# CONCEPTUAL **MODEL VALUE** CREATION USING ADO*xx*

Wilfrid Utz
wilfrid.utz@boc-eu.com

http://www.adoxx.org | info@adoxx.org

2nd International Business and System Conference

**Organized by IIBA Latvia Chapter on November 5, 2013**

Co-located with PoEM 2013

# AGENDA

## 1. INTRODUCTION/MODEL VALUE

[Motivation, Positioning of Approach]

## 2. BASLINE/DEFINITIONS

[Terms, Definition, Foundation, Concepts]

## 3. IMPLEMENT MODEL STRUCTURE using ADO*xx*

[Hands-On "Hello World" Model Structure, ADOxx Concepts]

## 4. ENABLE MODEL PROCESSING using ADO*xx*

[Configure/Implement Model Processing Mechanisms/Algorithms, ADOxx Development Concepts]

## 5. APPLICATION CASES

[Case studies and additional resources for development]

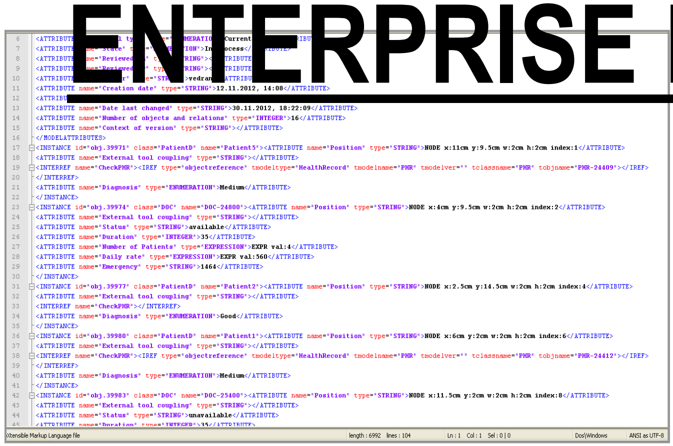
## 6. CONCLUSIONS/FURTHER READINGS

OMiLAB®

www.omilab.org

# EXAMPLE FOR MODEL VALUE:
## Enterprise Knowledge Platform as the Result of Modelling

**THE RESULTS OF MODELLING
CAN BE USED
FOR GENERATING SOFTWARE,
BUT ALSO ACT AS A BASIS OF
ENTERPRISE KNOWLEDGE PLATFORMS**



```
<ATTRIBUTE name="Creation date" type="STRING">12.11.2012, 14:08</ATTRIBUTE>
<ATTRIBUTE name="Date last changed" type="STRING">30.11.2012, 18:22:09</ATTRIBUTE>
<ATTRIBUTE name="Number of objects and relations" type="INTEGER">16</ATTRIBUTE>
<ATTRIBUTE name="Context of version" type="STRING"></ATTRIBUTE>
</MODELATTRIBUTES>
<INSTANCE id="obj.39971" class="PatientD" name="Patient5"><ATTRIBUTE name="Position" type="STRING">NODE x:11cm y:9.5cm w:2cm h:2cm index:1</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<INTERREF name="CheckPHR"><IREF type="objectreference" tmodeltype="HealthRecord" tmodelname="PHR" tmodelver="" tclassname="PHR" tobjname="PHR-24409"></IREF>
</INTERREF>
<ATTRIBUTE name="Diagnosis" type="ENUMERATION">Medium</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.39974" class="DOC" name="DOC-24800"><ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:9.5cm w:2cm h:2cm index:2</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="Status" type="STRING">available</ATTRIBUTE>
<ATTRIBUTE name="Duration" type="INTEGER">35</ATTRIBUTE>
<ATTRIBUTE name="Number of Patients" type="EXPRESSION">EXPR val:4</ATTRIBUTE>
<ATTRIBUTE name="Daily rate" type="EXPRESSION">EXPR val:560</ATTRIBUTE>
<ATTRIBUTE name="Emergency" type="STRING">1464</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.39977" class="PatientD" name="Patient2"><ATTRIBUTE name="Position" type="STRING">NODE x:2.5cm y:14.5cm w:2cm h:2cm index:4</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<INTERREF name="CheckPHR"></INTERREF>
<ATTRIBUTE name="Diagnosis" type="ENUMERATION">Good</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.39980" class="PatientD" name="Patient1"><ATTRIBUTE name="Position" type="STRING">NODE x:6cm y:2cm w:2cm h:2cm index:6</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<INTERREF name="CheckPHR"><IREF type="objectreference" tmodeltype="HealthRecord" tmodelname="PHR" tmodelver="" tclassname="PHR" tobjname="PHR-24412"></IREF>
</INTERREF>
<ATTRIBUTE name="Diagnosis" type="ENUMERATION">Medium</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.39983" class="DOC" name="DOC-25400"><ATTRIBUTE name="Position" type="STRING">NODE x:11.5cm y:2cm w:2cm h:2cm index:8</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="Status" type="STRING">unavailable</ATTRIBUTE>
```

Xtensible Markup Language file
length : 6992 lines : 104
Ln : 1 Col : 1 Sel : 0 | 0
Dos\Windows
ANSI as UTF-8

**HUMAN INTERPRETABLE**

**MACHINE PROCESSABLE**

Cf. (Karagiannis, 2012 – Presentation at FInES – “Translating Knowledge Into Growth: Views from ICT Research to Support Future Business Innovation”)

# SOME MACHINE-PROCESSABLE FORMATS …

CMMI®

DoDAF

Goals

BPSS

SCM-SCOR

Business Models

ADONIS

BPDM

BPMN

OWL-S

Balanced Scorecard

SITM framework

BPML

WS-CDL

EPC

BPEL

. . .

Business Processes

Projects

EPML

LOVEM

Prince2

IDEF

. . .

Casewise

PNML

WSFL

Reference Processes

KPIs

XPDL

Skills

Topics Maps

CRM

E-BPMS

WSCI

Knowledge Management

E-Learning

eTOM

XLANG

WSCL

Zachmann

XKMS

FEAF

GRID

XML

UML

Ontology

SAML

. . .

Internet

W-LAN

LOOM

TCP/IP

SAMM

OPM3

UML

. . .

TOGAF®

OWL

RFID

Mind Maps

LegalXML

CobiT®

EVM3

ISMS

. . .

DMAL-S

HumanML

RDF

EuroSOX

Metamodelling

GSM

Web Services

P-CMM

BASEII

GPRS

XBRL

ISO 27 002

BS 15 000

MOF

Agents

SW-CMM

SOX

Workflow

WiMax

ISO 27 001

KonotraG

Semantic Web

BS7799-2

ERP

ISO 20 000

UMTS

ITIL®

ISO 27001

WiFi

HR-XML

ebXML

LM3

Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006)

# … FROM AN EDITOR IMPLEMENTATION, TO …

CMMI®

DoDAF

Goals

SCM-SCOR

Business Models

BPSS

ADONIS

BPDM

BPMN

OWL-S

Balanced Scorecard

SITM framework

EPC

BPEL

BPML

WS-CDL

Business Processes

. . .

Projects

EPML

LOVEM

IDEF

. . .

Prince2

Casewise

PNML

WSFL

Reference Processes

KPIs

XPDL

Skills

Topics Maps

CRM

E-BPMS

WSCI

Knowledge Management

E-Learning

eTOM

XLANG

WSCL

XKMS

Zachmann

FEAF

GRID

XML

UML

. . .

SAML

Ontology

Internet

W-LAN

LOOM

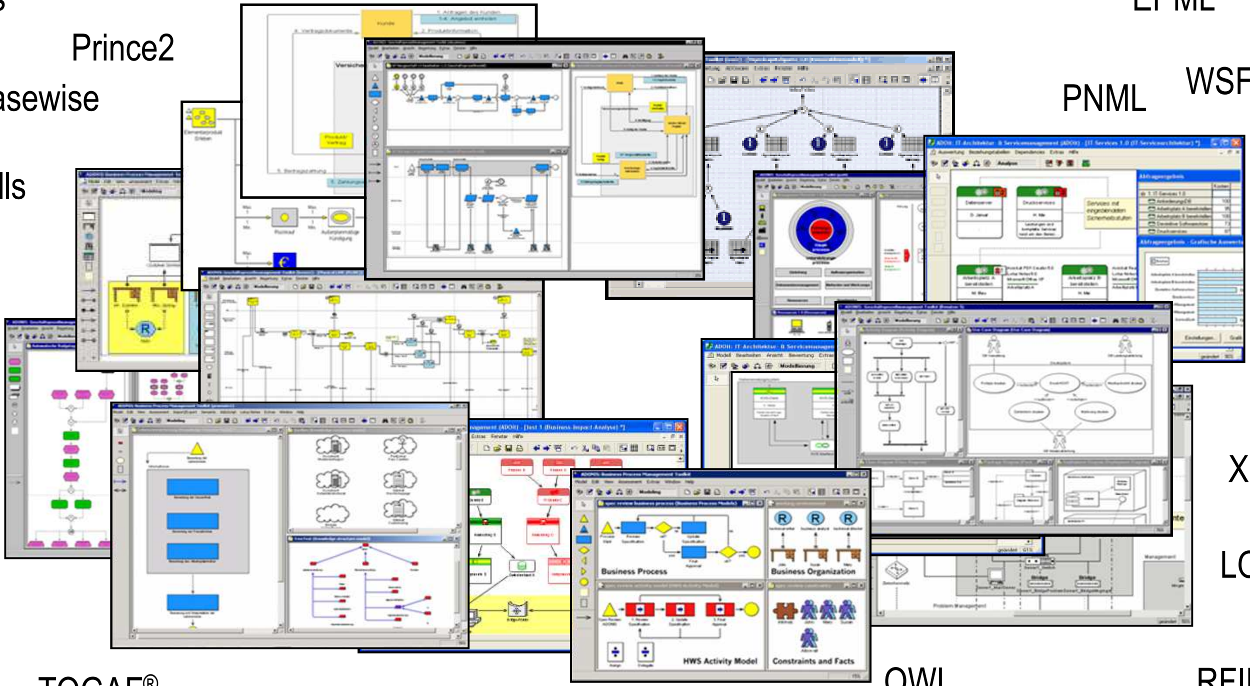
TCP/IP

SAMM

OPM3

UML

. . .



TOGAF®

CobiT®

EuroSOX

P-CMM

EVM3

ISMS

. . .

DMAL-S

RDF

Metamodelling

OWL

GSM

Mind Maps

RFID

LegalXML

HumanML

Web Services

BASEII

ISO 27 002

XBRL

GPRS

SW-CMM

BS 15 000

SOX

MOF

Workflow

Agents

WiMax

ISO 27 001

KonotraG

BS7799-2

Semantic Web

ebXML

WiFi

ERP

UMTS

ITIL®

ISO 27001

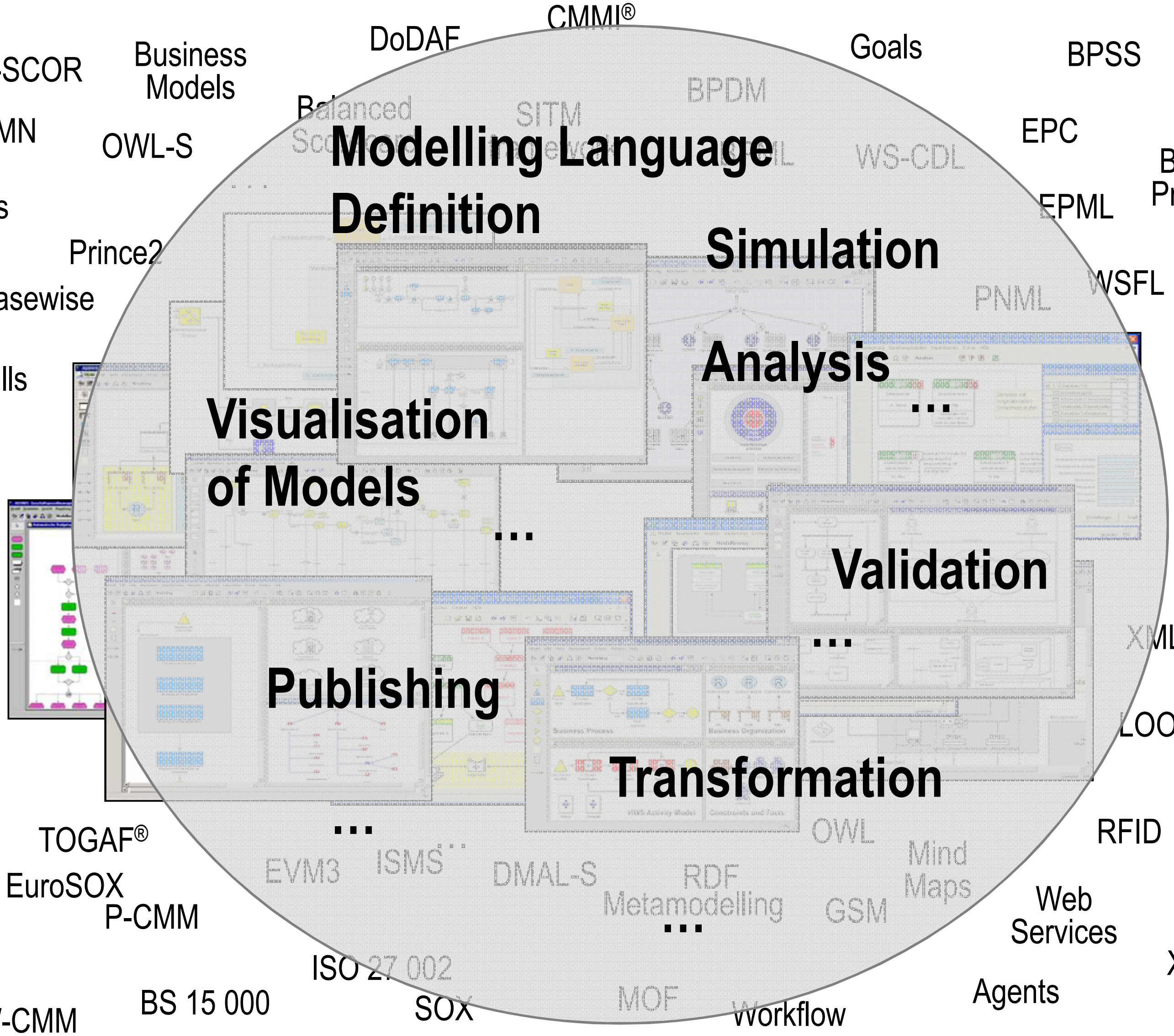
ISO 20 000

LM3

HR-XML

Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006)

# … TO FULL-FLEDGED MODELLING TOOL SUPPORT.



**Modelling Language Definition**

**Simulation**

**Analysis**

**…**

**Visualisation of Models**

**…**

**Validation**

**…**

**Publishing**

**Transformation**

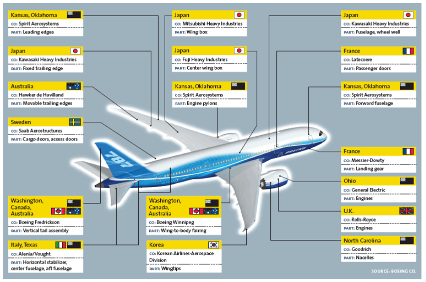
**…**

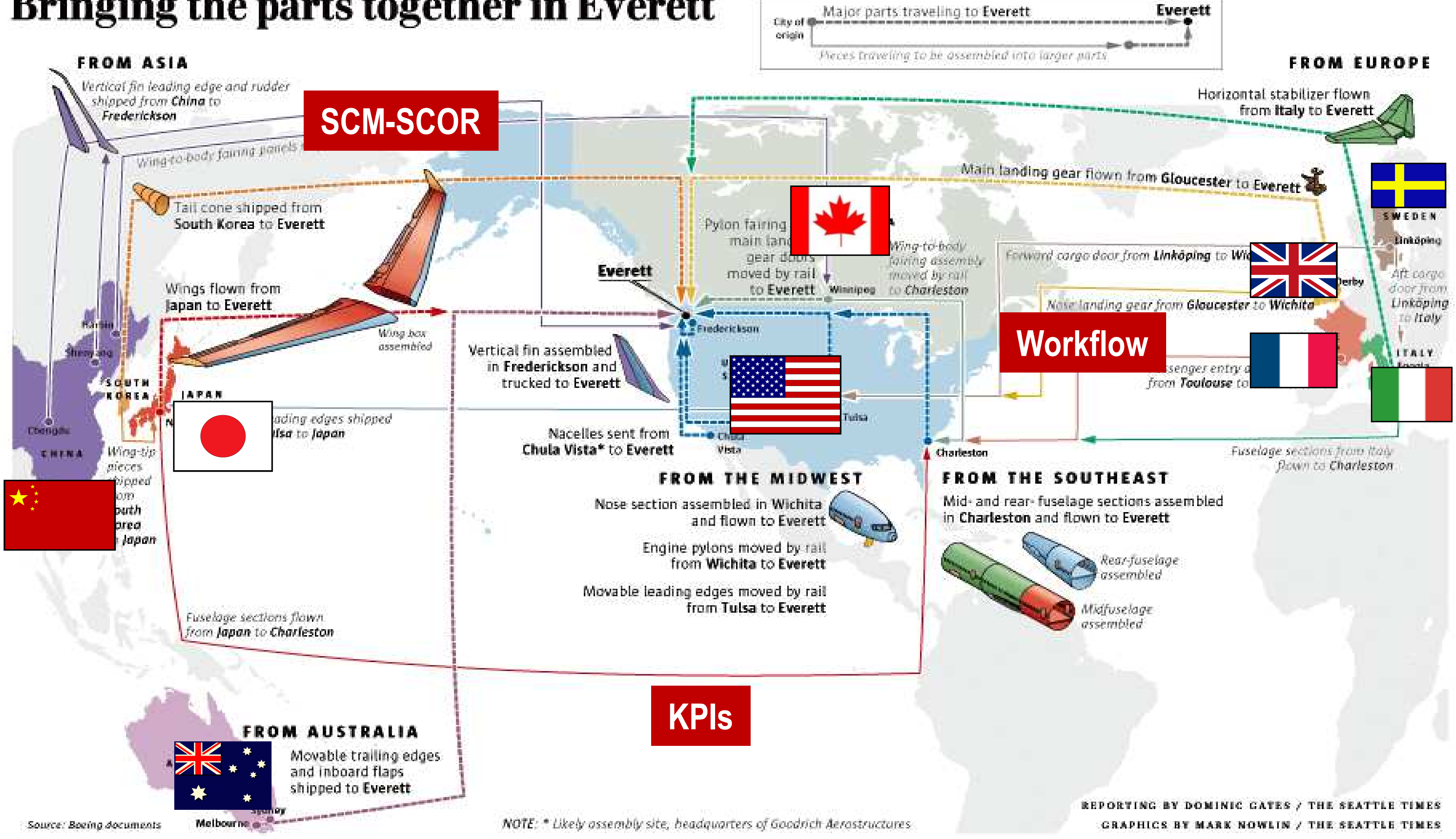
**…**

SCM-SCOR
Business Models
DoDAF
CMMI®
Goals
BPSS
ADONIS
BPMN
OWL-S
Balanced Scorecard
SITM
BPDM
WS-CDL
EPC
BPEL
Projects
Business Processes
EPML
LOVEM
Prince2
IDEF
. . .
Casewise
PNML
WSFL
Reference Processes
KPIs
XPDL
Topics Maps
Skills
CRM
E-BPMS
WSCI
Knowledge Management
E-Learning
eTOM
XLANG
WSCL
XKMS
Zachmann
FEAF
GRID
XML
UML
SAML
Ontology
. . .
Internet
W-LAN
LOOM
TCP/IP
OPM3
SAMM
UML
. . .
RFID
TOGAF®
LegalXML
CobiT®
EVM3
ISMS
DMAL-S
OWL
Mind Maps
HumanML
EuroSOX
RDF
Metamodelling
GSM
Web Services
P-CMM
XBRL
GPRS
BASEII
ISO 27 002
MOF
SOX
Workflow
Agents
SW-CMM
BS 15 000
WiMax
ISO 27 001
KonotraG
Semantic Web
ERP
UMTS
BS7799-2
ITIL®
ISO 27001
ISO 20 000
ebXML
WiFi
HR-XML
LM3

Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006)

# SCENARIO: INTEGRATED SUPPLY CHAIN SIMULATIO




Bringing the parts together in Everett
City of origin
Major parts traveling to Everett
Everett
Pieces traveling to be assembled into larger parts
FROM ASIA
Vertical fin leading edge and rudder shipped from China to Frederickson
Wing-to-body fairing panels
Tail cone shipped from South Korea to Everett
Wings flown from Japan to Everett
Wing box assembled
Harbin
Shenyang
SOUTH KOREA
JAPAN
Chengdu
CHINA
Wing-tip pieces shipped from South Korea to Japan
ading edges shipped lsa to Japan
Fuselage sections flown from Japan to Charleston
Everett
Vertical fin assembled in Frederickson and trucked to Everett
Nacelles sent from Chula Vista* to Everett
Pylon fairing main landing gear doors moved by rail to Everett
Winnipeg
Wing-to-body fairing assembly moved by rail to Charleston
Frederickson
Chula Vista
Tulsa
Charleston
FROM THE MIDWEST
Nose section assembled in Wichita and flown to Everett
Engine pylons moved by rail from Wichita to Everett
Movable leading edges moved by rail from Tulsa to Everett
FROM THE SOUTHEAST
Mid- and rear- fuselage sections assembled in Charleston and flown to Everett
Rear-fuselage assembled
Midfuselage assembled
FROM EUROPE
Horizontal stabilizer flown from Italy to Everett
Main landing gear flown from Gloucester to Everett
SWEDEN
Linköping
Forward cargo door from Linköping to Wi
Derby
Nose landing gear from Gloucester to Wichita
Aft cargo door from Linköping to Italy
ITALY
Foggia
ssenger entry d from Toulouse to
Fuselage sections from Italy flown to Charleston
FROM AUSTRALIA
Movable trailing edges and inboard flaps shipped to Everett
Sydney
Melbourne
Source: Boeing documents
NOTE: * Likely assembly site, headquarters of Goodrich Aerostructures
REPORTING BY DOMINIC GATES / THE SEATTLE TIMES
GRAPHICS BY MARK NOWLIN / THE SEATTLE TIMES

**SCM-SCOR**

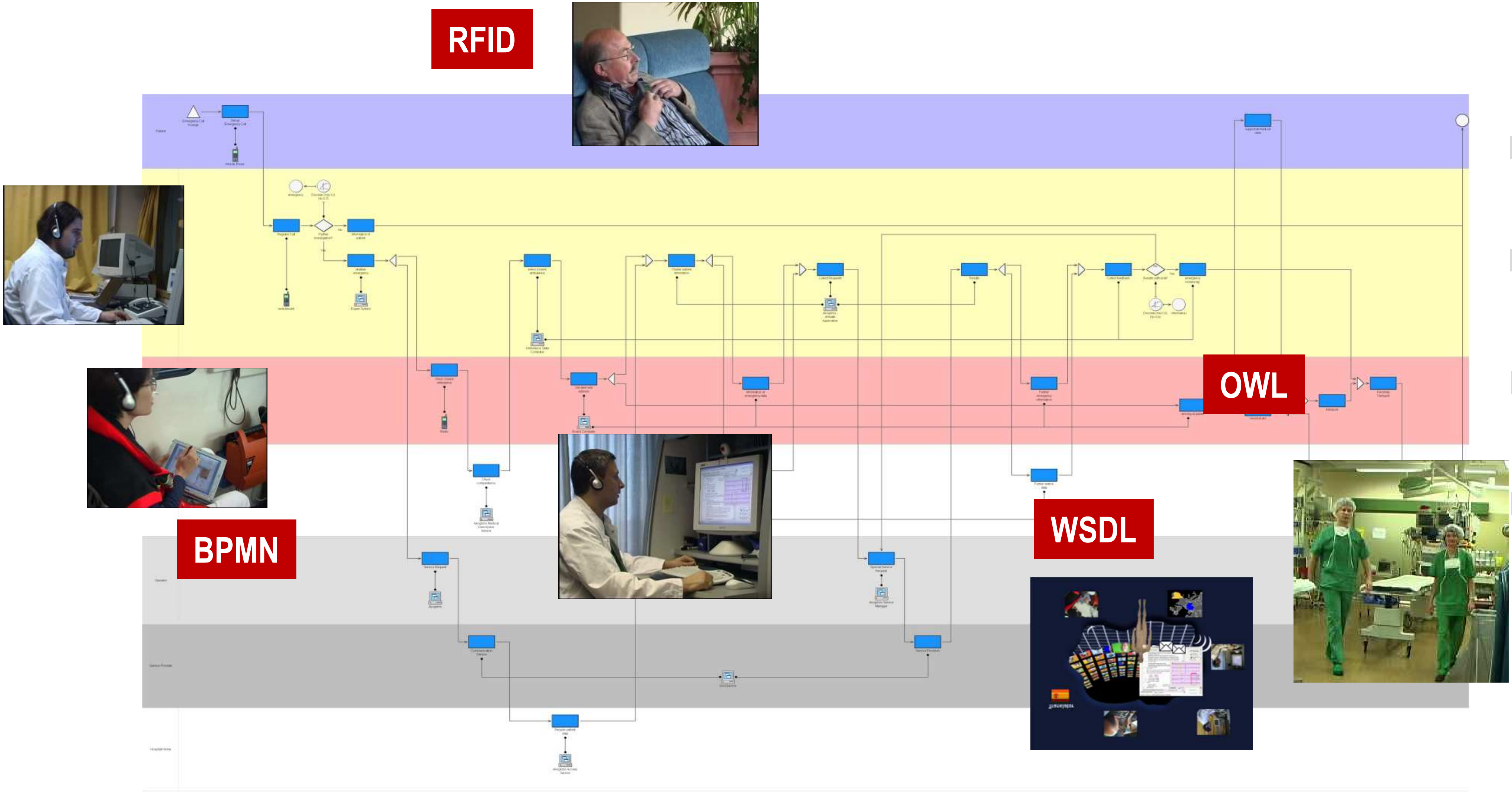
**Workflow**

**KPIs**

787 Dreamliner Manufacturing Reference: Seattle Times "Boeing 787: Build globally, assemble locally" Originally published September 11, 2005
URL: http://seattletimes.com/html/businesstechnology/2002484189_787globalintro11.html

# SCENARIO: MOBILE EHEALTH ANALYSIS AND SIMULATION

## AKOGRIMO Project

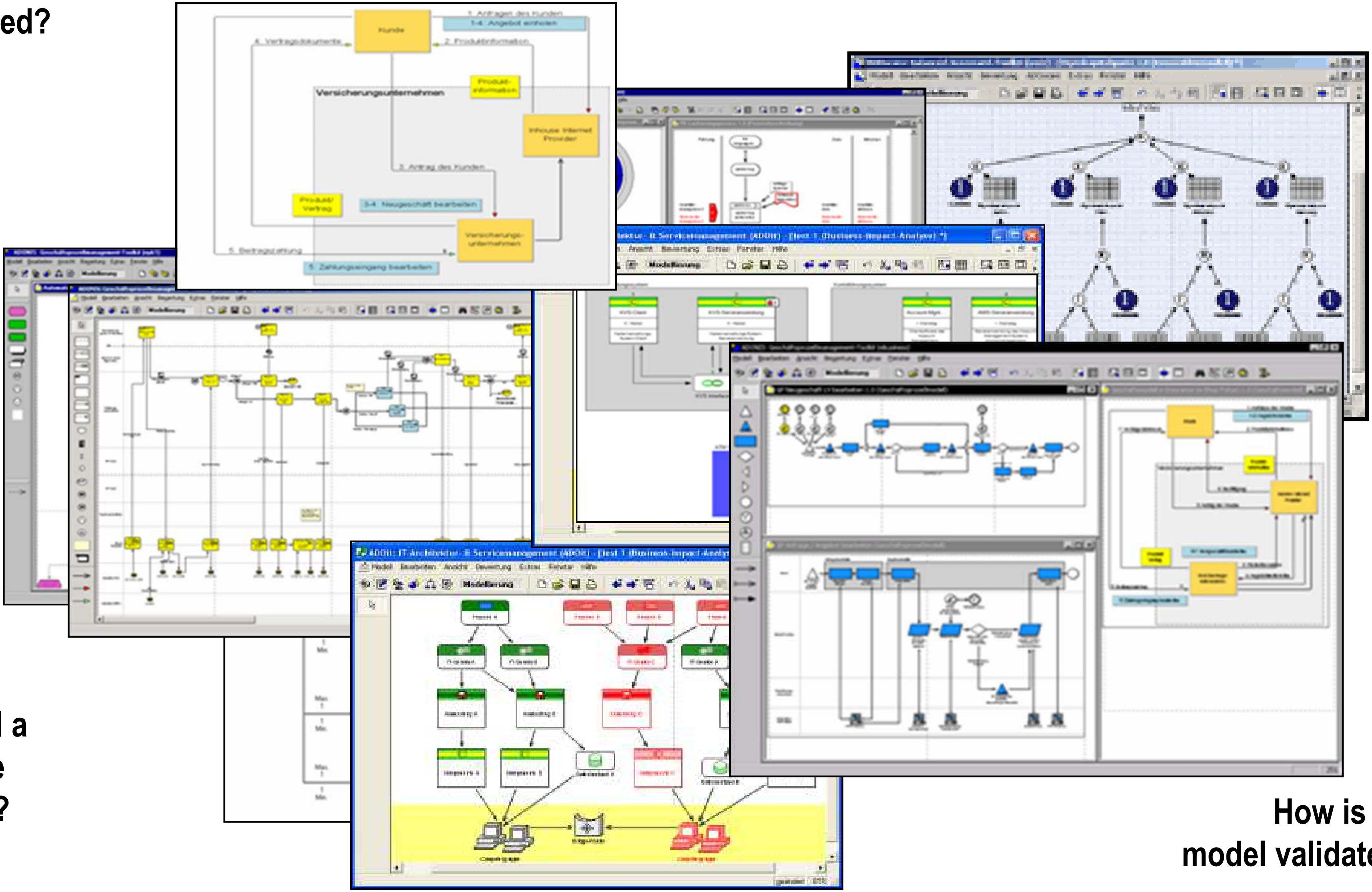


E-health scenario, for more information see video on http://www.mobilegrid.org

# THE NECESSARY INFORMATION FOR MODEL PROCESSING

What data
is contained?

Which algorithms should be
applied?



What will a
model be
used for?

How is the
model validated?

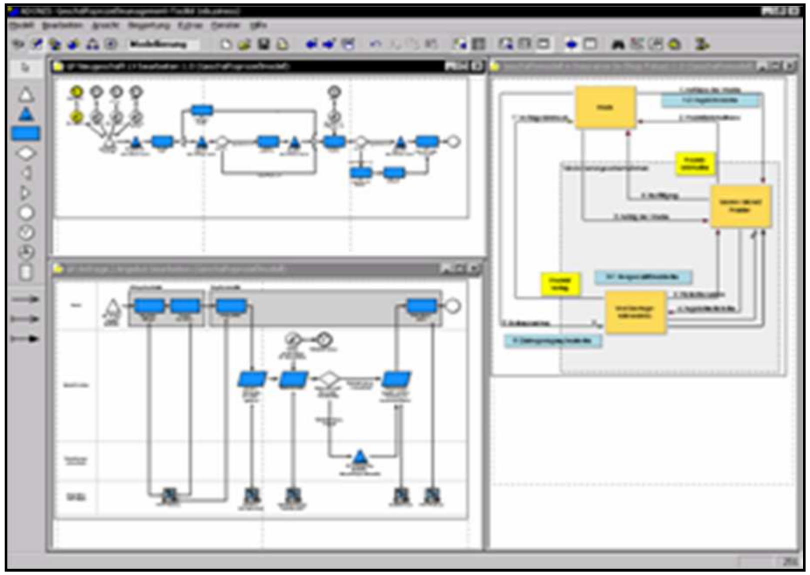
# SOME FUNCTIONALITIES OF MODELLING TOOLS

Visualisation of models

User interaction like: drag and drop, zoom, grid snap, print, etc.

Simulation of models

Modelling language definition



Publishing in multiple formats

Transformation of models

Exchange of models

Analyse models and evaluate the results

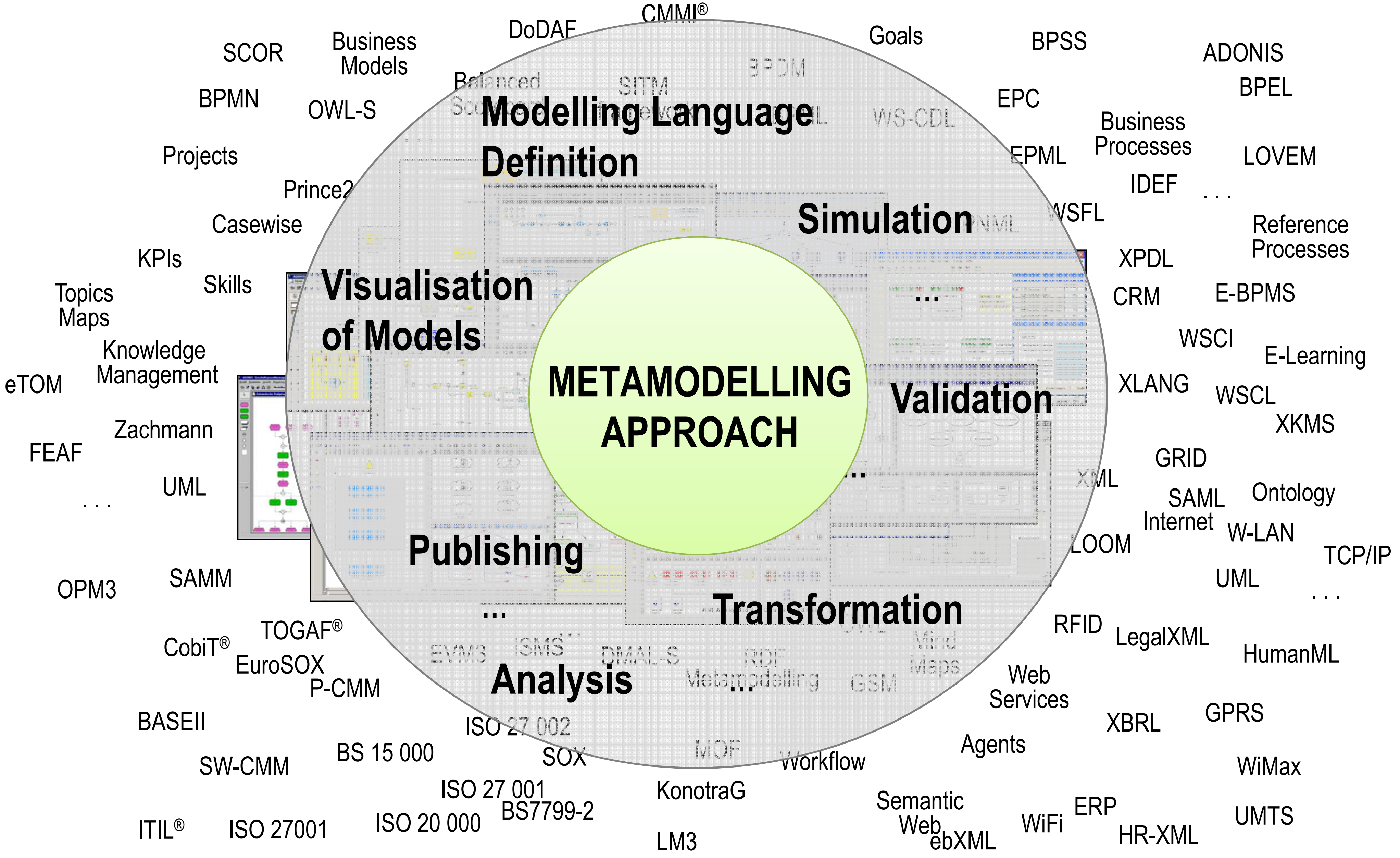
User access rights

Storage and Manipulation of Models

Security and Safety

Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006; Fill, 2009)

# A METAMODEL-BASED REALISATION APPROACH



Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006)

# SIXTEEN REASONS OTHER THAN PREDICTION TO BUILD MODELS

1. Explain (very distinct from predict)
2. Guide data collection
3. Illuminate core dynamics
4. Suggest dynamical analogies
5. Discover new questions
6. Promote a scientific habit of mind
7. Bound (bracket) outcomes to plausible ranges
8. Illuminate core uncertainties.
9. Offer crisis options in near-real time
10. Demonstrate tradeoffs / suggest efficiencies
11. Challenge the robustness of prevailing theory through perturbations
12. Expose prevailing wisdom as incompatible with available data
13. Train practitioners
14. Discipline the policy dialogue
15. Educate the general public
16. Reveal the apparently simple (complex) to be complex (simple)

**Joshua M. Epstein (2008)**

# AGENDA

1. INTRODUCTION/MODEL VALUE

[Motivation, Positioning of Approach]

2. BASLINE/DEFINITIONS

[Terms, Definition, Foundation, Concepts]

3. IMPLEMENT MODEL STRUCTURE using ADO*xx*

[Hands-On "Hello World" Model Structure, ADOxx Concepts]

4. ENABLE MODEL PROCESSING using ADO*xx*

[Configure/Implement Model Processing Mechanisms/Algorithms, ADOxx Development Concepts]

5. APPLICATION CASES

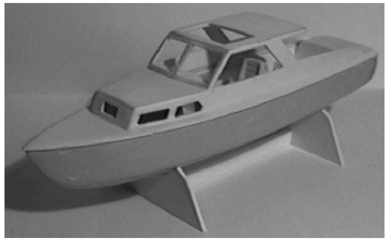
[Case studies and additional resources for development]

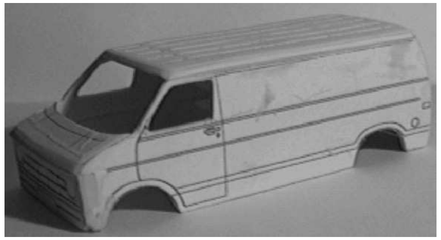
6. CONCLUSIONS/FURTHER READINGS

www.omilab.org

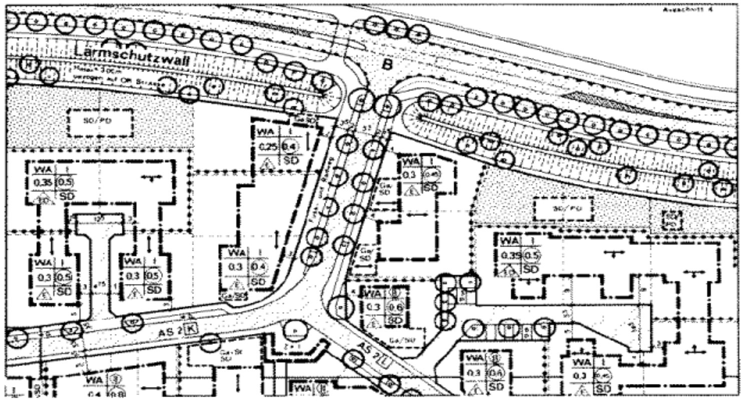
# MODEL DEFINITIONS

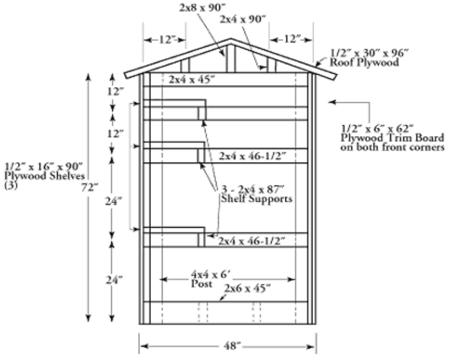
- **Model as mappings of reality**
  *…models as mappings of parts of reality for a particular purpose…*





- **Model as a construction**
  *...the result of a construction of a modeler who declares for model users a representation of an original as significant at a given time using a language…*





(Source: Schütte and Becker, 1998)

# MODEL WITH DIFFERENT VALUES

Representation Characteristic

*“Models as a representation of natural or artifical originals, that again can be models.“ [1] (translated)*

Abstraction Characteristic

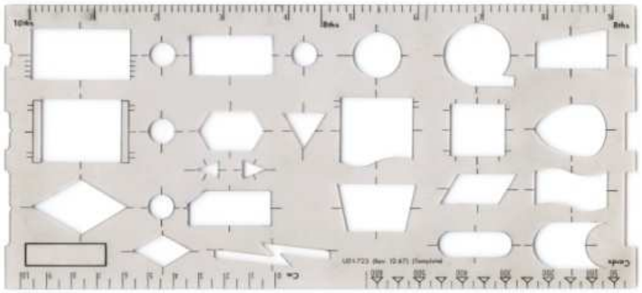
*“Models in general do not capture all attributes of the represented original, but only those that seem relevant to the modeller or model user.“[1] (translated)*

Pragmatic Characteristic

*Models meet their substituion function for specific subjects, within a pre-determined time interval and with limitations on defined intellectual and/or real operations. [1] (translated)*

Source: Stachowiak 1973

# TERMS/DEFINITIONS USED



- **Modelling Language:**
  Modelling constructs (object types) and their relations (relation types) to each other to declare a model.

- **Metamodel:**
  The model of the syntax of the modelling language

- **Meta$^2$ Model:**
  Model of abstract syntax of a language to describe meta models.

- **Modelling Technique:**
  A modelling language and proceeding instructions for creation of a model in this modelling language.

- **Mechanisms und Algorithms:**
  Provision of functionalities to process models such as manipulation, visualisation, query, transformation or simulation depending on the modelling language and modelling procedure.

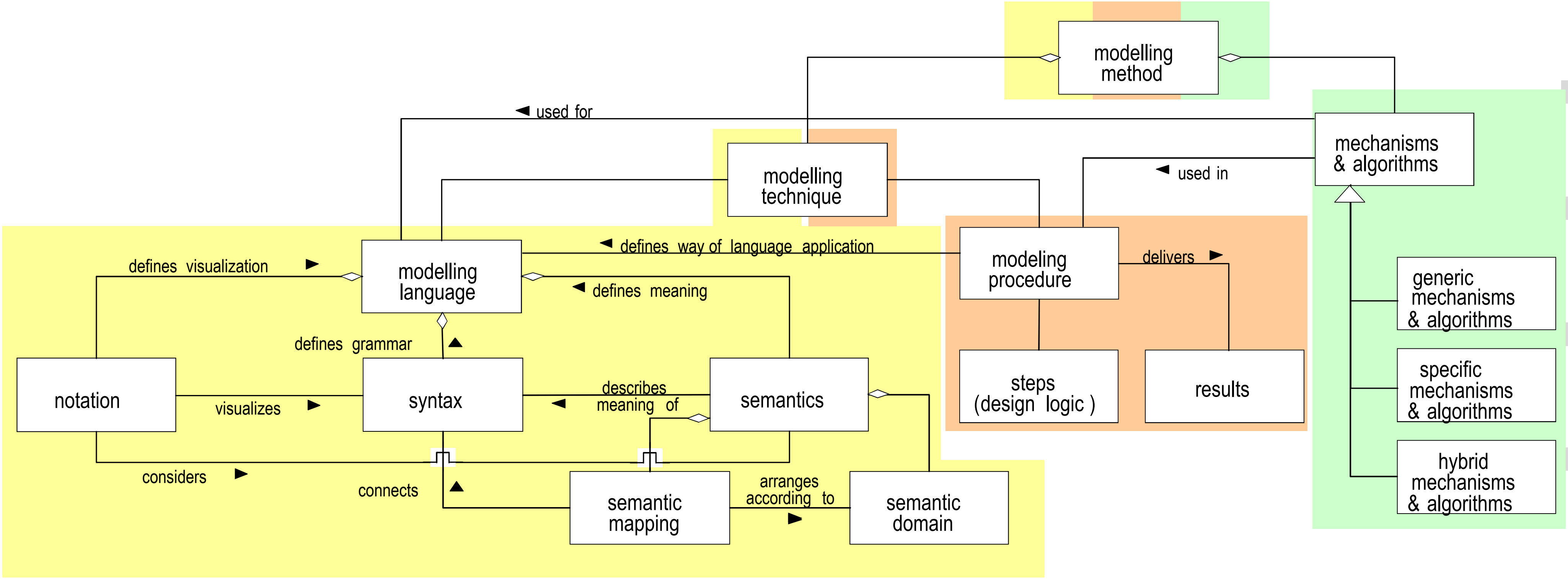
Cf. (Karagiannis and Kühn, 2002; Karagiannis and Höfferer, 2006; Kühn 2004; Karagiannis and Visic, 2011)

Pictures [SXC]

# META MODELS APPLICATION SCENARIOS

| Application Scenario of Models |
|---|
| Enterprise Modelling |
| Business Rules and eContracts |
| Decision Support and Case handling |
| Business Process & Workflow Management |
| Information Systems Architecture and System Configuration |
| Requirements Engineering |
| Knowledge  Representation |
| Data Processing, Data Management and Data Integration |

# GENERIC CONCEPT MODEL SPECIFICATION FRAMEWORK



Reference: Karagiannis, D., Kühn, H.: „Metamodelling Platforms“. In Bauknecht, K., Min Tjoa, A., Quirchmayer, 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 ff.

# AGENDA

## 1. INTRODUCTION/MODEL VALUE

[Motivation, Positioning of Approach]

## 2. BASLINE/DEFINITIONS

[Terms, Definition, Foundation, Concepts]

## 3. IMPLEMENT MODEL STRUCTURE using ADO*xx*

[Hands-On "Hello World" Model Structure, ADOxx Concepts]

## 4. ENABLE MODEL PROCESSING using ADO*xx*

[Configure/Implement Model Processing Mechanisms/Algorithms, ADOxx Development Concepts]

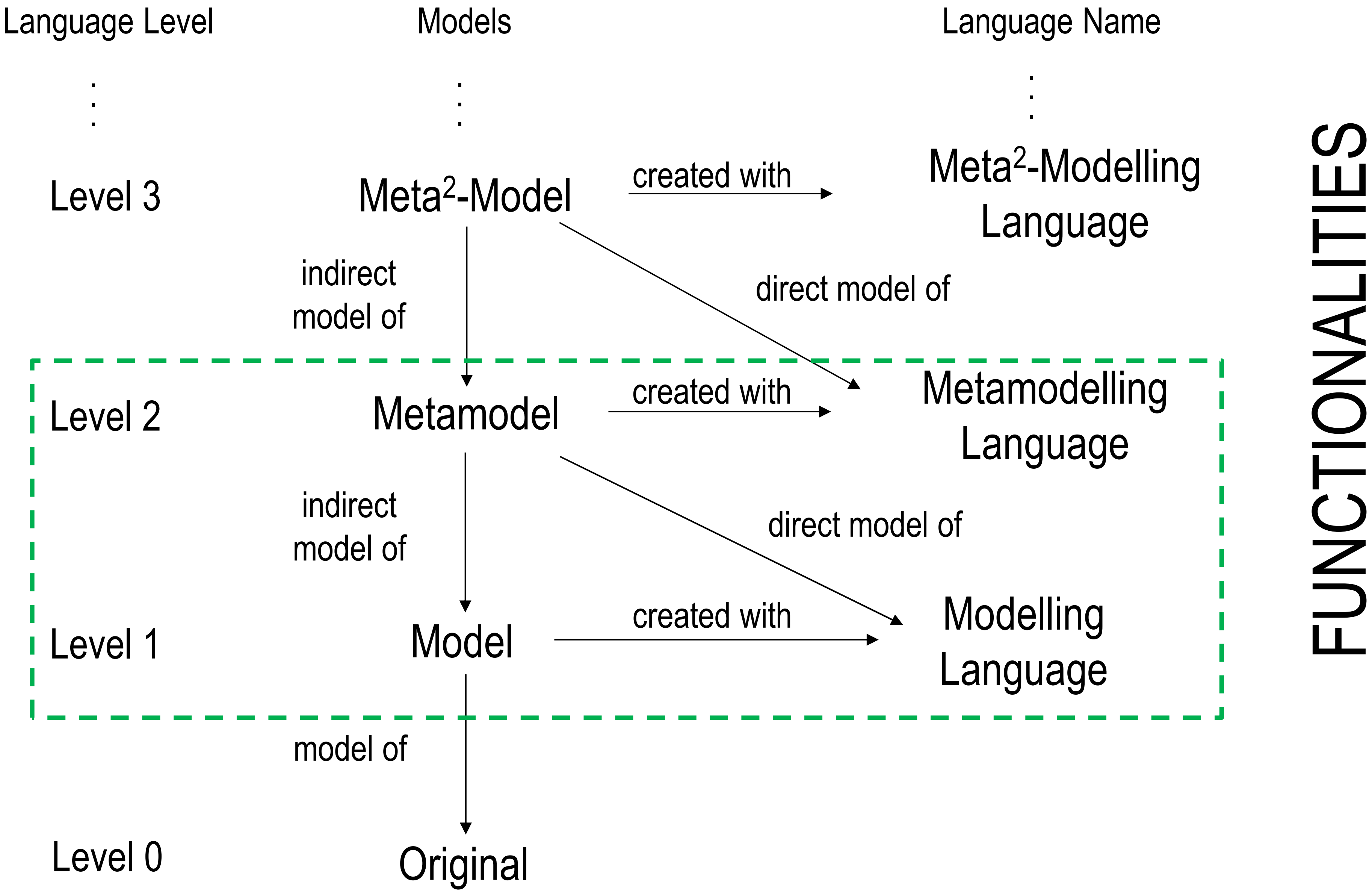
## 5. APPLICATION CASES

[Case studies and additional resources for development]

## 6. CONCLUSIONS/FURTHER READINGS

OMiLAB®
www.omilab.org

# DEFINITION OF MODEL STRUCTURE AND FUNCTIONALITIES



Language Level

Models

Language Name

Level 3

Meta$^2$-Model

created with

Meta$^2$-Modelling Language

indirect model of

direct model of

Level 2

Metamodel

created with

Metamodelling Language

indirect model of

direct model of

Level 1

Model

created with

Modelling Language

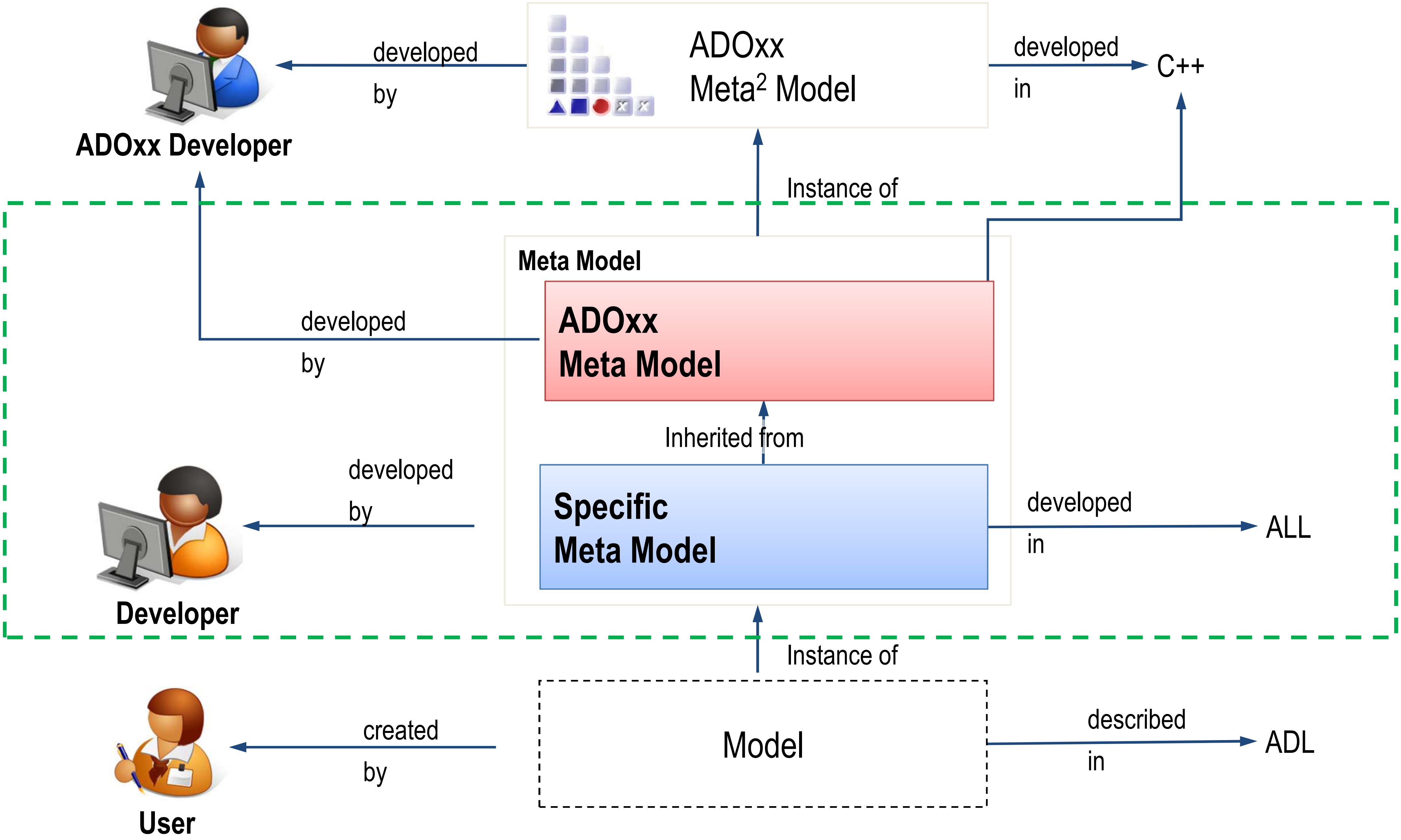
model of

Level 0

Original

FUNCTIONALITIES

# ADO*xx* PLATFORMS HIERARCHY



ADOxx Meta² Model

developed by

**ADOxx Developer**

developed in

C++

Instance of

**Meta Model**

**ADOxx Meta Model**

developed by

Inherited from

**Specific Meta Model**

developed by

**Developer**

developed in

ALL

Instance of

Model

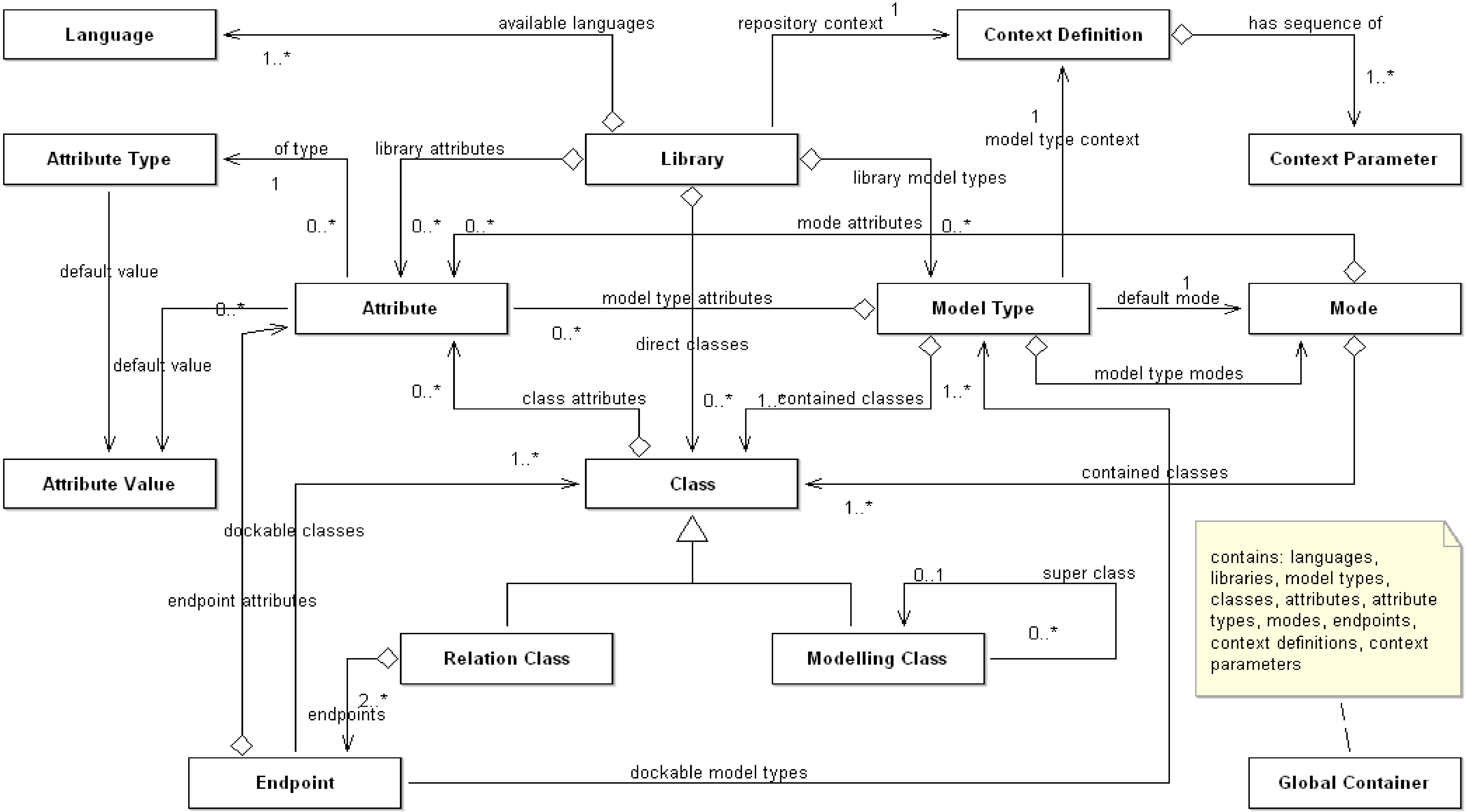
created by

**User**

described in

ADL

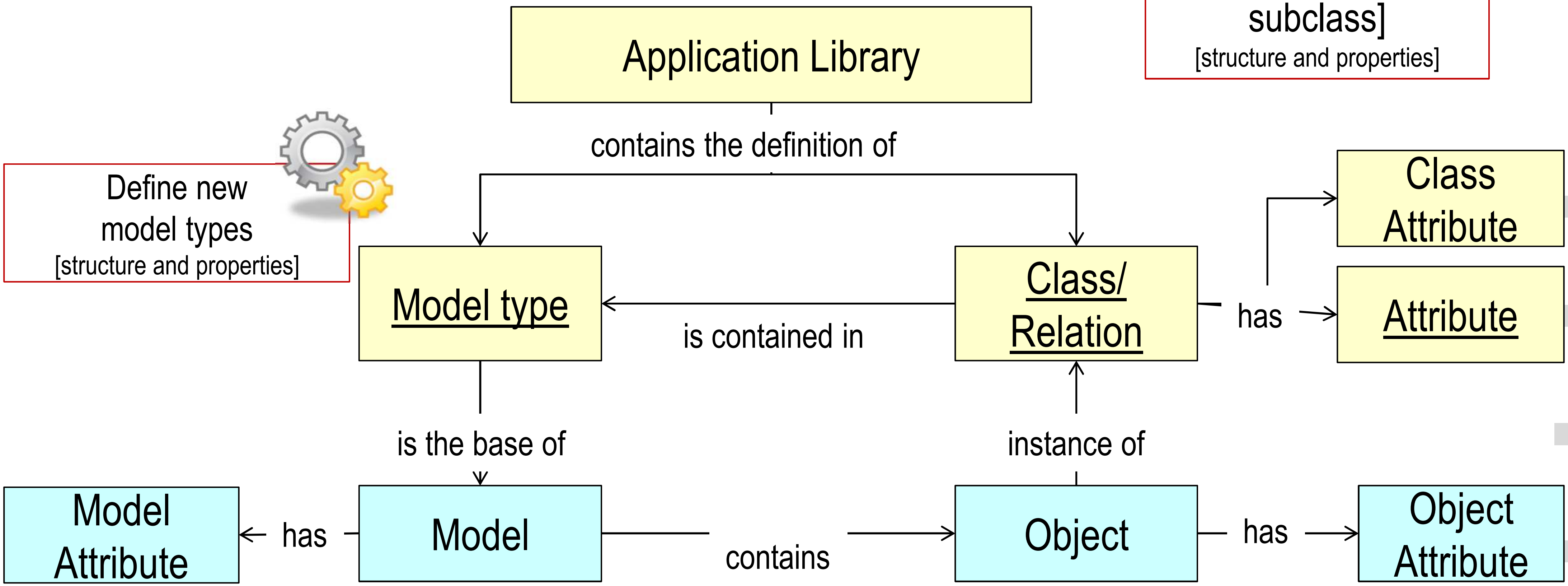
# ADO*xx* META²-MODEL



contains: languages, libraries, model types, classes, attributes, attribute types, modes, endpoints, context definitions, context parameters

# THE ADOXX LIBRARY CONCEPT: APPLICATION LIBRARY

Define new classes
[abstract | concrete | subclass]
[structure and properties]



Application Library

contains the definition of

Define new model types
[structure and properties]

Model type

is contained in

Class/ Relation

has

Class Attribute

Attribute

is the base of

instance of

Model Attribute

has

Model

contains

Object

has

Object Attribute

**Model Types:** A model type is a well-defined sub collection of classes and relation classes of a meta model.

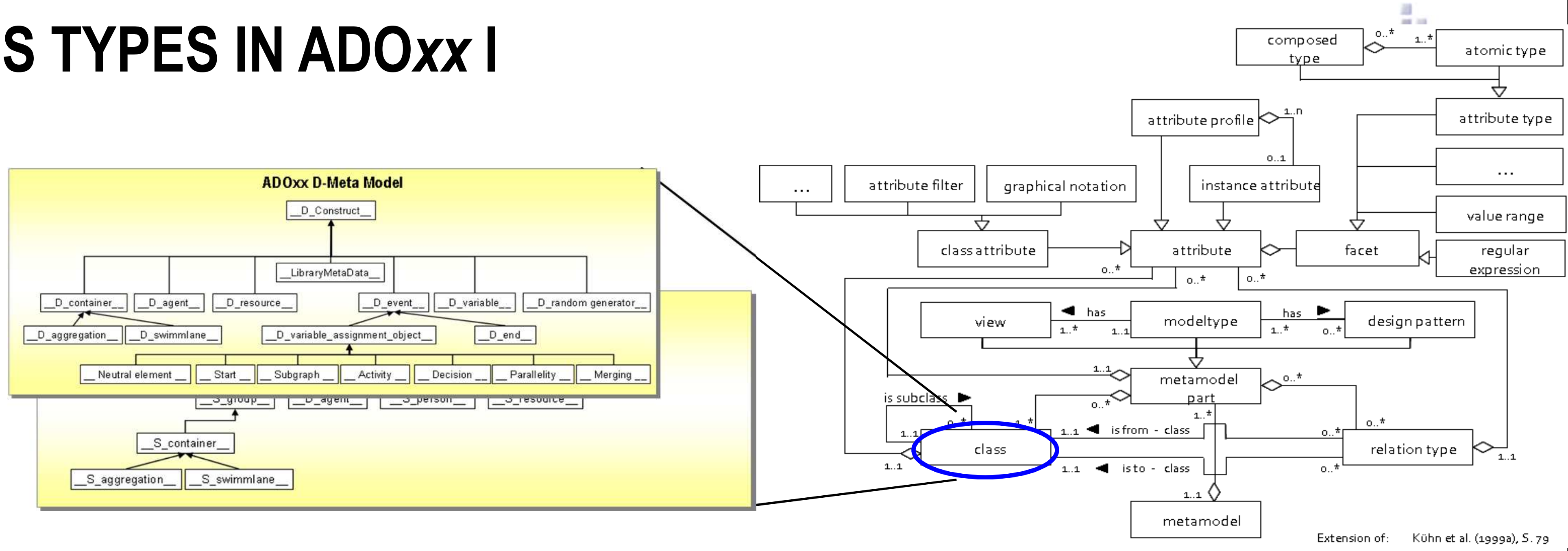
**Classes:** A class is a construct that is used as a template to create objects of that class. The objects of a class are alternatively called "instances"

**Attributes:** An attribute is a property of a modelling construct such as a model, object or relation. Each attribute has a type and a value.

**Relations:** A relation class is a construct that is used as a template to create relations between objects. A relation class is defined between classes. A relation is always a directed connection between objects, i.e. each relation has a from-side and a to-side.

Cf. (Junginger et al., 2000; Kühn, 2004; Fill, Redmond, Karagiannis, 2012)

# CLASS TYPES IN ADO*xx* I



Extension of: Kühn et al. (1999a), S. 79

- **Pre-defined Abstract Classes (ADOxx meta model class)**
  - Pre-defined abstract classes are classes that are provided by ADOxx with a given semantic and basic syntax in form of attributes. They can be used to inherit the pre-defined syntax and the attributes to either self-defined abstract classes or to classes.
  - ADOxx functionality that is provided for the pre-defined abstract classes can be used for any inherited concrete class. Hence pre-defined and provided ADOxx functionality is consumed due to inheritance of such pre-defined abstract classes.
  - Pre-defined abstract classes are the ADOxx meta model, hence they exist in every meta model based on ADOxx.
  - Nomenclature: __ Class Name __

# CLASS TYPES IN ADO*xx* II

Define new classes
[abstract | concrete |
subclass]
[structure and properties]

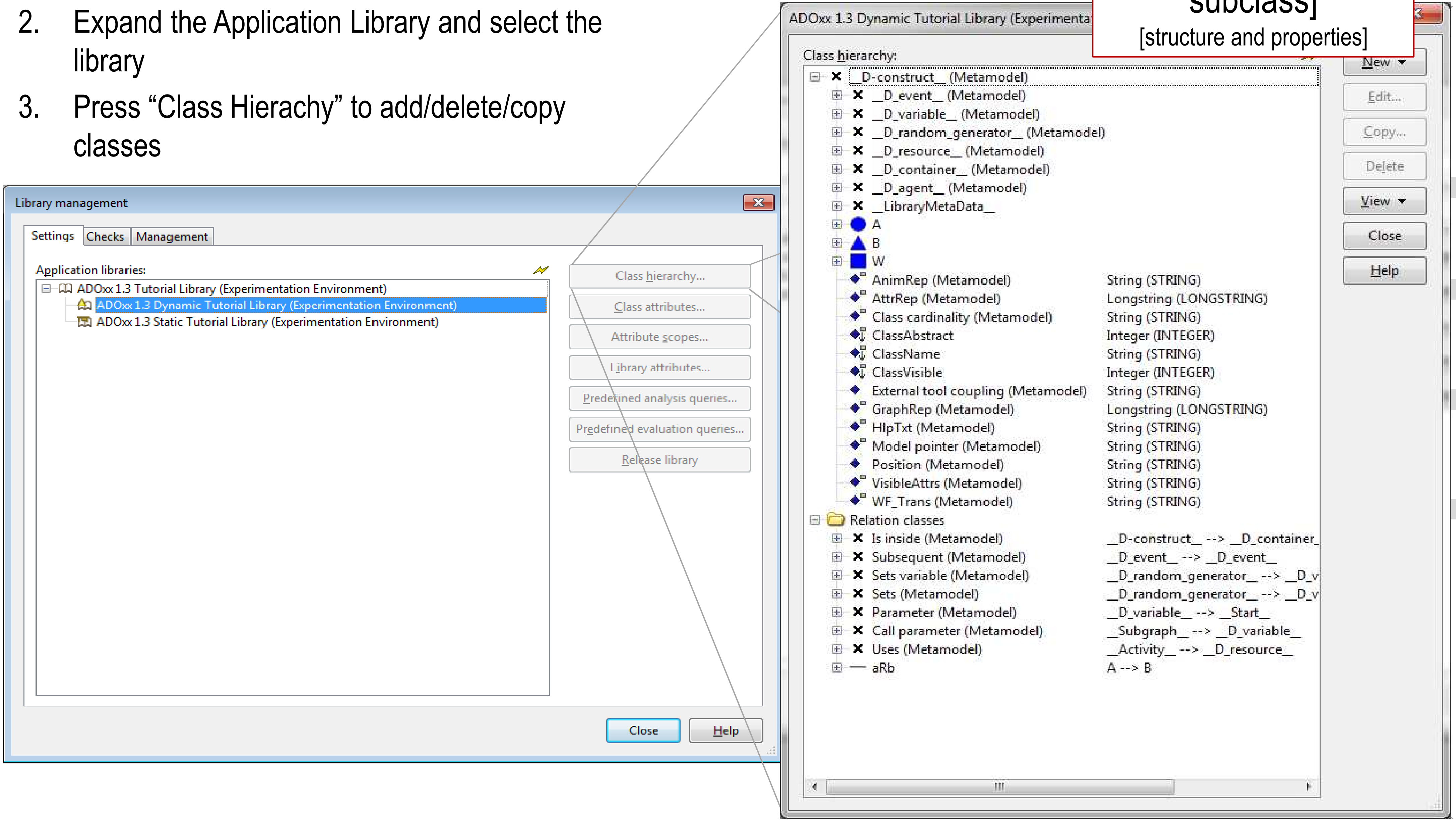
- **Abstract Classes**
  - Abstract classes are self-defined classes enabling to structure the meta model and define syntax in form of attributes and semantic, which is inherited by sub-classes.
  - Abstract classes either inherit from the root class of the meta model, or from any other class of the meta model. Hence, they inherit the behaviour from their super-class – which is often a pre-defined abstract class from the ADOxx meta model.
  - Abstract classes enable an efficient meta model, hence they may not be in every ADOxx meta model.
  - Nomenclature: _ Class Name _
- **(Concrete) Classes**
  - Classes are self-defined classes defining a concrete modelling class that can be used, when applying the corresponding modelling language. Hence all model objects in every model created on ADOxx is an instance of a class.
  - Classes inherit the semantic and the attributes from the Pre-defined abstract class and additionally - in case of inheriting - from the abstract class.
  - Classes enable the realisation of a concrete meta model.
  - Nomenclature: Class Name

Cf. (Fill, Redmond, Karagiannis, 2012)

# DEMONSTRATION: CLASS DEFINITION I

Define new classes [abstract | concrete | subclass]
[structure and properties]

1. Open the Library Management Component
2. Expand the Application Library and select the library
3. Press “Class Hierachy” to add/delete/copy classes



Application libraries:
- ADOxx 1.3 Tutorial Library (Experimentation Environment)
  - ADOxx 1.3 Dynamic Tutorial Library (Experimentation Environment)
  - ADOxx 1.3 Static Tutorial Library (Experimentation Environment)

Class hierarchy:
- __D-construct__ (Metamodel)
  - __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) | 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) |
| VisibleAttrs (Metamodel) | String (STRING) |
| WF_Trans (Metamodel) | String (STRING) |

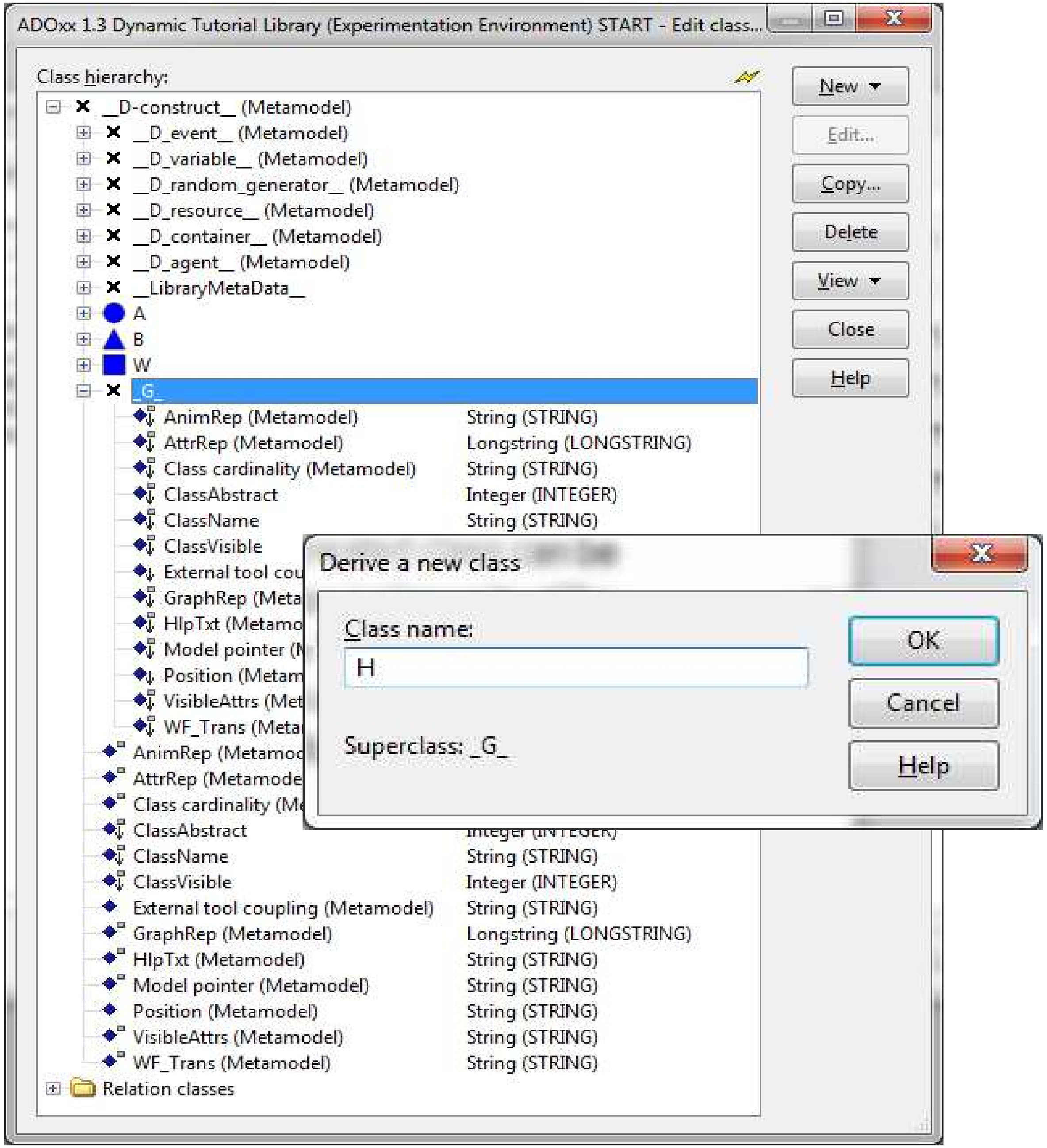
Relation classes

| | |
|---|---|
| Is inside (Metamodel) | __D-construct__ --> __D_container_ |
| Subsequent (Metamodel) | __D_event__ --> __D_event__ |
| Sets variable (Metamodel) | __D_random_generator__ --> __D_v |
| Sets (Metamodel) | __D_random_generator__ --> __D_v |
| Parameter (Metamodel) | __D_variable__ --> __Start__ |
| Call parameter (Metamodel) | __Subgraph__ --> __D_variable__ |
| Uses (Metamodel) | __Activity__ --> __D_resource__ |
| aRb | A --> B |

# DEMONSTRATION: CLASS DEFINITION II

Define new classes [abstract | concrete | subclass]

[structure and properties]


ADOxx 1.3 Dynamic Tutorial Library (Experimentation Environment) START - Edit class...
New
Edit...
Copy...
Delete
View
Close
Help
Derive a new class
OK
Cancel
Help

Class hierarchy:

| | |
|---|---|
| _D-construct_ (Metamodel) | |
| _D_event_ (Metamodel) | |
| _D_variable_ (Metamodel) | |
| _D_random_generator_ (Metamodel) | |
| _D_resource_ (Metamodel) | |
| _D_container_ (Metamodel) | |
| _D_agent_ (Metamodel) | |
| _LibraryMetaData_ | |
| A | |
| B | |
| W | |
| _G_ | |
| AnimRep (Metamodel) | String (STRING) |
| AttrRep (Metamodel) | Longstring (LONGSTRING) |
| Class cardinality (Metamodel) | String (STRING) |
| ClassAbstract | Integer (INTEGER) |
| ClassName | String (STRING) |
| ClassVisible | |
| External tool cou | |
| GraphRep (Meta | |
| HlpTxt (Metamo | |
| Model pointer (M | |
| Position (Metam | |
| VisibleAttrs (Met | |
| WF_Trans (Meta | |
| AnimRep (Metamod | |
| AttrRep (Metamode | |
| Class cardinality (Me | |
| 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) |
| VisibleAttrs (Metamodel) | String (STRING) |
| WF_Trans (Metamodel) | String (STRING) |
| Relation classes | |

Class name: H

Superclass: _G_

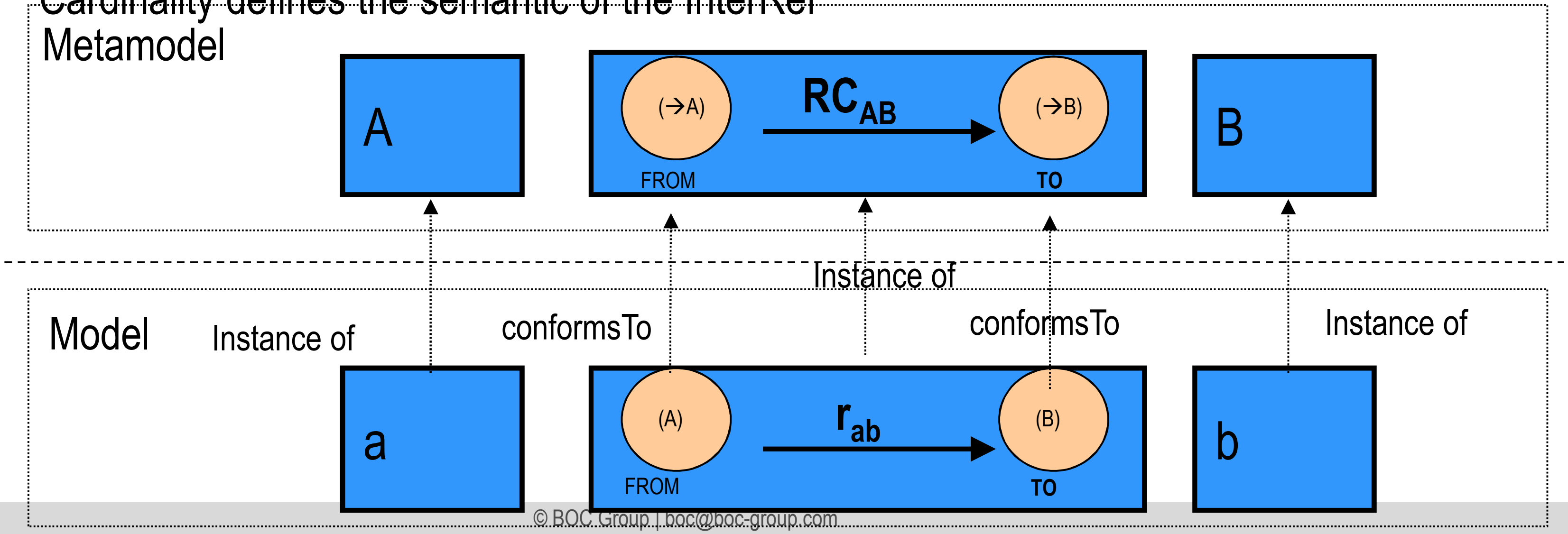
1. Add a new concrete class below the abstract element that is used to define a concrete class
2. Select the abstract class, click "New" -> "New class"
3. Name the new class

The new created class can be identified on instance level by the "Name" attribute. This attribute is automatically/implict available for each class

# RELATION TYPES

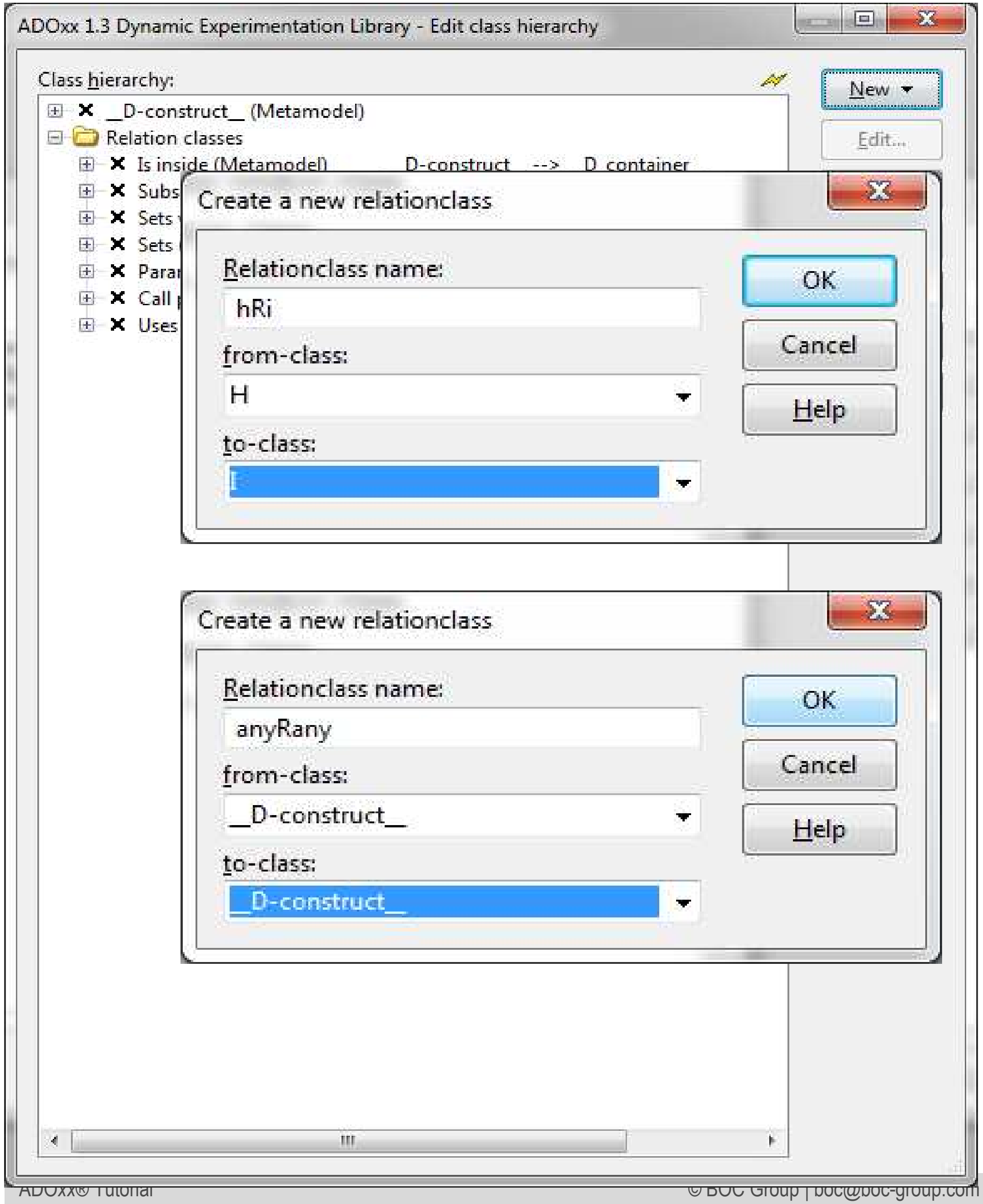
Define new classes
[abstract | concrete | subclass]
[structure and properties]

- Relations in ADOxx are expressed either as a class “Relation Class” or as a pointer in form of an attribute called “InterRef”.
- Relation as Class “RC”
  - describes relationship between two objects from two or more classes within one model.
  - has start and endpoints define which (abstract) classes a relation can connect
  - Cardinality and attribute defined the semantic of the relations class
- Relation as Attribute “InterRef”
  - Is a special configuration of a Relation Class and describes the relationship between two objects from two or more classes within or across models.
  - Is a pointer represented as an attributed in the class the relation starts from, with defined classes the relation can point to.
  - Cardinality defines the semantic of the InterRef



# DEMONSTRATION: RELATION CLASS DEFINITION

Define new classes [abstract | concrete | subclass]
[structure and properties]


ADOxx 1.3 Dynamic Experimentation Library - Edit class hierarchy
Class hierarchy:
__D-construct__ (Metamodel)
Relation classes
Is inside (Metamodel) D-construct --> D container
Subs
Sets
Sets
Para
Call
Uses
New
Edit...
Create a new relationclass
OK
Cancel
Help

Create a new relationclass

Relationclass name: hRi
from-class: H
to-class:

Create a new relationclass

Relationclass name: anyRany
from-class: __D-construct__
to-class: __D-construct__

- Add two new relation classes to connect classes
  - Click “New” -> “New relation class”
  - Name new relation class
  - Define from-class
  - Define to-class

# DEFINITION OF ATTRIBUTES

- Attributes for classes and relation classes have to be defined in the definition section of the class/relation class with 'TYPE'.

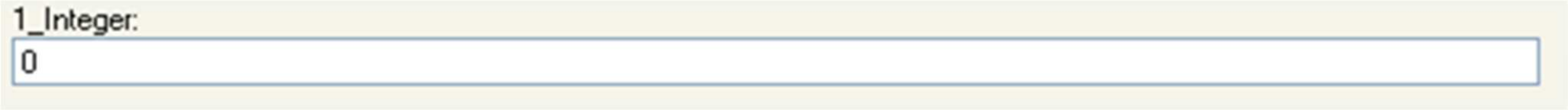
- The following attribute types are possible:

| | |
|---|---|
| **INTEGER** | **integer** |
| **DOUBLE** | **floating number** |
| **STRING** | **string – max. 3699 symbols** |
| LONGSTRING | string – max. 32000 symbols |
| ENUMERATION | enumeration for selecting a characteristic |
| ENUMERATIONLIST | enumeration for selecting one or several characteristics |
| TIME | time |
| DATE | date |
| DATETIME | date and time |
| ***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 attribuite values |

# DEMONSTRATION:
# ATTRIBUTE TYPES AND THEIR APPEARANCE

Define new classes
[abstract | concrete | subclass]
[structure and properties]

## Numerical Attributes: Integer (INTEGER)

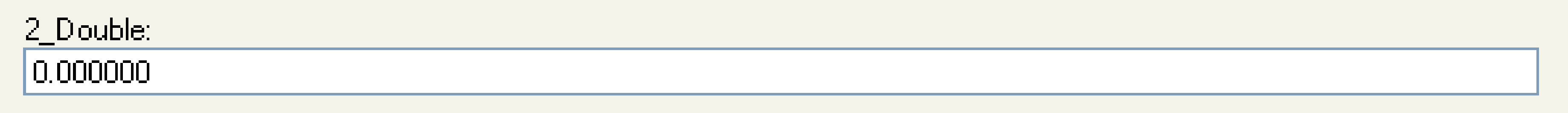

1_Integer:
0

- An attribute of the type "Integer" is defined as an integer from -1,999,999,999 to 1,999,999,999.
- An ADOxx integer is limited to 10 digits plus an optional sign ('+' or '-')
- The standard value of attributes of this type is "0" or a value defined

# DEMONSTRATION:
# ATTRIBUTE TYPES AND THEIR APPEARANCE

## Numerical Attributes: Floating number (DOUBLE)

Define new classes
[abstract | concrete | subclass]
[structure and properties]

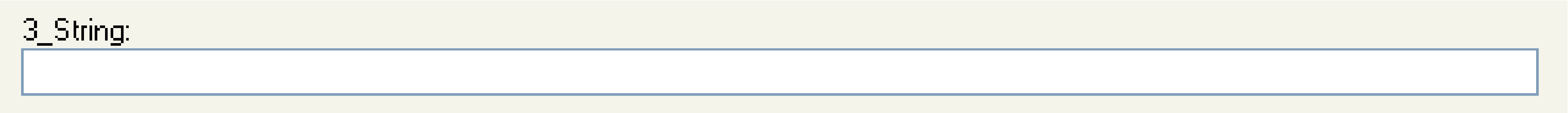

2_Double:
0.000000

- The amount of decimal places is defined by the attribute definition
- An attribute of the type "Double" is defined for a float within +/-999,999,999,999,999 for an integer (without decimal places) or +/-999,999,999.999999 for figures with 6 decimals.
- The corresponding attribute value is displayed to 6 decimal places. That means that a double value should not exceed a total of 15 significant digits with at last 6 decimal digits!
- The standard value of attributes of this type is "0.000000" or a value defined in the application library.

# DEMONSTRATION: ATTRIBUTE TYPES AND THEIR APPEARANCE

## String attributes: String (STRING)

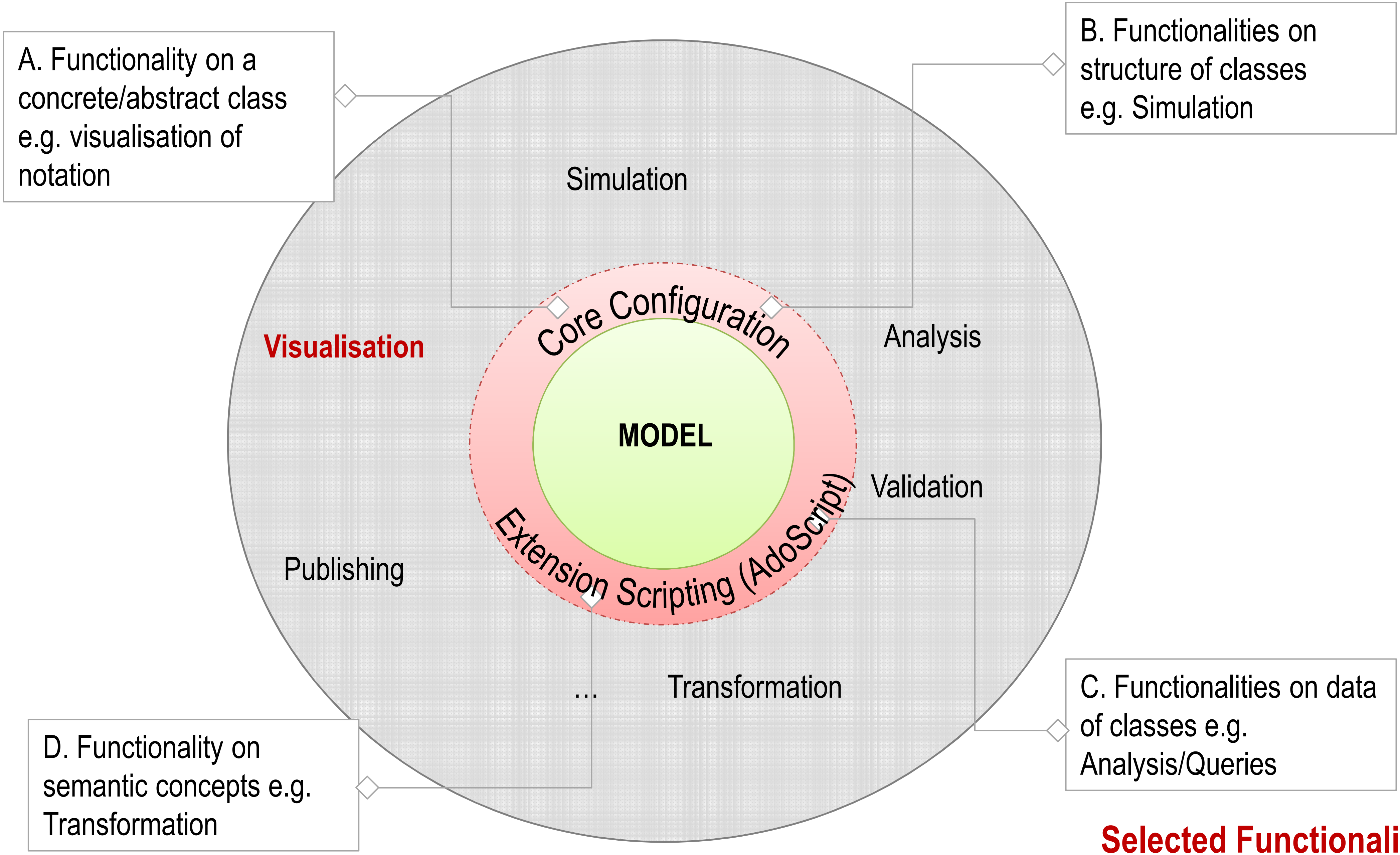
Define new classes [abstract | concrete | subclass]
[structure and properties]



- An attribute of the type "String" is defined for texts up to 3700 characters of any type.
  - Hint: The maximum number of characters is 250 for name. That concerns classes, relation, instances, attributes, application models, libraries and application libraries.
  - Model names have a special rule!

- The standard value of attributes of this type is "" (no entry) or a value defined in the application library.

# MODEL PROCESSING CLASSIFICATION



A. Functionality on a concrete/abstract class e.g. visualisation of notation

B. Functionalities on structure of classes e.g. Simulation

C. Functionalities on data of classes e.g. Analysis/Queries

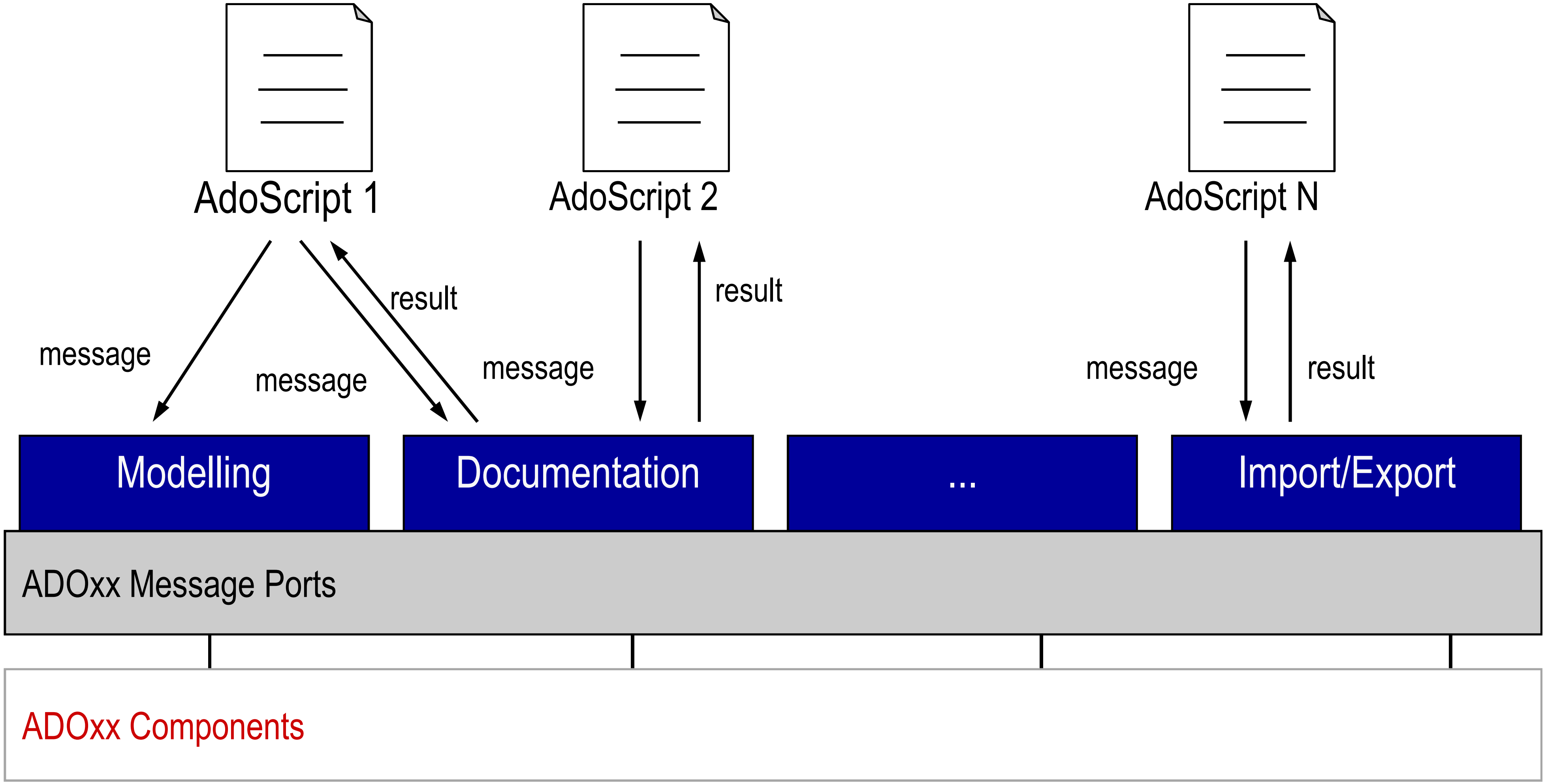
D. Functionality on semantic concepts e.g. Transformation

**Selected Functionalities for Tutorial**

# CORE CONFIGURATION

- User and Access Right Management
- File Management
- Library Persistence (DB and File Persistence)
- Model Persistence (DB and File Persistence)
- Serialization Functionality (Import/Export)
- …

# EXTENSION SCRIPTING (AdoScript)



# PROGRAMMABLE THROUGH SCRIPTING APIS

- Method-specific development of functionalities through scripting
- Function calls/APIs of the platform are possible through scripting.

***Component APIs***

Messageport **Acquisition**
Messageport **Modeling**
Messageport **Analysis**
Messageport **Simulation**
Messageport **Evaluation**
Messageport **ImportExport**
Messageport **Documentation**
Messageport **AQL**

***UI APIs***

Messageport **AdoScript**
Messageport **CoreUI**
Messageport **Explorer**

***Manipulation APIs***

Messageport **Core**
Messageport **DB**
Messageport **UsrMgt**

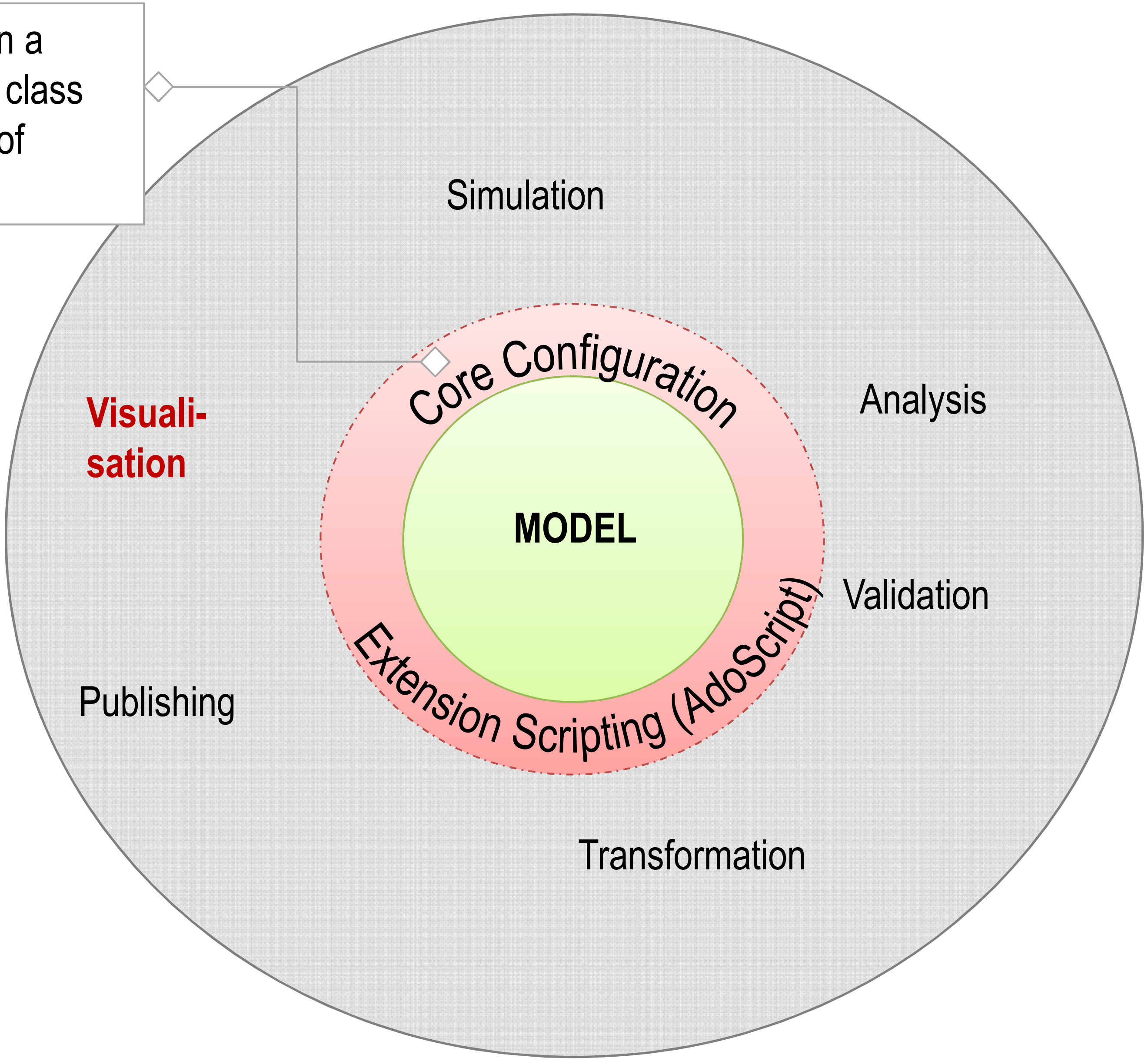
***Application APIs***

Messageport **Drawing**
Messageport **Application**

About 400 APIs are available.

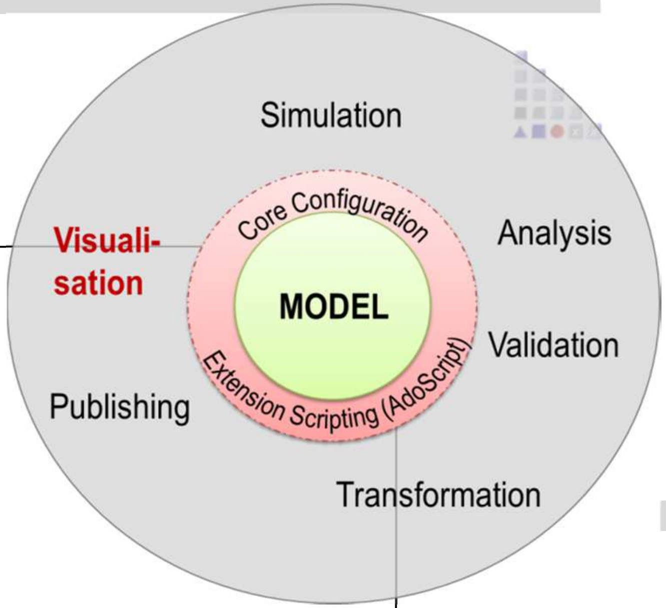
# SELECTED MODEL PROCESSING FUNCTIONALITY: VISUALISATION

A. Functionality on a concrete/abstract class e.g. visualisation of notation



**Selected Functionalities for Tutorial**

# Object Visualisation



**Platform Functionality**

- Object visualisation
- Model visualisation
  - Tabular view incl. view concept
  - Graphical view incl. view concept
  - Machine-generated models
    - Model analysis visualisation
    - Information visualisation
  - Human-generated models
    - Support functionality (automatic & user-defined)

**OMiLAB Development Tools**

- OMITool GraphRepGenerator
- OMITool AdoScript Syntax Highlighter

OPEN SOURCE

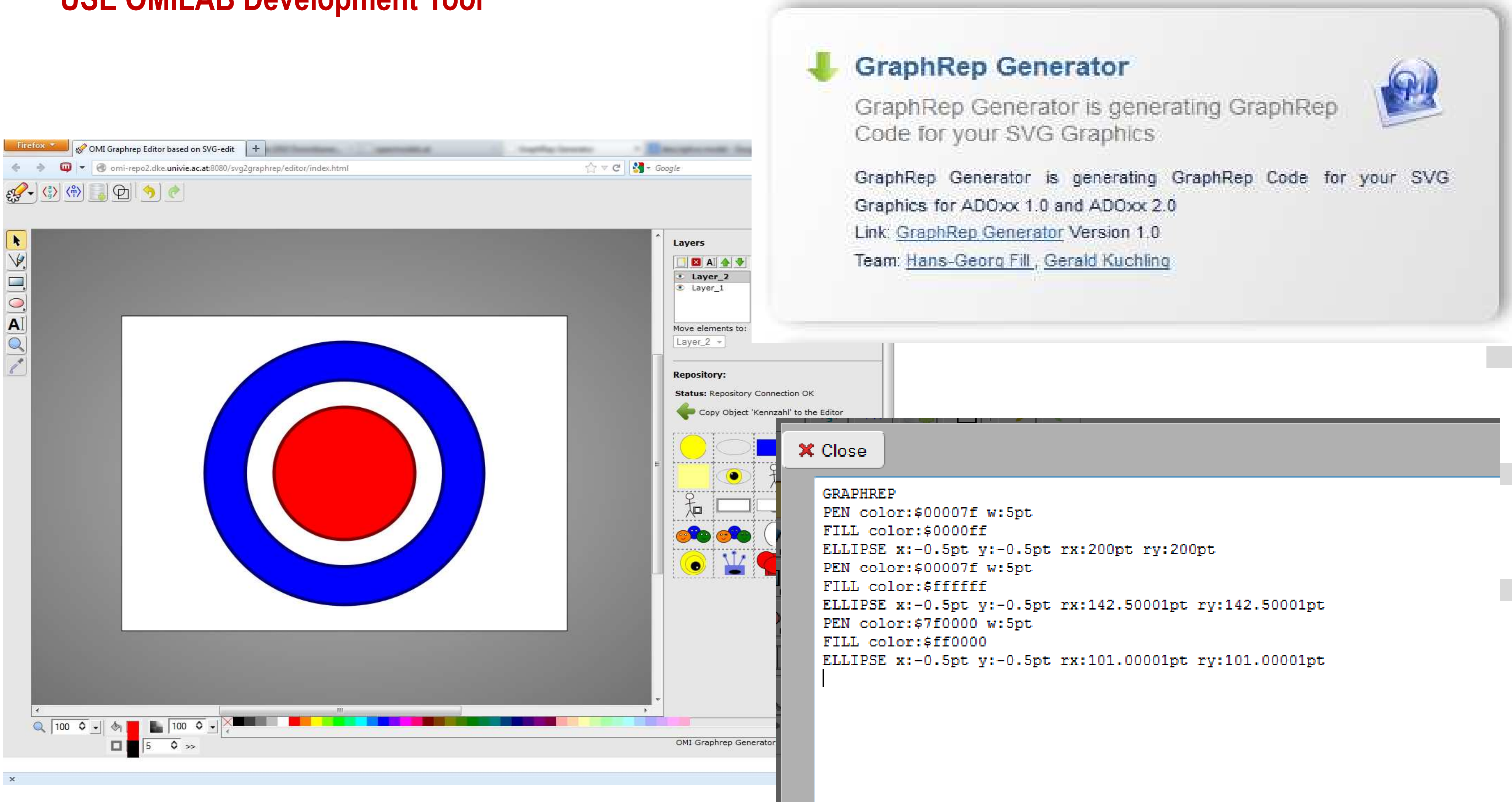
**Platform Technologies**

- GraphRep
- AdoScript

OPEN USE

# DEMONSTRATION: IMPLEMENTATION OF OBJECT VISUALISATION

## USE OMiLAB Development Tool


Firefox
OMI Graphrep Editor based on SVG-edit
omi-repo2.dke.univie.ac.at:8080/svg2graphrep/editor/index.html
Google
Layers
Layer_2
Layer_1
Move elements to:
Layer_2
100
100
5
OMI Graphrep Generator
Close

**Repository:**

**Status:** Repository Connection OK

Copy Object 'Kennzahl' to the Editor

**GraphRep Generator**

GraphRep Generator is generating GraphRep Code for your SVG Graphics

GraphRep Generator is generating GraphRep Code for your SVG Graphics for ADOxx 1.0 and ADOxx 2.0

Link: GraphRep Generator Version 1.0

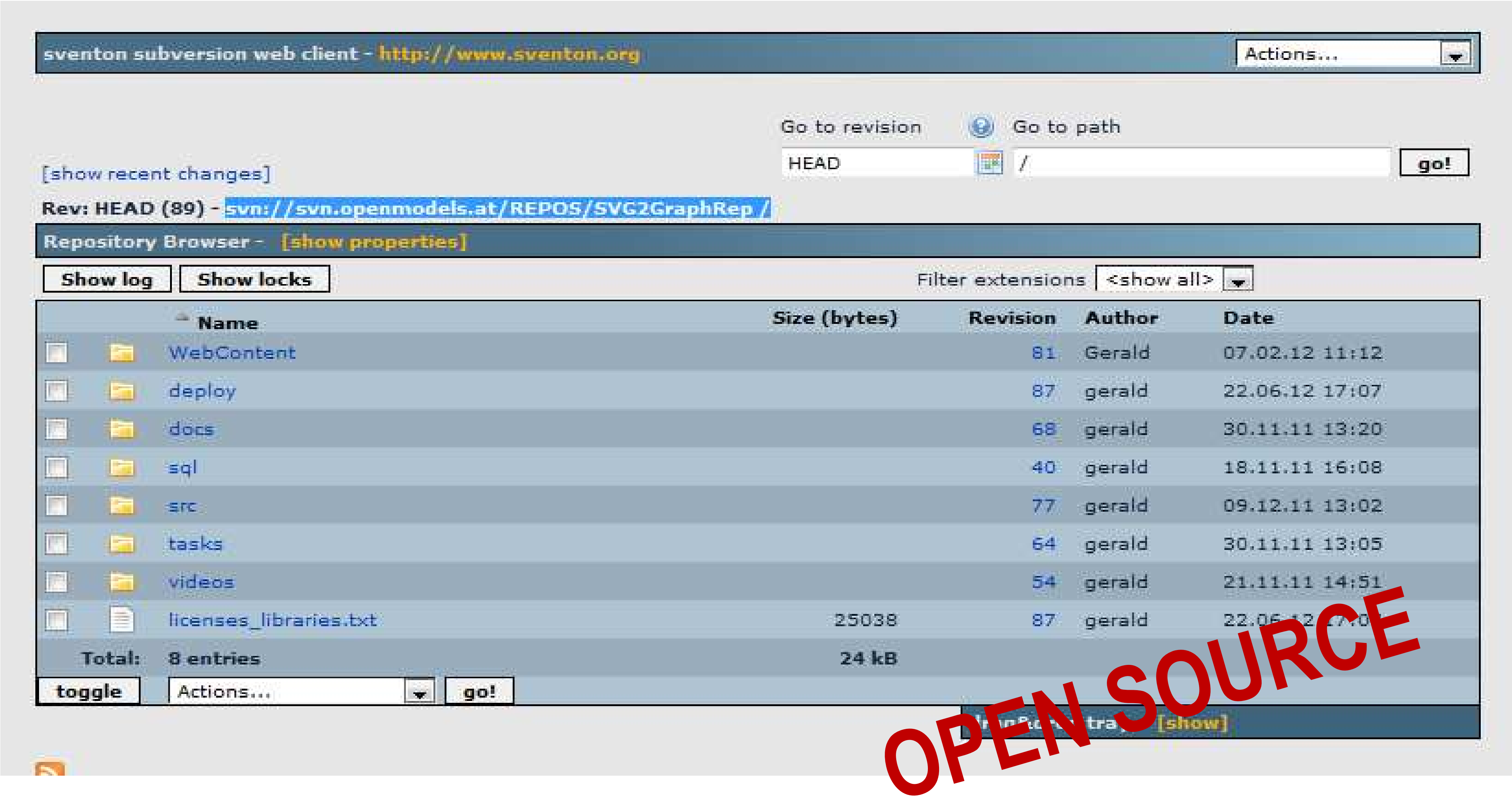
Team: Hans-Georg Fill , Gerald Kuchling

```
GRAPHREP
PEN color:$00007f w:5pt
FILL color:$0000ff
ELLIPSE x:-0.5pt y:-0.5pt rx:200pt ry:200pt
PEN color:$00007f w:5pt
FILL color:$ffffff
ELLIPSE x:-0.5pt y:-0.5pt rx:142.50001pt ry:142.50001pt
PEN color:$7f0000 w:5pt
FILL color:$ff0000
ELLIPSE x:-0.5pt y:-0.5pt rx:101.00001pt ry:101.00001pt
```

http://omi-repo2.dke.univie.ac.at:8080/svg2graphrep/editor/index.html

# DEMONSTRATION: IMPLEMENTATION OF OBJECT VISUALISATION

## CONTRIBUTE to OMiLAB Development



HEAD

/

Rev: HEAD (89) - svn://svn.openmodels.at/REPOS/SVG2GraphRep /

| | | Name | Size (bytes) | Revision | Author | Date |
|---|---|---|---|---|---|---|
| ☐ | | WebContent | | 81 | Gerald | 07.02.12 11:12 |
| ☐ | | deploy | | 87 | gerald | 22.06.12 17:07 |
| ☐ | | docs | | 68 | gerald | 30.11.11 13:20 |
| ☐ | | sql | | 40 | gerald | 18.11.11 16:08 |
| ☐ | | src | | 77 | gerald | 09.12.11 13:02 |
| ☐ | | tasks | | 64 | gerald | 30.11.11 13:05 |
| ☐ | | videos | | 54 | gerald | 21.11.11 14:51 |
| ☐ | | licenses_libraries.txt | 25038 | 87 | gerald | 22.06.12 17:0[illegible] |
| | Total: | 8 entries | 24 kB | | | |

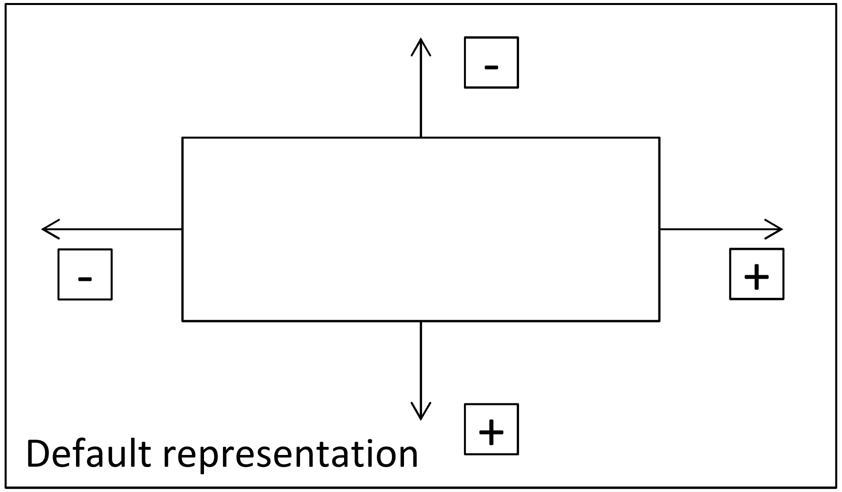
OPEN SOURCE

HTTP: http://omi-repo2.dke.univie.ac.at:8080/svg2graphrep/svn.htm

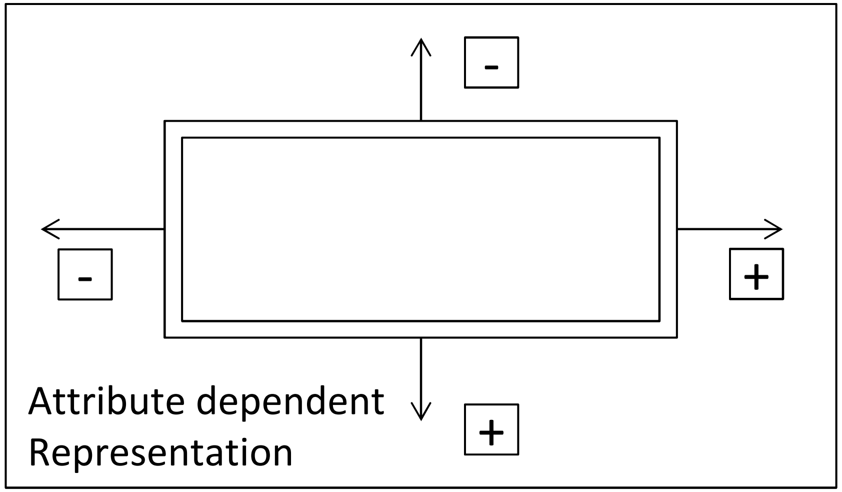
SVN: svn://svn.openmodels.at/REPOS/SVG2GraphRep /

# DEMONSTRATION: IMPLEMENTATION OF OBJECT VISUALISATION

## DEVELOPMENT on ADOxx Platform



Default representation



Attribute dependent Representation

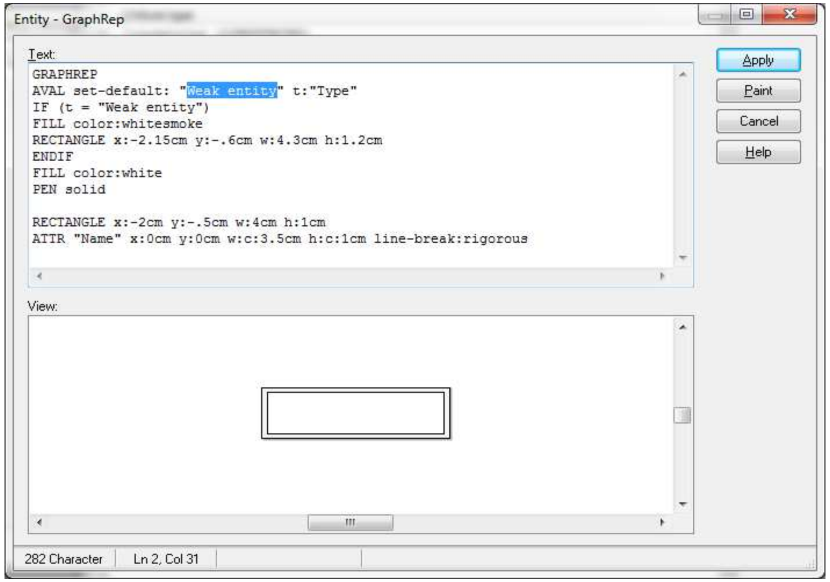
```
GRAPHREP
AVAL t:"Type"
IF (t = "Weak entity")
  FILL color:whitesmoke
  RECTANGLE x:-2.15cm y:-.6cm  w:4.3cm h:1.2cm
ENDIF
FILL color:white
PEN solid

RECTANGLE x:-2cm y:-.5cm w:4cm h:1cm
ATTR "Name" x:0cm y:0cm w:c:3.5cm h:c:1cm line-
break:rigorous
```

Conditional representation

Default representation

Name representation

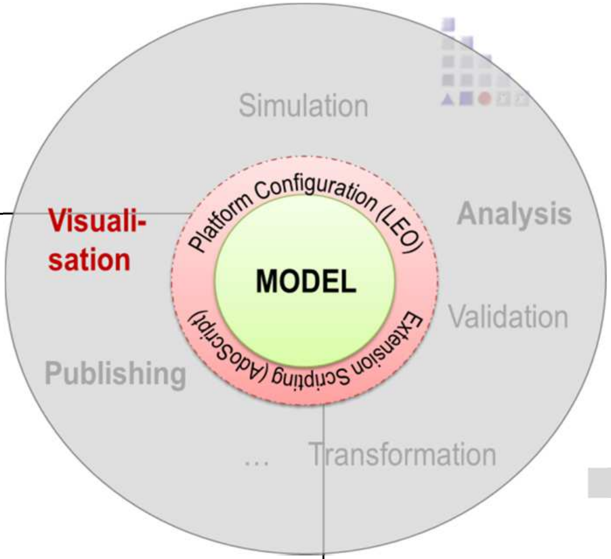


```
GRAPHREP
AVAL set-default: "Weak entity" t:"Type"
IF (t = "Weak entity")
FILL color:whitesmoke
RECTANGLE x:-2.15cm y:-.6cm w:4.3cm h:1.2cm
ENDIF
FILL color:white
PEN solid

RECTANGLE x:-2cm y:-.5cm w:4cm h:1cm
ATTR "Name" x:0cm y:0cm w:c:3.5cm h:c:1cm line-break:rigorous
```

282 Character | Ln 2, Col 31

# MODEL ANALYSIS VISUALISATION



**Platform Functionality**

- Object visualisation
- Model visualisation
  - Tabular view incl. view concept
  - Graphical view incl. view concept
  - Machine-generated models
    - Model analysis visualisation
    - Information visualisation
  - Human-generated models
    - Support functionality (automatic & user-defined)

**OMiLAB Development Tools**

- OMITool GraphRepGenerator
- OMITool AdoScript Syntax Highlighter

OPEN SOURCE

**Platform Technologies**

- GraphRep
- AdoScript

OPEN USE

# DEMONSTRATION: MODEL ANALYSIS VISUALISATION

## DEVELOPMENT on ADOxx Platform

```
## Get acti
CC "Modelir
SETL id_sta

# make an i
convert val
CC "AdoScri
STR(id_star
id!"

## count ho
modelled ir

# get the i
CC "Core" g
SETL id_cla

#----------
GET_CLASS_I
#----------
[ecode:intV

# get all o
CC "Core" g
modelid:(ic
```

```
IF (LEN (objid
{
  CC "AdoScrip
instances of c
  EXIT
}
SETL debug cou
CC "AdoScript"
STR(count_of_o
objects of cla

## Creating a
CC "CoreUI" MO
models title:"
boxtext:"Selek
in der Datenba

# ------------
This MODEL SEL
variables
# ------------
modelids: idLi
mgroupids: idL
extraValues ]
# ------------
global variabl
MODEL
```

```
CC "Core" CREATE_MODEL modeltype:"Result-Type 1"
                       modelname:"My First own
result"
                       version:"1.0"
                       mgroups:(mgroupids)

# open the new created model AND to make the new
model ACTIVE
IF (ecode = 0)
{
   CC "Modeling" CREATE_WINDOW_FOR_LOADED_MODEL
modelid:(modelid)
}

## Create objects in the new model

# get the model ide of the new model
CC "Modeling" GET_ACT_MODEL
SETL id_resultmodel:(modelid)

# make an info box for debuggin reasons -
convert value of id_actmodel into a string
CC "AdoScript" INFOBOX ("Hello " +
STR(id_resultmodel) + "!") title:"Result model
id!"
```

# AGENDA

1. INTRODUCTION/MODEL VALUE

[Motivation, Positioning of Approach]

2. BASLINE/DEFINITIONS

[Terms, Definition, Foundation, Concepts]

3. IMPLEMENT MODEL STRUCTURE using ADO*xx*

[Hands-On "Hello World" Model Structure, ADOxx Concepts]

4. ENABLE MODEL PROCESSING using ADO*xx*

[Configure/Implement Model Processing Mechanisms/Algorithms, ADOxx Development Concepts]

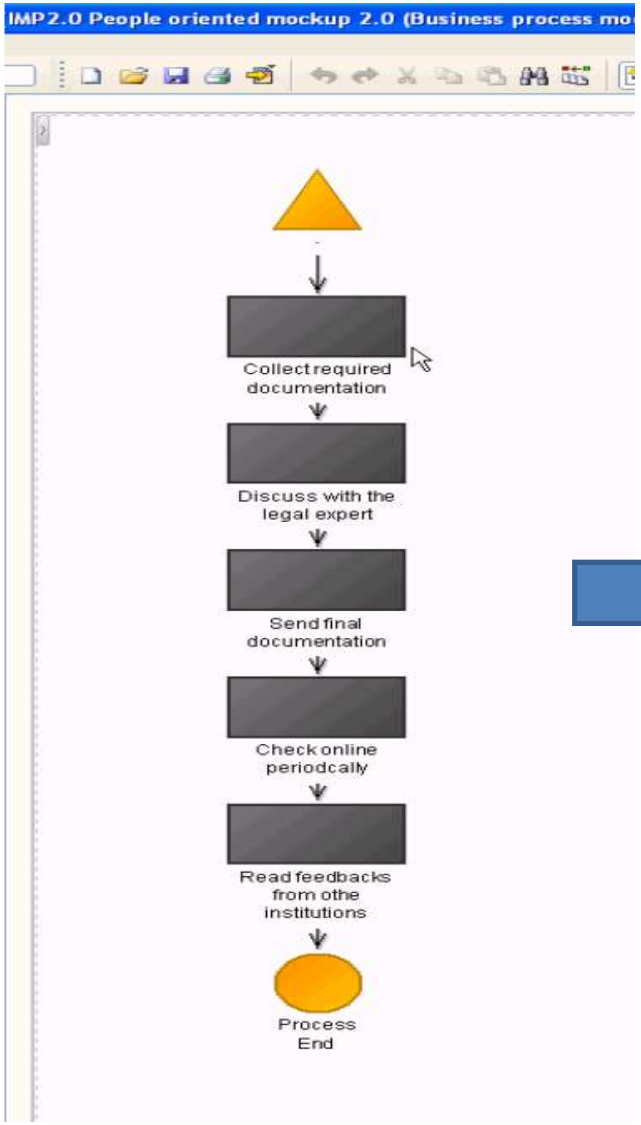
**5. APPLICATION CASES**

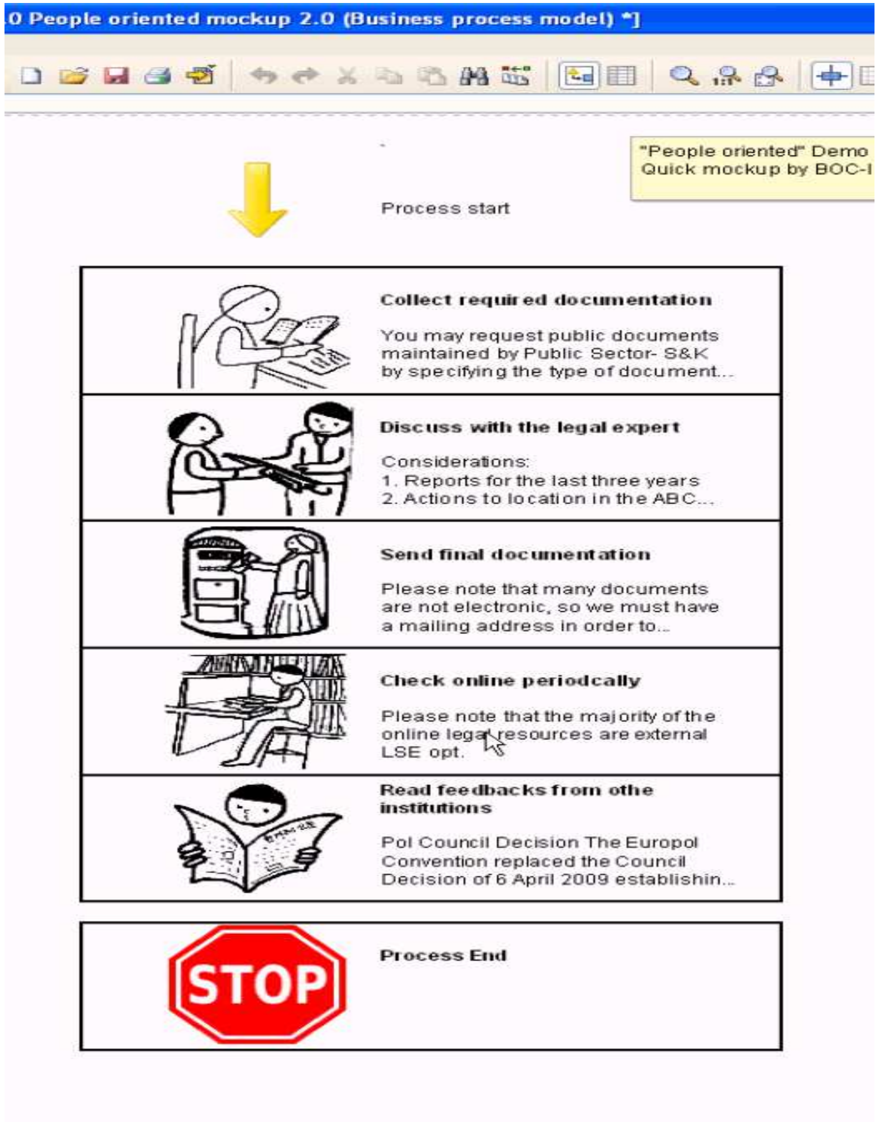
**[Case studies and additional resources for development]**

6. CONCLUSIONS/FURTHER READINGS

OMiLAB®
www.omilab.org

# VISUALISATION CASE:
# PEOPLE-CENTRIC PROCESS VIEWS


IMP2.0 People oriented mockup 2.0 (Business process mo
Collect required documentation
Discuss with the legal expert
Send final documentation
Check online periodcally
Read feedbacks from othe institutions
Process End



**Collect required documentation**

You may request public documents maintained by Public Sector- S&K by specifying the type of document...

**Discuss with the legal expert**

Considerations:
1. Reports for the last three years
2. Actions to location in the ABC...

**Send final documentation**

Please note that many documents are not electronic, so we must have a mailing address in order to...

**Check online periodcally**

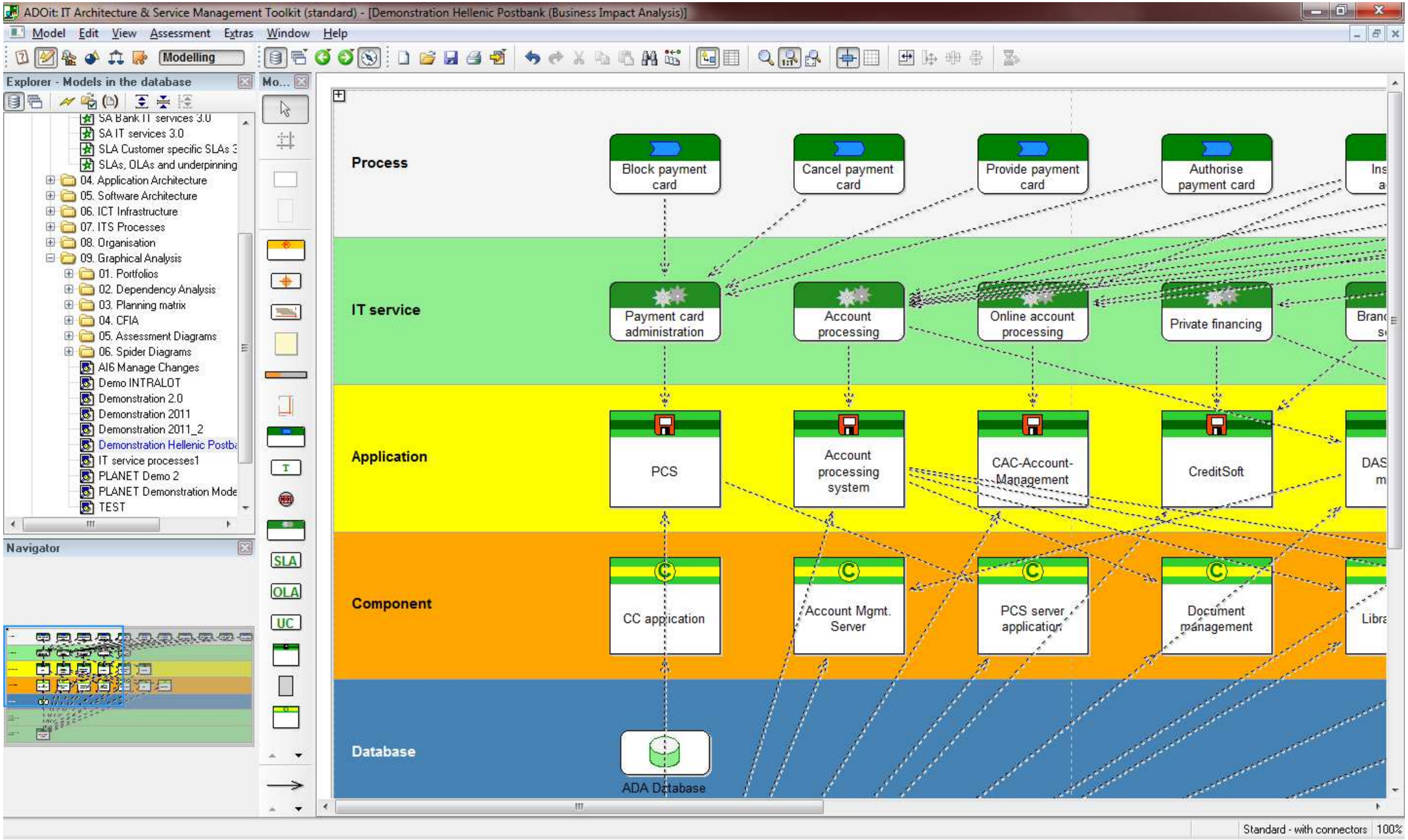
Please note that the majority of the online legal resources are external LSE opt.

**Read feedbacks from othe institutions**

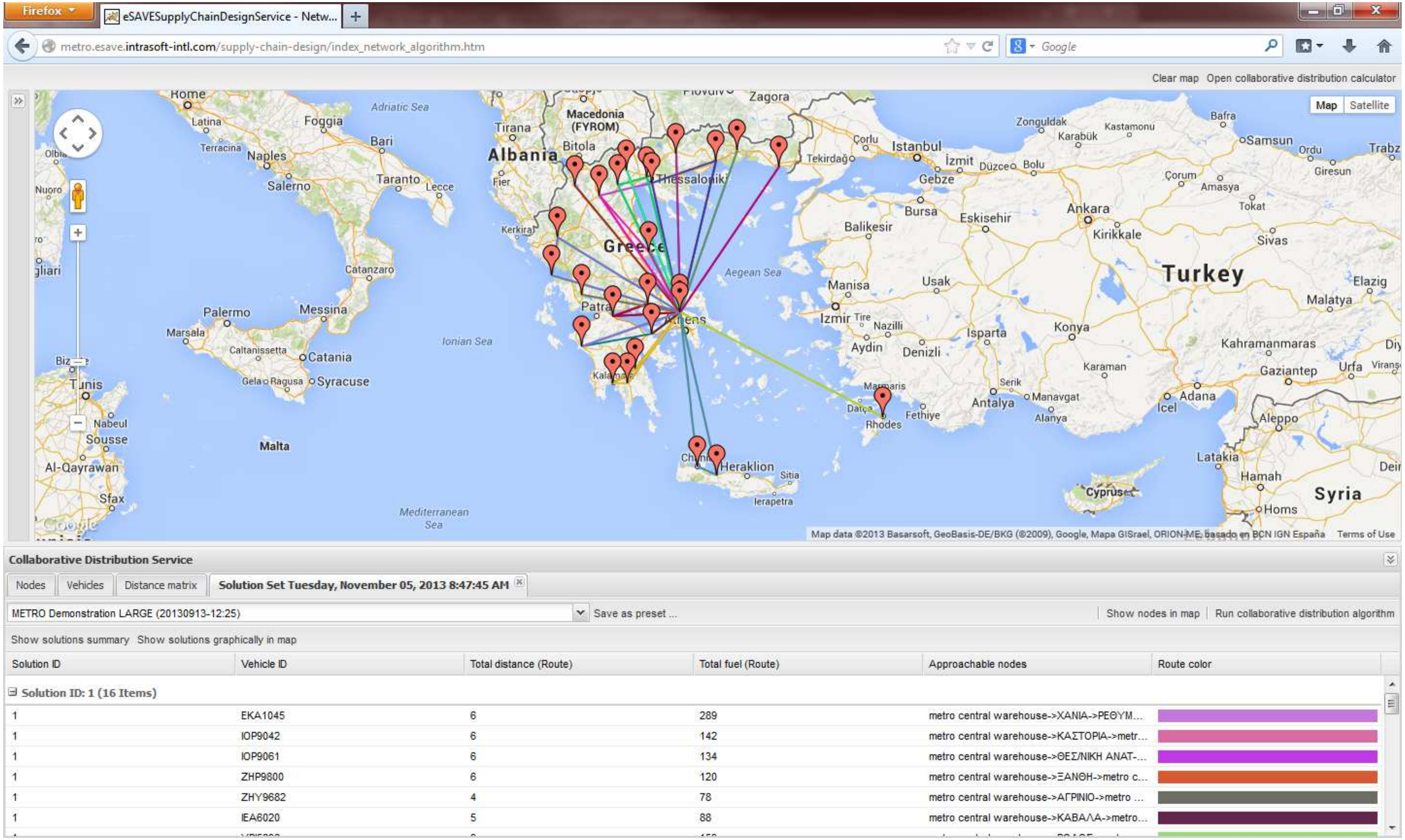
Pol Council Decision The Europol Convention replaced the Council Decision of 6 April 2009 establishin...

**Process End**

# VISUALISATION CASES: BUSINESS IMPACT ANALYSIS IN IT ARCHITECTURES



# VISUALISATION CASE:
# OPTIMIZATION OF SUPPLY CHAINS FOR ENERGY EFFICIENCY


Firefox
eSAVESupplyChainDesignService - Netw...
metro.esave.intrasoft-intl.com/supply-chain-design/index_network_algorithm.htm
Google
Clear map
Open collaborative distribution calculator
Map
Satellite
Adriatic Sea
Ionian Sea
Aegean Sea
Mediterranean Sea
Albania
Macedonia (FYROM)
Greece
Turkey
Syria
Thessaloniki
Athens
Patra
Heraklion
Rhodes
Istanbul
Izmir
Ankara
Map data ©2013 Basarsoft, GeoBasis-DE/BKG (©2009), Google, Mapa GISrael, ORION-ME, basado en BCN IGN España
Terms of Use

**Collaborative Distribution Service**

Nodes | Vehicles | Distance matrix | **Solution Set Tuesday, November 05, 2013 8:47:45 AM**

METRO Demonstration LARGE (20130913-12:25) | Save as preset ... | Show nodes in map | Run collaborative distribution algorithm

Show solutions summary   Show solutions graphically in map

| Solution ID | Vehicle ID | Total distance (Route) | Total fuel (Route) | Approachable nodes | Route color |
|---|---|---|---|---|---|
| **⊟ Solution ID: 1 (16 Items)** | | | | | |
| 1 | EKA1045 | 6 | 289 | metro central warehouse->XANIA->PEΘYM... | 🟪 |
| 1 | IOP9042 | 6 | 142 | metro central warehouse->ΚΑΣΤΟΡΙΑ->metr... | 🟥 |
| 1 | IOP9061 | 6 | 134 | metro central warehouse->ΘΕΣ/ΝΙΚΗ ΑΝΑΤ-... | 🟪 |
| 1 | ZHP9800 | 6 | 120 | metro central warehouse->ΞΑΝΘΗ->metro c... | 🟧 |
| 1 | ZHY9682 | 4 | 78 | metro central warehouse->ΑΓΡΙΝΙΟ->metro ... | ⬛ |
| 1 | IEA6020 | 5 | 88 | metro central warehouse->ΚΑΒΑΛΑ->metro... | 🟪 |

# AGENDA

1. INTRODUCTION/MODEL VALUE

[Motivation, Positioning of Approach]

2. BASLINE/DEFINITIONS

[Terms, Definition, Foundation, Concepts]

3. IMPLEMENT MODEL STRUCTURE using ADOxx

[Hands-On "Hello World" Model Structure, ADOxx Concepts]

4. ENABLE MODEL PROCESSING using ADOxx

[Configure/Implement Model Processing Mechanisms/Algorithms, ADOxx Development Concepts]

5. APPLICATION CASES

[Case studies and additional resources for development]

6. CONCLUSIONS/FURTHER READINGS

OMiLAB®

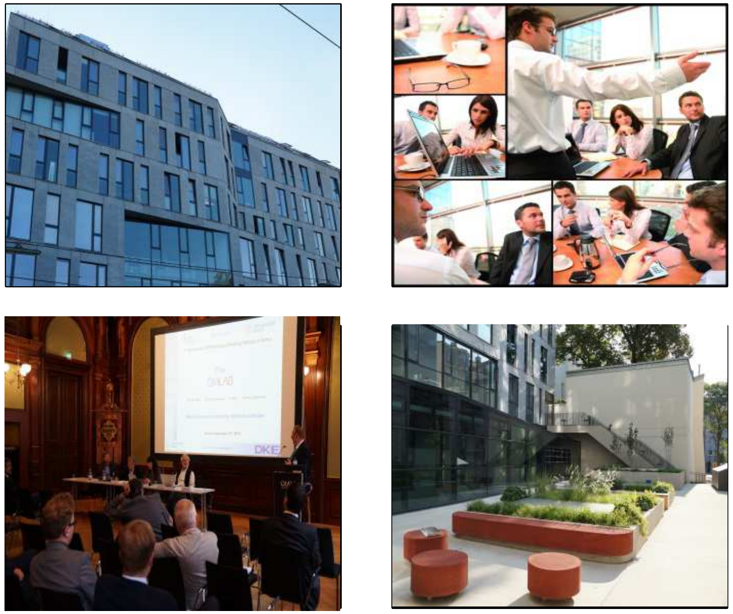
www.omilab.org

# OMiLAB: Approach

OMiLAB®

www.omilab.org

- A **research and experimental laboratory** for the conceptualization, development and deployment of modelling methods and the models designed with them.
- Project space for Engineering of modelling methods and **modelling tools**
- A space for a community of researchers and practitioners sharing a common understanding about **model value**



OMiLAB@Faculty of Computer Science
Währinger Str. 29

**Organisation:** University of Vienna,
Faculty of Computer Science

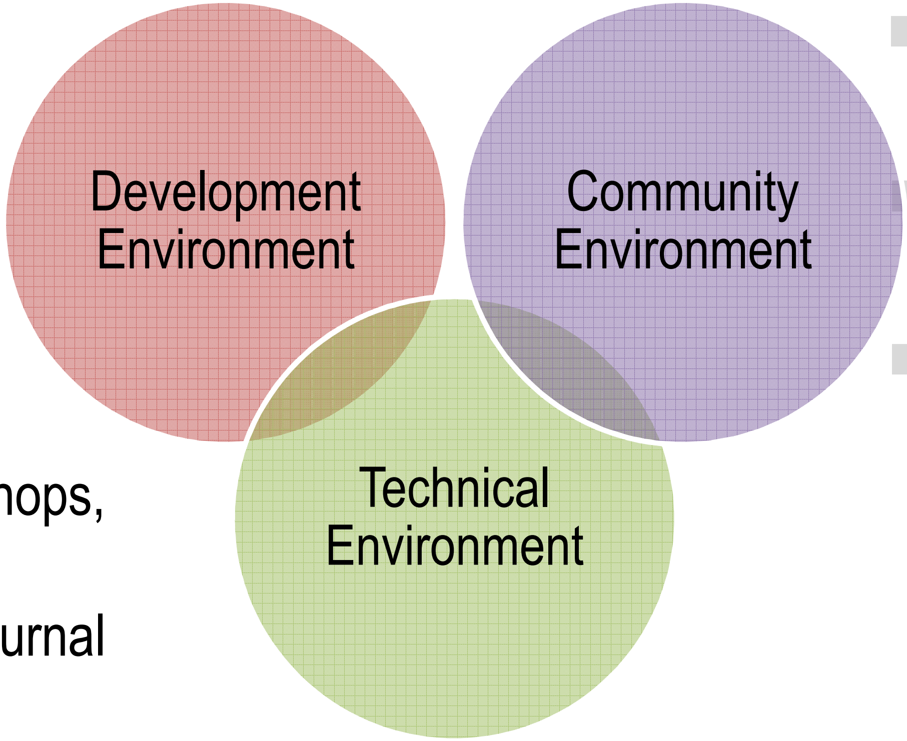
**Research Group:** Knowledge Engineering

Fakultät für Informatik

# OMiLAB: Environment

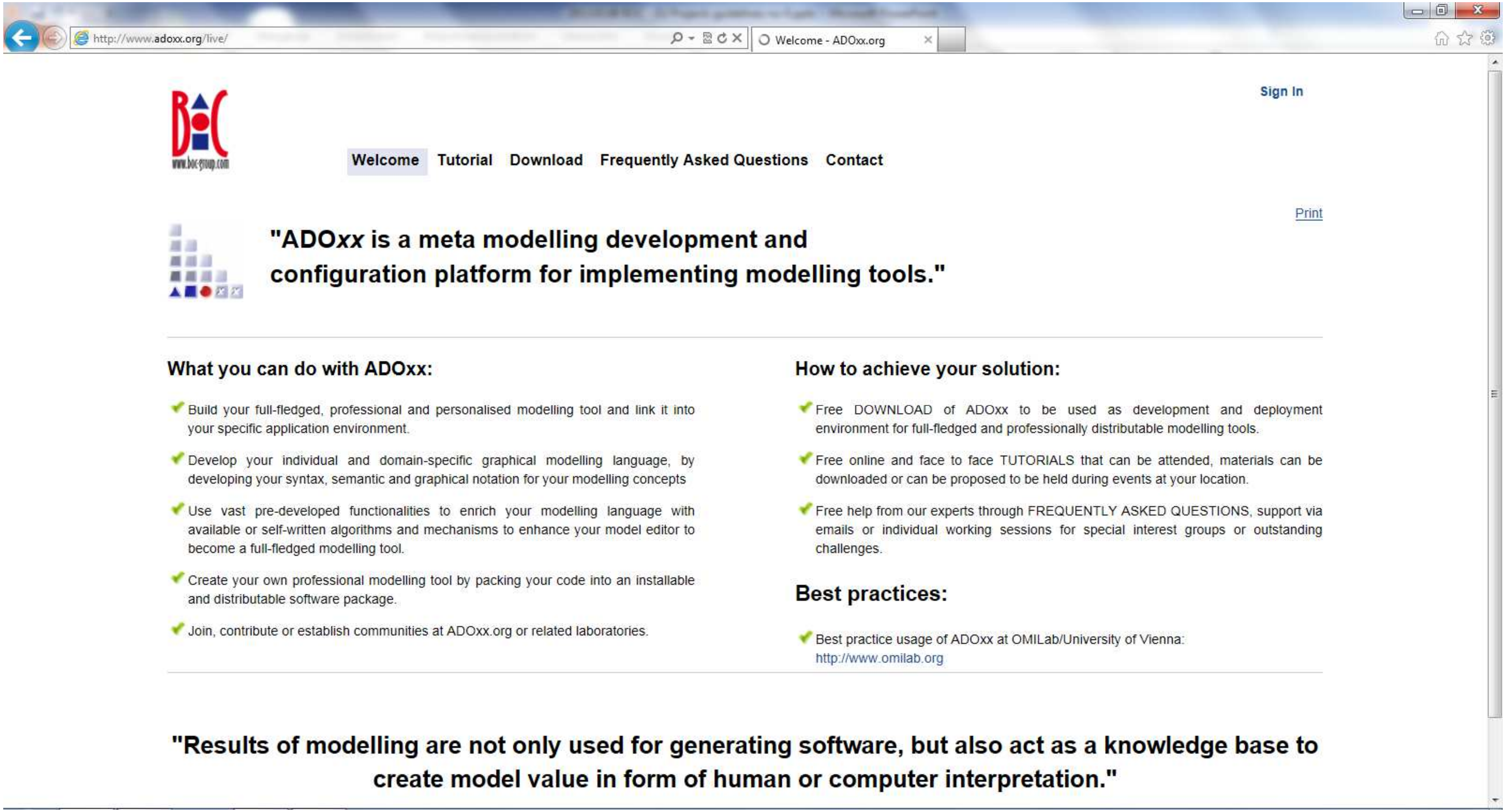
www.omilab.org

- Development environment consists of
  - Core (Open Use): ADOxx on OMiLAB
  - Add-Ons (Open Source): implemented community tools such as Model Annotator, GraphRep Generator, Model Publisher, Method Publisher, OM-Repository, Meta-Model Browser, MLEA – Modelling Language Engineering Assistant
- Technical environment supports
  - virtual and physical accessibility
  - packaging and deployment capabilities
- Community environment provides
  - Web-platform based on Liferay
  - Community events like conferences, workshops, summer schools
  - Publications like books, conference and journal papers
  - Project networking activities
  - Newsletters, media and OM-TV



**www.omilab.org**

# ADOxx.ORG : A Meta Model Platform Community



"ADO*xx* is a meta modelling development and configuration platform for implementing modelling tools."

**What you can do with ADOxx:**

- ✓ Build your full-fledged, professional and personalised modelling tool and link it into your specific application environment.
- ✓ Develop your individual and domain-specific graphical modelling language, by developing your syntax, semantic and graphical notation for your modelling concepts
- ✓ Use vast pre-developed functionalities to enrich your modelling language with available or self-written algorithms and mechanisms to enhance your model editor to become a full-fledged modelling tool.
- ✓ Create your own professional modelling tool by packing your code into an installable and distributable software package.
- ✓ Join, contribute or establish communities at ADOxx.org or related laboratories.

**How to achieve your solution:**

- ✓ Free DOWNLOAD of ADOxx to be used as development and deployment environment for full-fledged and professionally distributable modelling tools.
- ✓ Free online and face to face TUTORIALS that can be attended, materials can be downloaded or can be proposed to be held during events at your location.
- ✓ Free help from our experts through FREQUENTLY ASKED QUESTIONS, support via emails or individual working sessions for special interest groups or outstanding challenges.

**Best practices:**

- ✓ Best practice usage of ADOxx at OMILab/University of Vienna: http://www.omilab.org

**"Results of modelling are not only used for generating software, but also act as a knowledge base to create model value in form of human or computer interpretation."**

## www.adoxx.org

# We thank you for your attention!

For further questions please contact:

www.boc-group.com

Mag. Wilfrid Utz
Researcher/ADOxx.org Team
-------------------------------------------------------
BOC Asset Management GmbH
Operngasse 20b, A-1040 Vienna

Phone: +43-1-905 10 56 0
Fax: +43-1-905 10 56 3007
eMail: wilfrid.utz@boc-eu.com