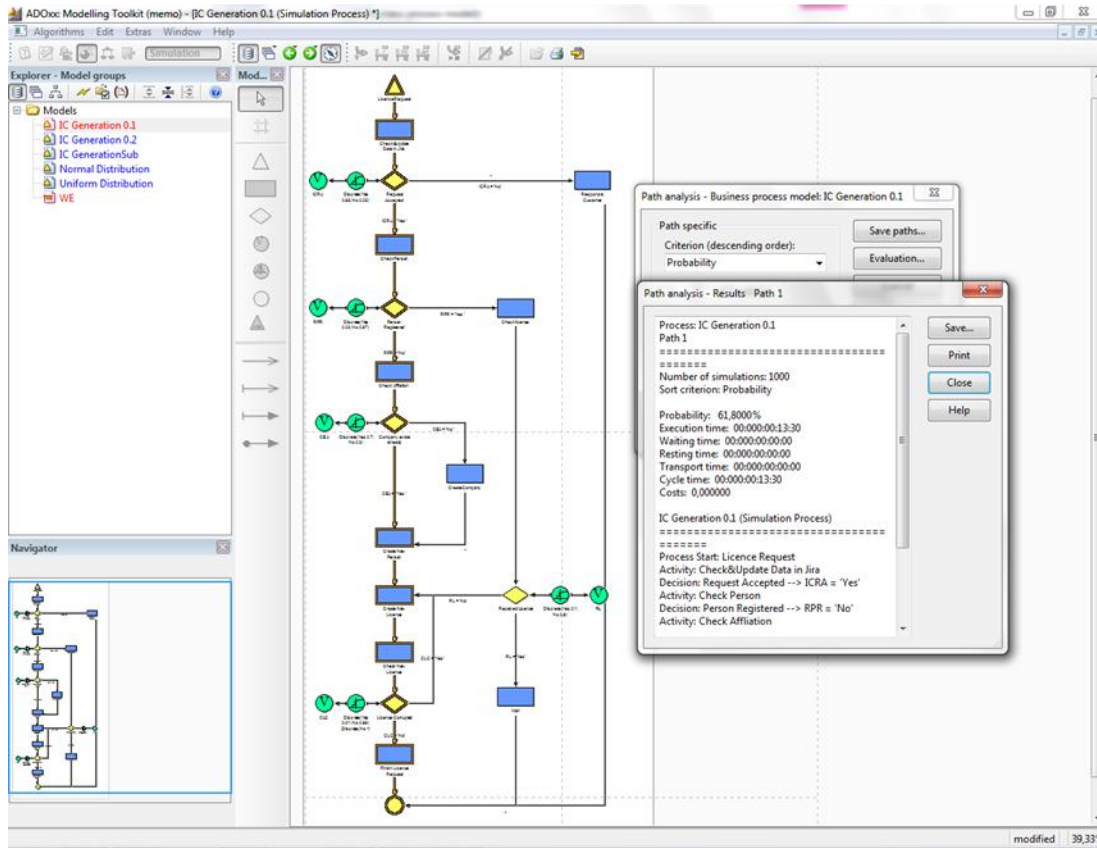
1. Select number of Simulations



2. Order outputs by criteria you want



Simulation



- Select any path you want and click "OK" to display information of it. The selected path will be marked on your model.
- The simulation results can be
 - saved and
 - printed