

Tutorial: DB-Main 9.1.1 (based on DB-Main help)



Profa. Dra. Sarajane Marques Peres
www.each.uspnet.usp.br/sarajane

DB-Main 9.1.1 – General Information

- ▶ General Purpose Modeling Tool For Database Applications Engineering
 - ▶ supporting toolset for system engineering;
 - ▶ modeling tool development environment – Computer Aided Software Engineering (CASE) tool.
- ▶ A product of the Laboratory of Database Application Engineering **University of Namur.**
- ▶ Developed and distributed by: REVER S.A. (Belgium)
- ▶ Web Site: www.db-main.eu Support: dbm@rever.eu Basic file extension: .lun
- ▶ Release 9.1.1 – February, 2010.



Modelling functionalities (examples)

- ▶ This tool offers support for some forward and reverse engineering activities.
- ▶ Forward engineering
 - ▶ A generic, wide-spectrum representation model for conceptual, logical and physical objects;
 - ▶ A graphical representation of ERA Schemas, UML Class, Activity and Use Case Diagrams;
 - ▶ A generic model that describes procedural components of information systems at various abstraction levels as well as their relations;
 - ▶ Semantic and technical semi-formal annotations attached to each specification object;



Modelling functionalities (examples)

▶ Reverse engineering

- ▶ Code parsers extracting physical schemas from SQL, COBOL, CODASYL, IMS, XML DTD and RPG source programs;
- ▶ Interactive and programmable source text analyzers that can be used, a.o., to detect complex programming patterns or *clichés* in source texts, and to build data flow diagrams through program variables;



Modelling functionalities (examples)

▶ CASE - Typical features

- ▶ A toolbox of semantics-preserving transformational operators intended to carry out in a systematic way such activities as conceptual normalization, development of optimized logical and physical schemas from conceptual ones, and conversely (i.e. reverse engineering);
- ▶ A set of assistants. An assistant is a kind of expert in a specific kind of tasks, or in a class of problems. It is intended to help the analyst to carry out frequent, tedious or complex tasks.



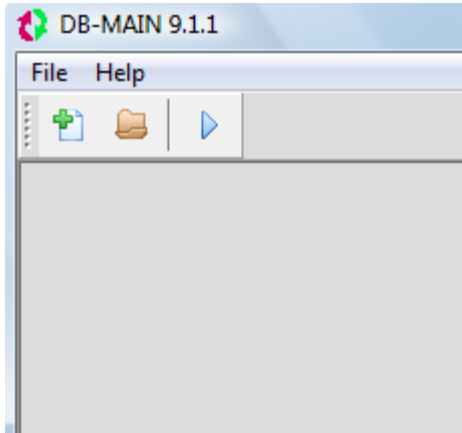
Modelling functionalities (examples)

▶ Meta level functions

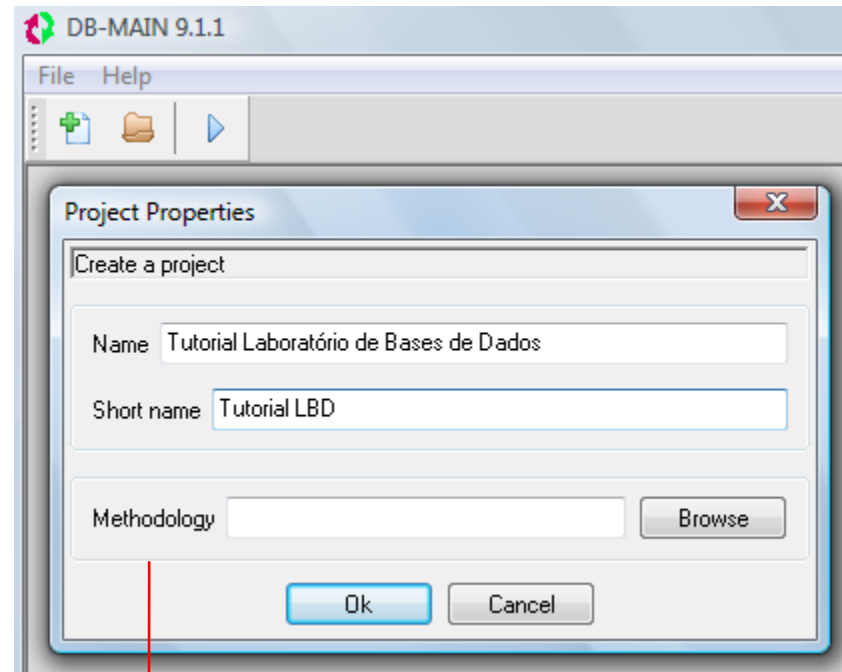
- ▶ *Java Interface for DB-MAIN (JIDBM)* language and the *Voyager 2* allow the engineer (analyst or method engineer) to develop new functions that can be seamlessly incorporated into the tool;
- ▶ Extension of the repository : new properties can be dynamically added and managed through plug-ins;
- ▶ Methodological customization: the tool is methodologically neutral and can assist the analyst in following a large spectrum of methodologies.



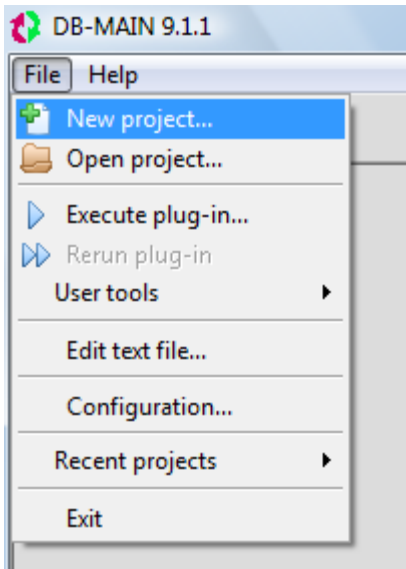
Basic Interface Views



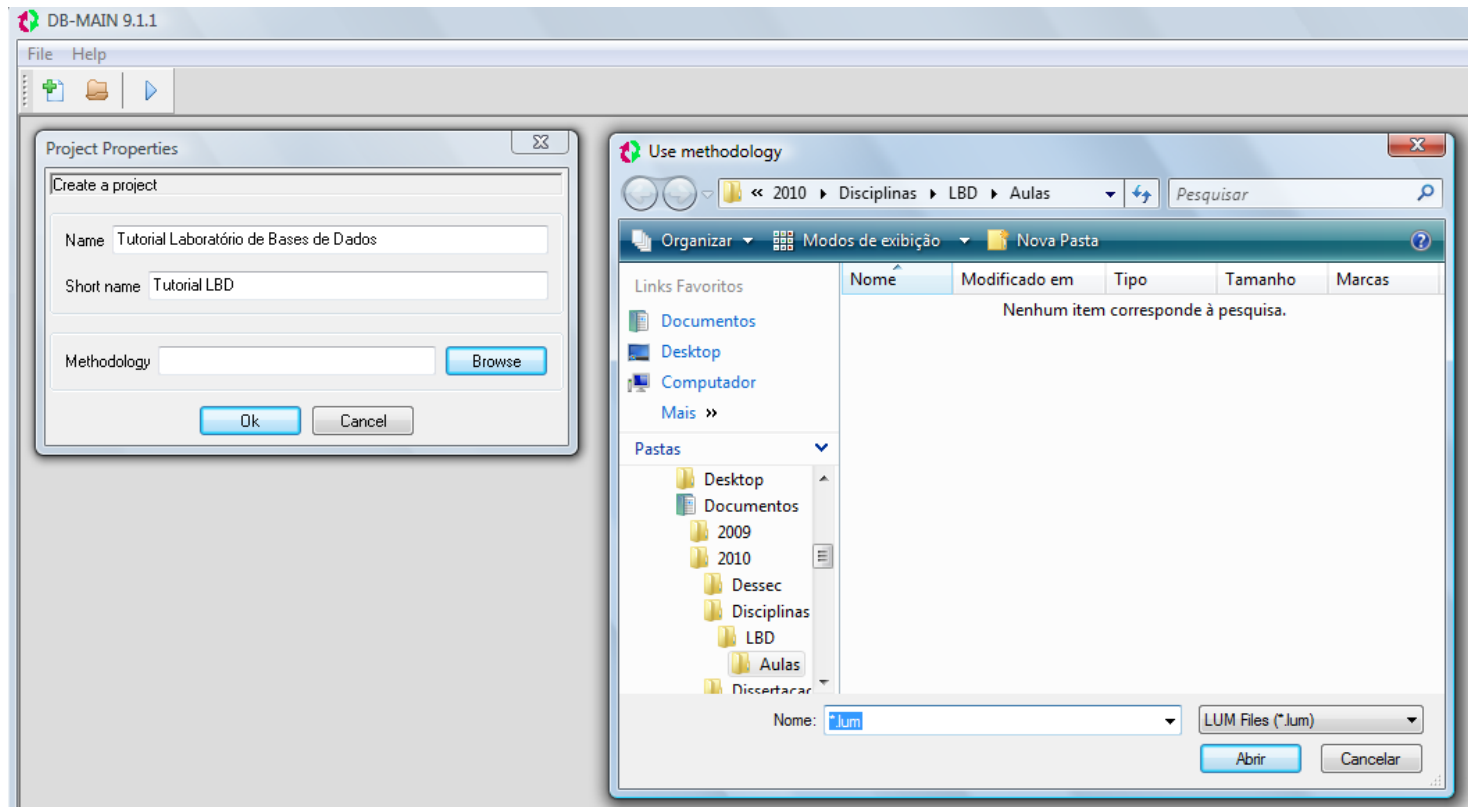
Our project



to choose a *.LUM file containing a method to be followed during the project

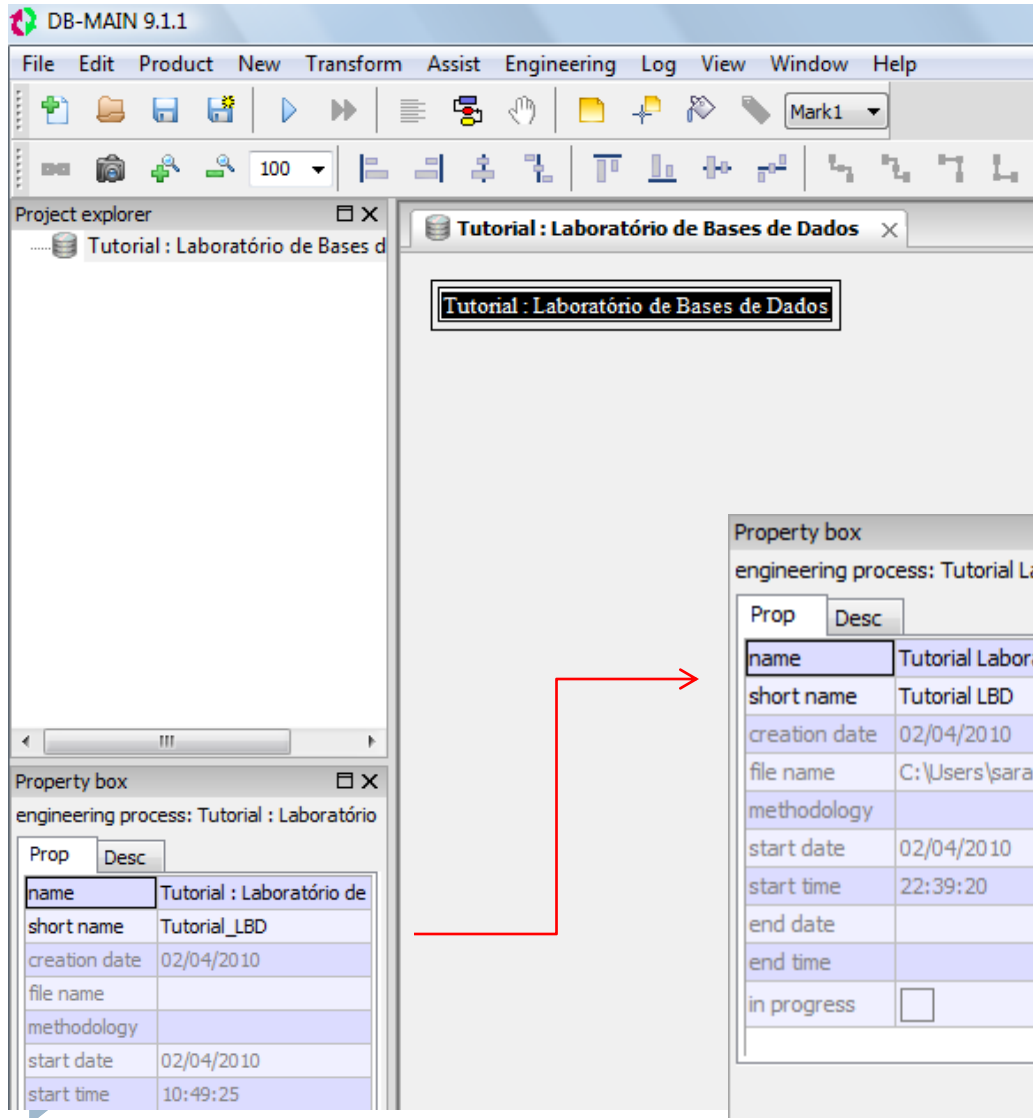


Basic Interface Views

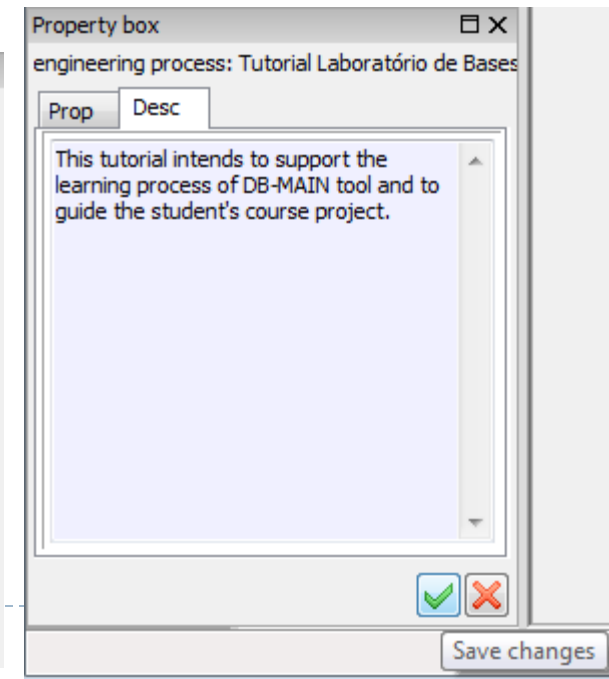


Engineering methods must be written in MDL (Method Description Language) and compiled with the external MDL compiler to produce a ***.LUM** file. This file can be associated to the project at creation time (**File/New project** menu item). If no method is specified, the default one allows to do anything.

Basic Interface Views



The Property Box (or similar) with context description must be completed (in the course student project).

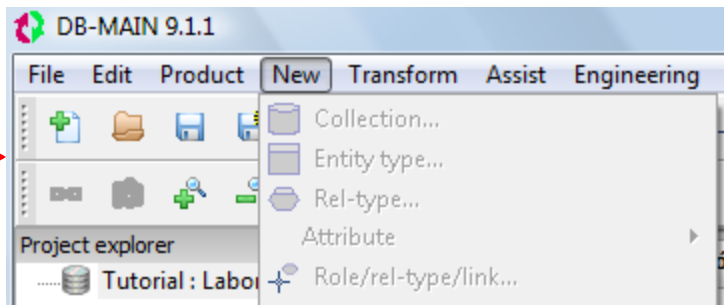
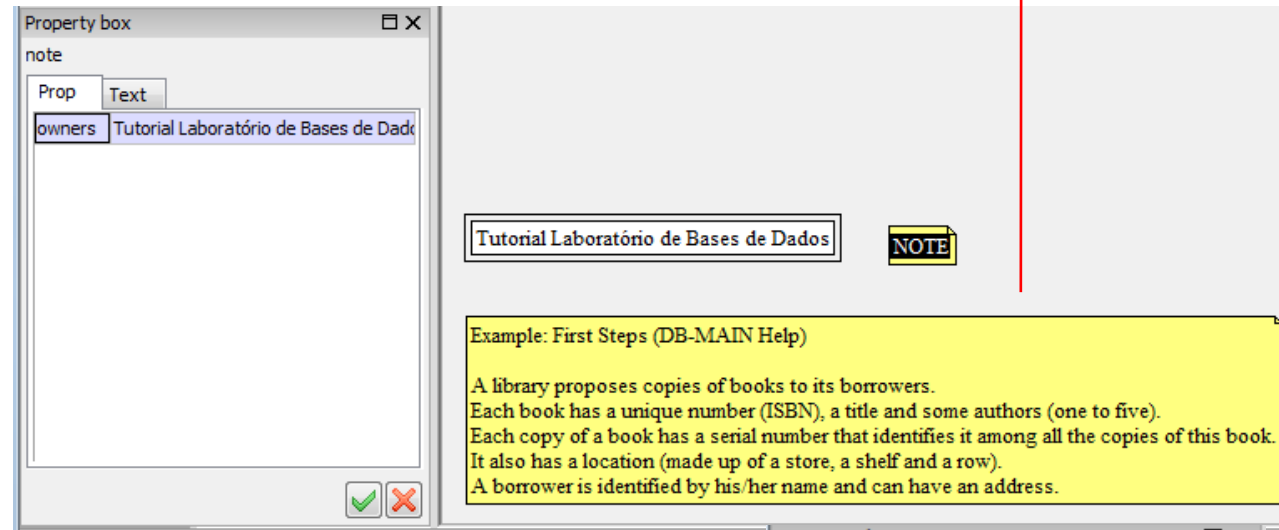


Object: Notes

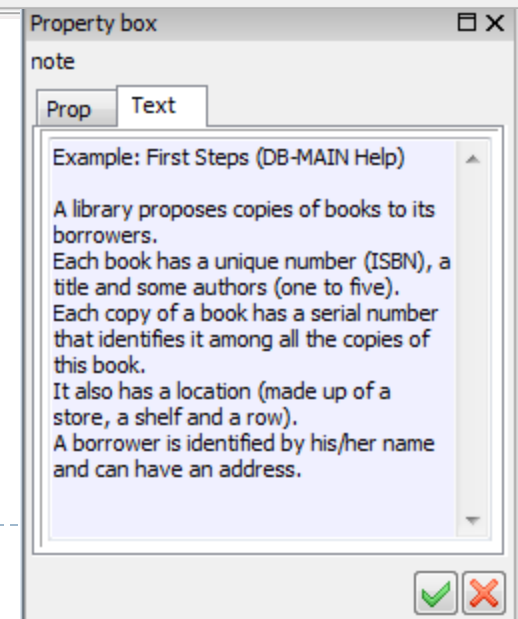
A note is a kind of post-it that can be inserted in a schema or attached to some objects of a schema.

The links between a note and its owners are represented by dotted lines, except for those associated to schemas.

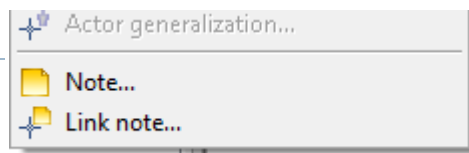
Simplified requirements system.



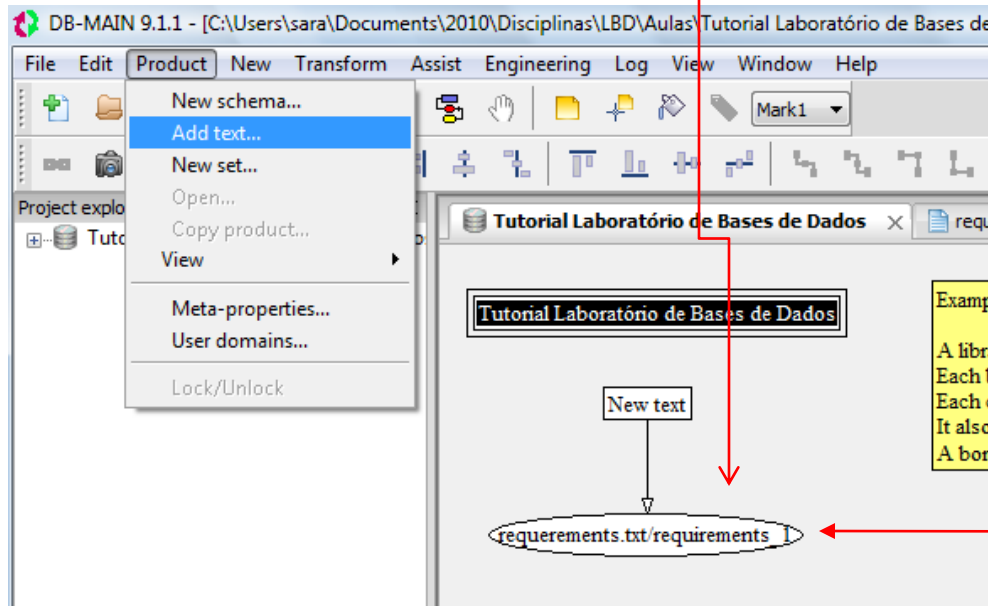
Use in your project in order to improve the specification legibility !!



No menu NEW.

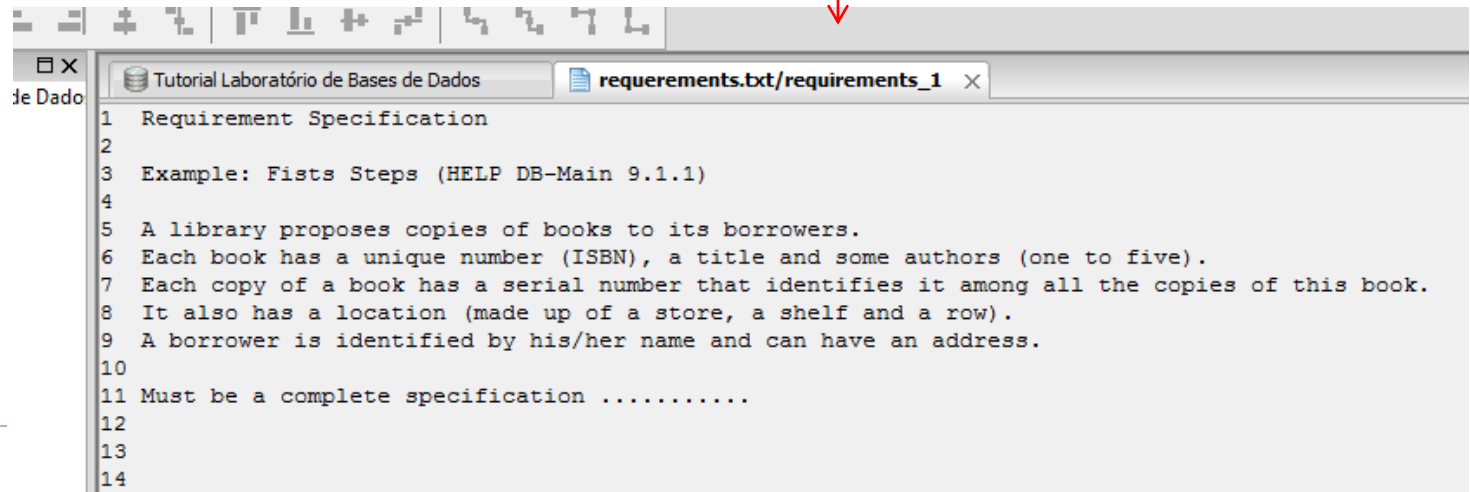


Inserting Text Information

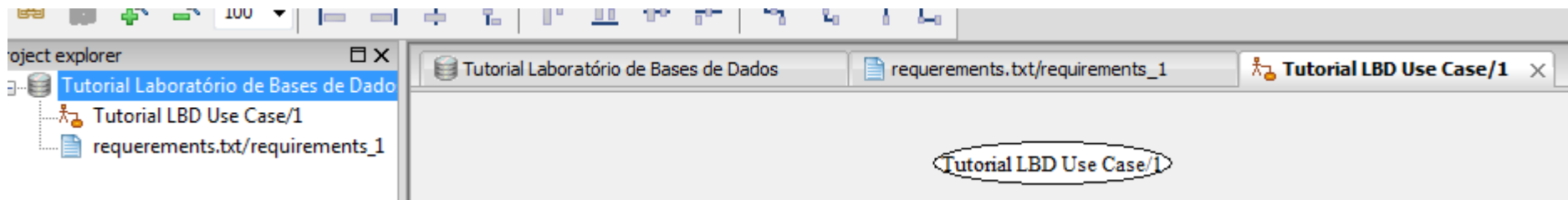
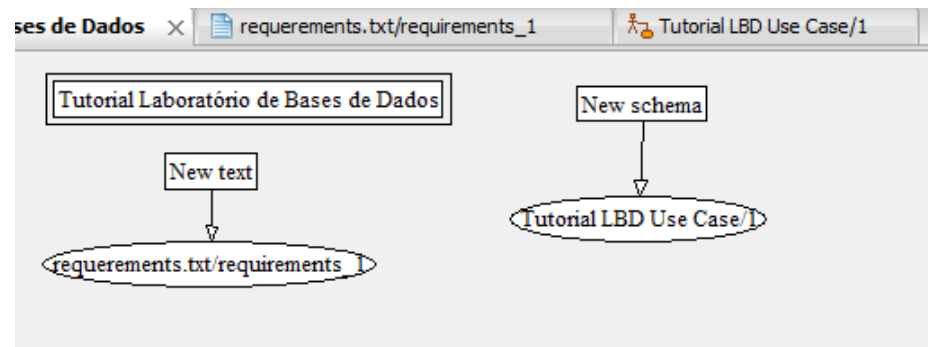
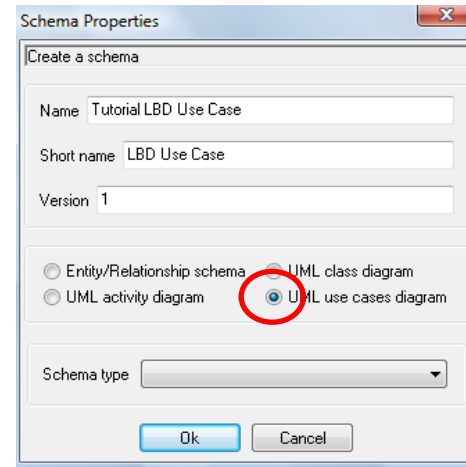
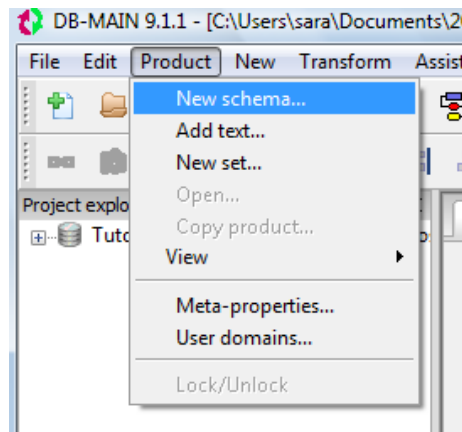


System documentation!!

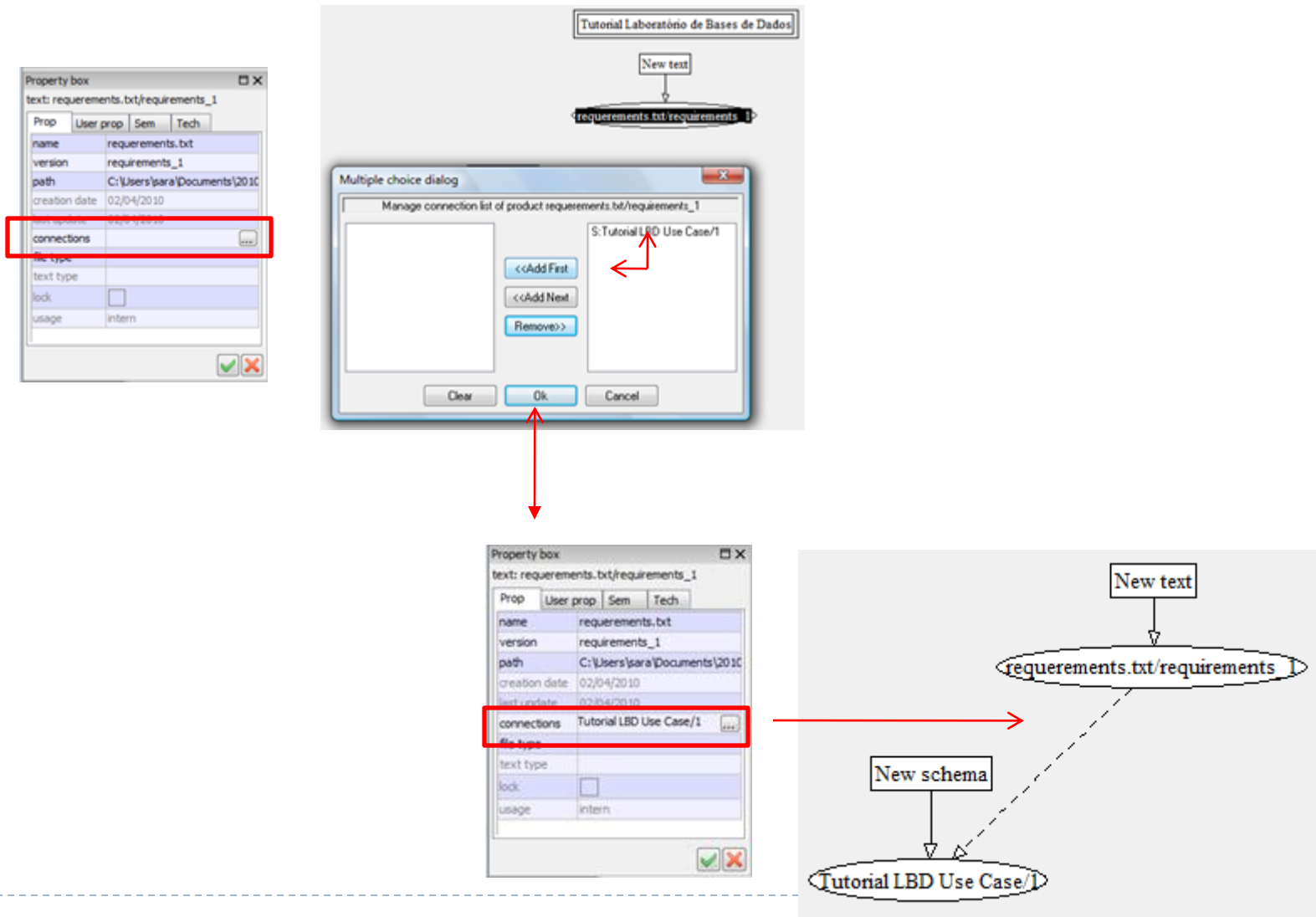
Double Click!



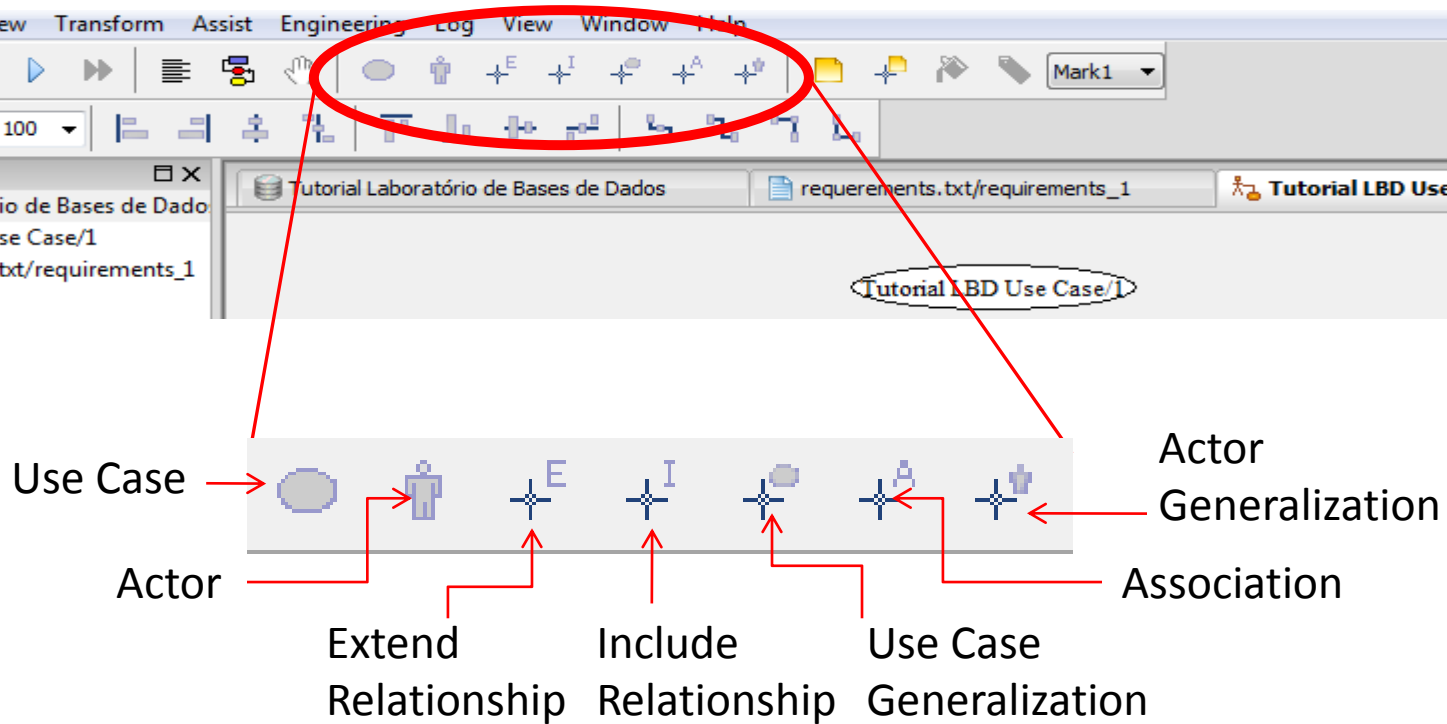
Building a Use Case Diagram



Linking the new schema to other object



Use Case – Toolbar



Use Case - Definitions

- ▶ *Use Case*: A use case is a kind of classifier representing a coherent unit of functionality provided by a system, a subsystem, or a class.
 - ▶ a *use case generalization* from a use case *A* to a use case *B* indicates that *A* is a specialization of *B* (is a kind of *relation*).
- ▶ *Actor*: an actor represents a coherent set of roles that users can play when interacting with a system. An actor materializes any resource (man, machine, ...) that can be associated with an action.
 - ▶ an *actor generalization* from an actor *A* to an actor *B* indicates that an instance of *A* can communicate with the same kinds of use-case instances as can do an instance of *B* (is a kind of *relation*).

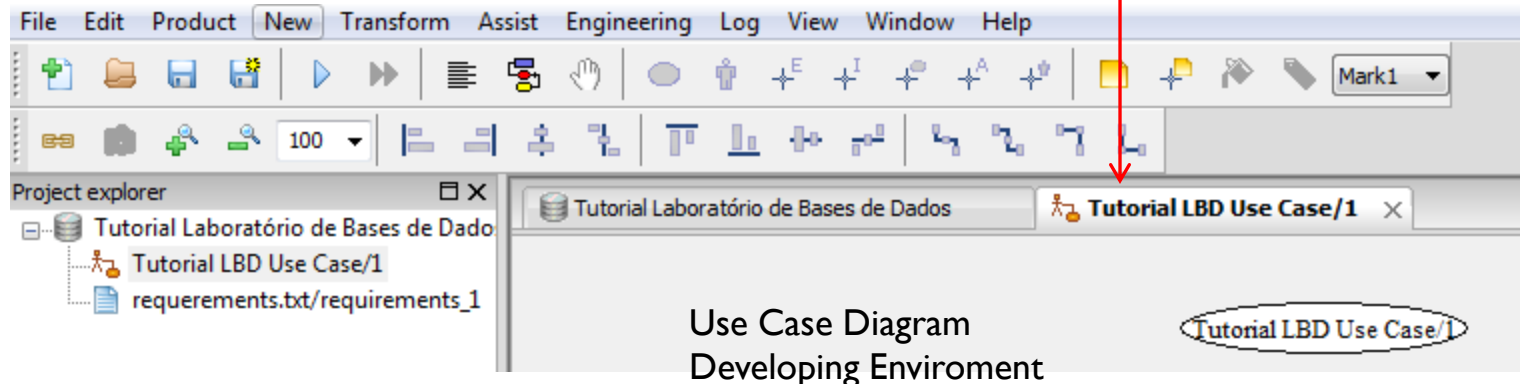
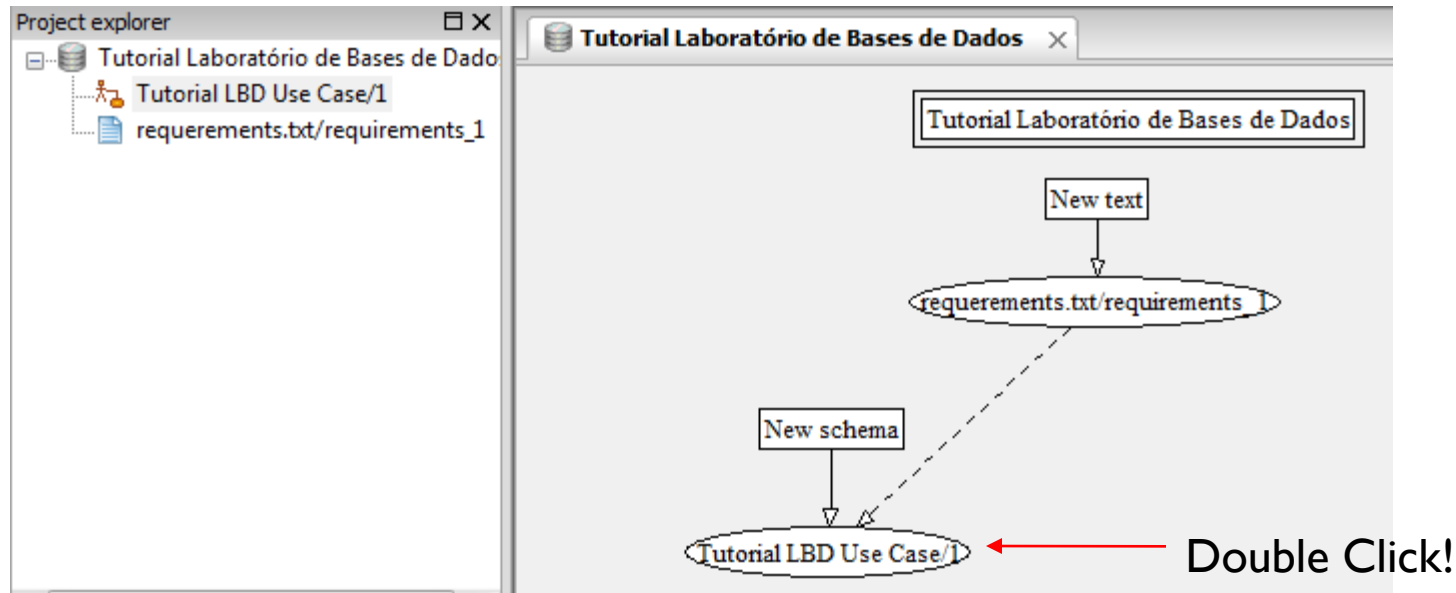


Use Case - Definitions

- ▶ These are also kinds of relations:
 - ▶ an *extend relationship* from a use case *A* to a use case *B* indicates that an instance of *B* may be augmented by the behavior specified by *A*;
 - ▶ an *include relationship* from a use case *A* to a use case *B* indicates that an instance of *A* will also contain the behavior specified by *B*;
 - ▶ an *association* between a use case and an actor indicates the participation of an actor in a use case.



Use Case – building the diagram



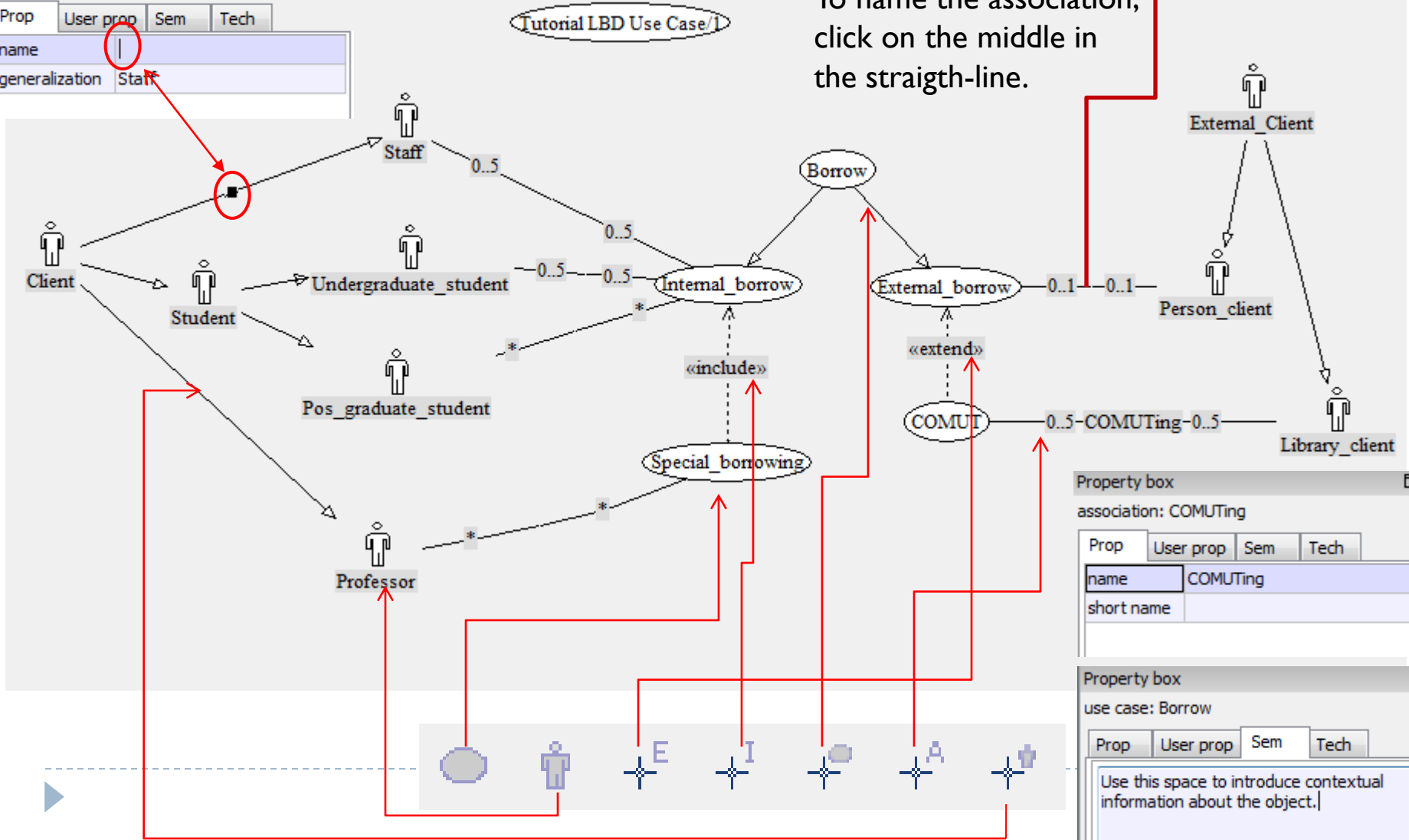
Use Case - elements

Property box

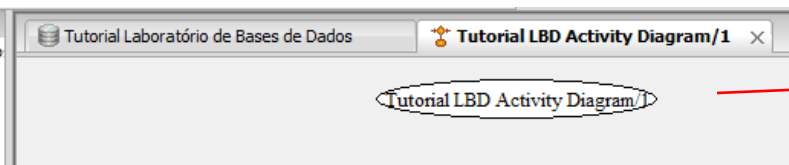
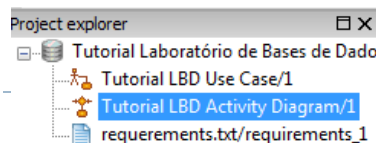
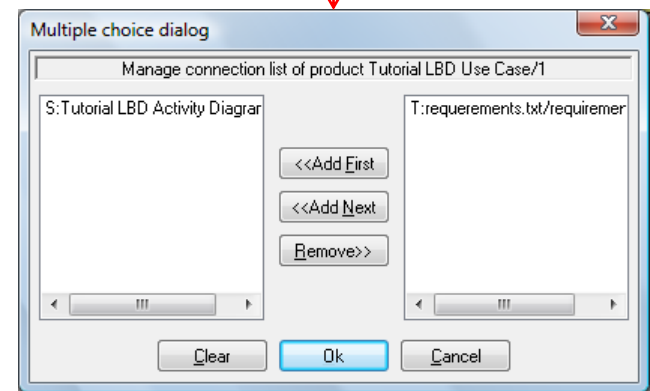
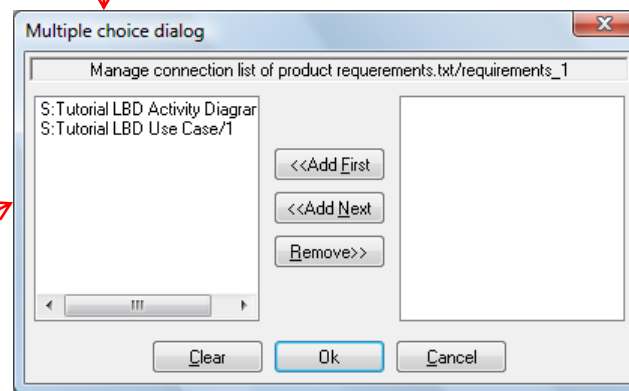
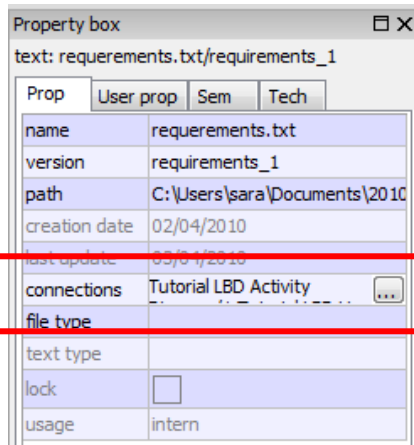
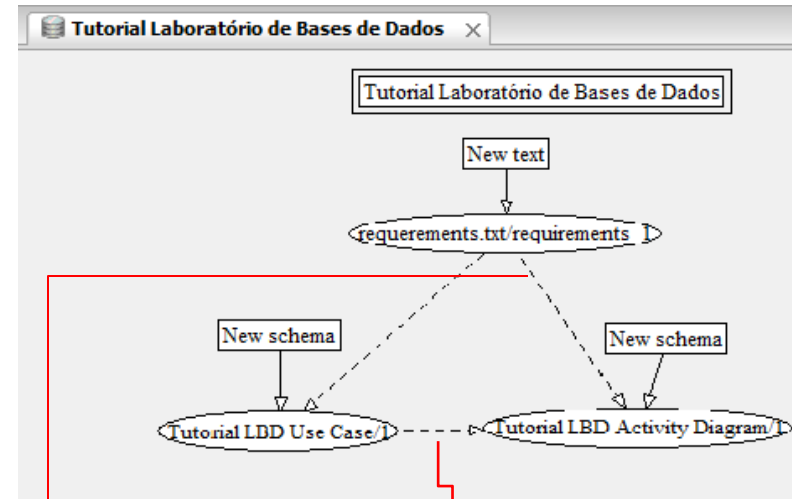
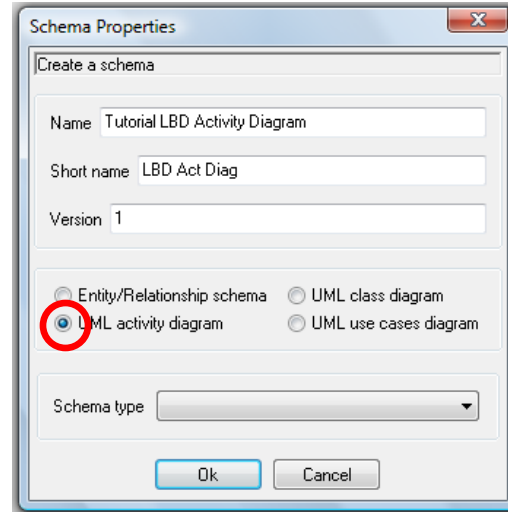
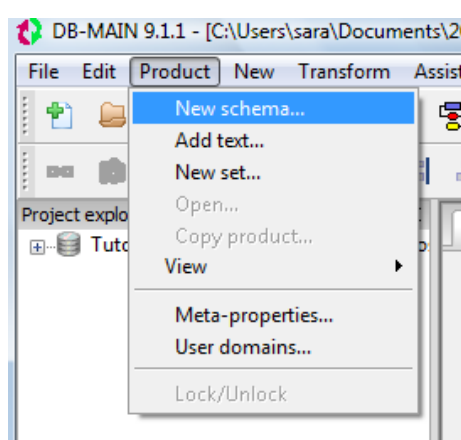
actor generalization:

Prop	User prop	Sem	Tech
name			
generalization	Staff		

To name the association, click on the middle in the straight-line.

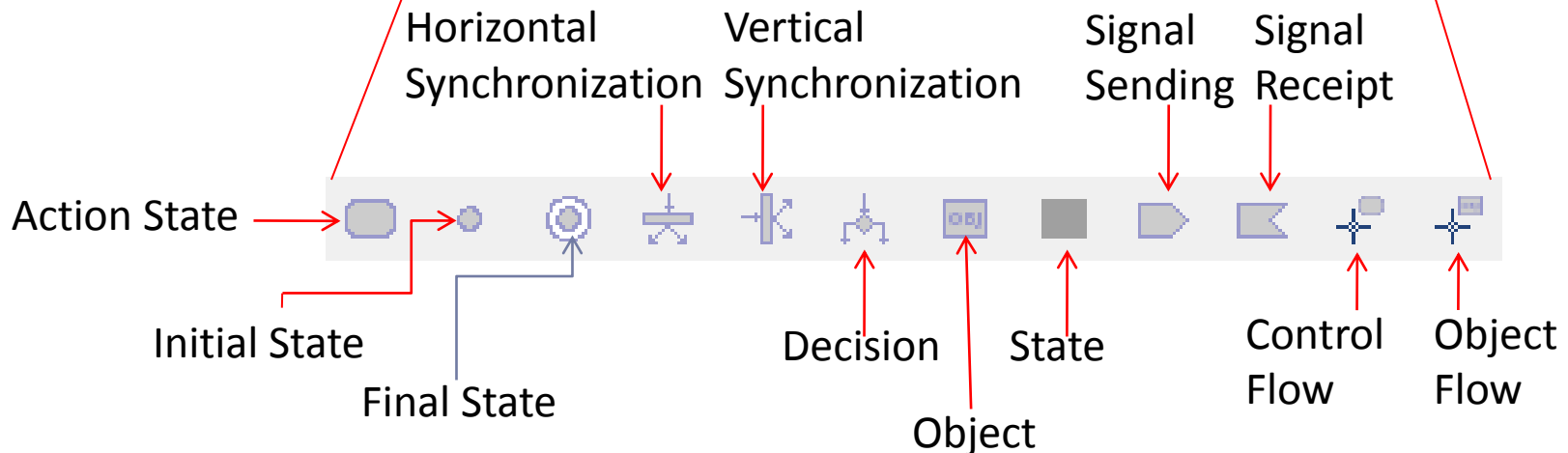
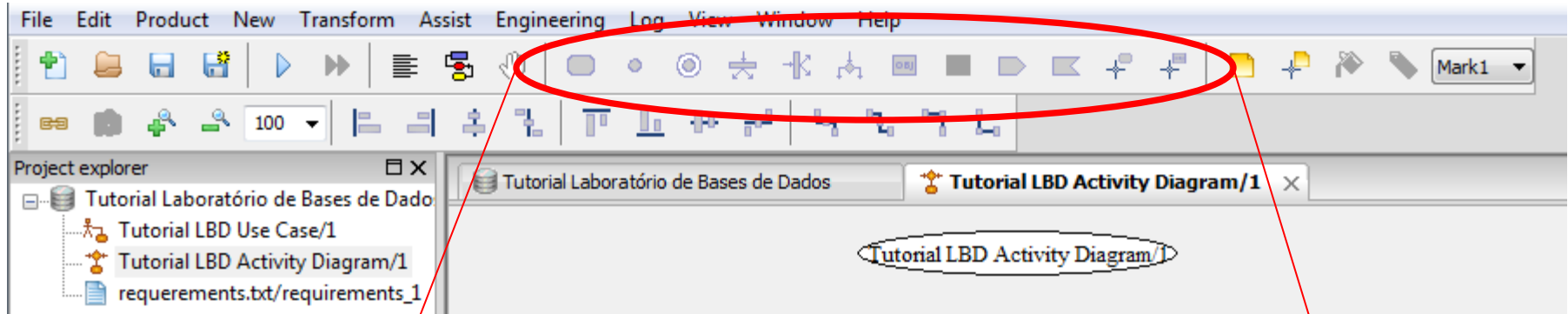


Building an Activity Diagram



Activity Diagram
Developing Environment

Activity Diagram – Toolbar



Activity Diagram - definitions

- ▶ *Objects* are used as input or output of action states.
 - ▶ an *internal object* can be a data type, a variable, a constant or any object that is known by the action states of the schema but that is unknown outside;
 - ▶ an *external object* is defined in a data schema and used in a UML Activity Diagram. Such is the case of entity types, attributes, collections or rel-types.
- ▶ *State* (or object state) is a picture of an object at a precise time. In fact, an object that can be transformed during the process described by the activity diagram can be in a different state before and after one of the actions.
 - ▶ For instance, a glass can be empty before the action of filling it, and full after that action. Since only the empty glass can be put in a cupboard and only the full glass can be drunk, it is important, for other actions, to distinguish the various states of the object.



Activity Diagram - definitions

- ▶ *Action state* is a shorthand for a state with an entry action and at least one outgoing transition involving the implicit event of completing the entry action.
 - ▶ *Initial state* is a special kind of action state that represents the beginning of an activity diagram.
 - ▶ *Final state* is a special kind of action state that represents the completion of an activity diagram.
 - ▶
 - ▶ *Synchronization state*, either horizontal or vertical allows to synchronize concurrent regions. It is used in conjunction with forks and joins to insure that one region leaves a particular state before another region can enter a particular state.
 - ▶ *Decision state* is expressed when conditions are used to indicate various possible transitions that depend on boolean conditions.
-

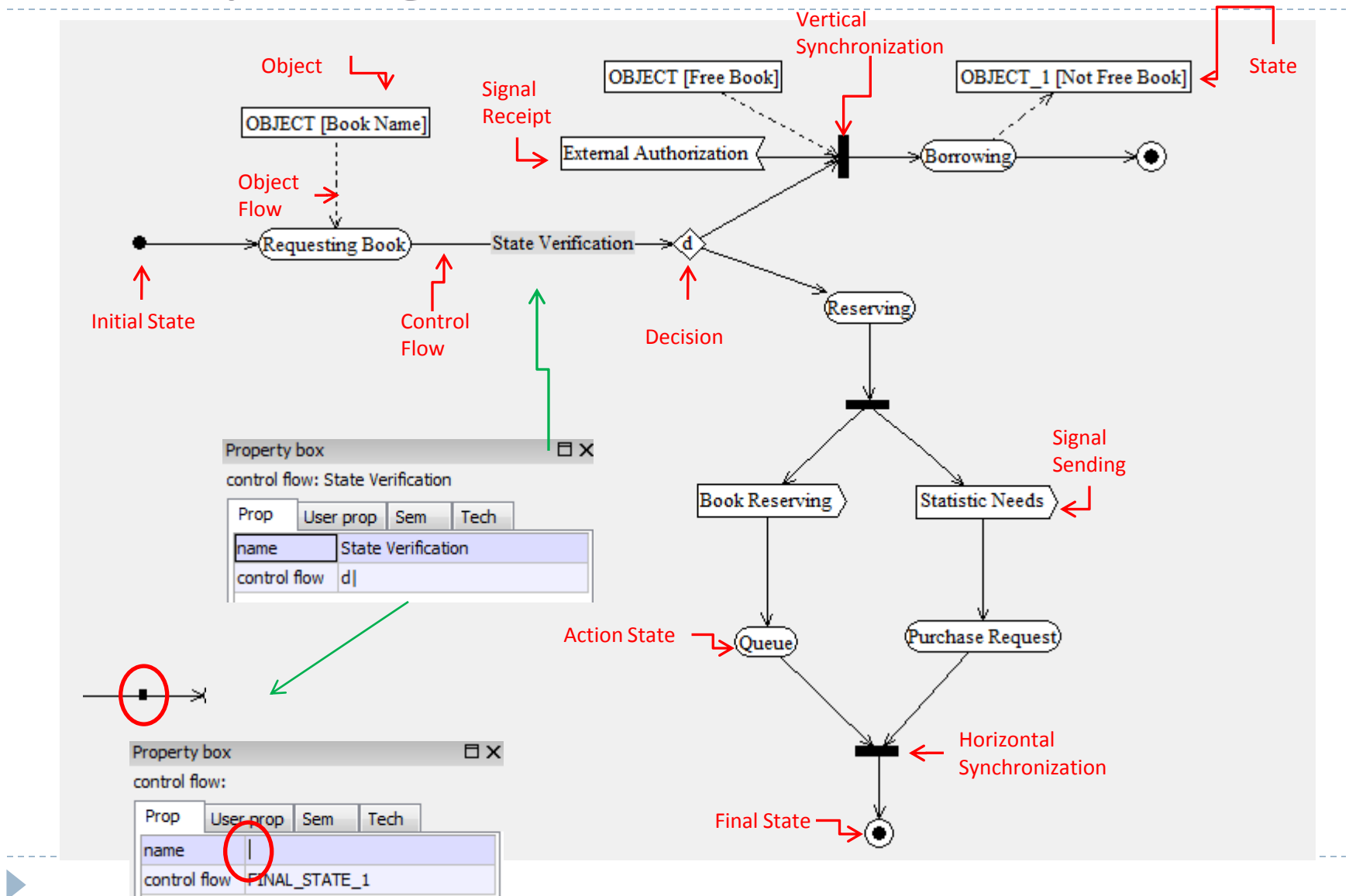


Activity Diagram - definitions

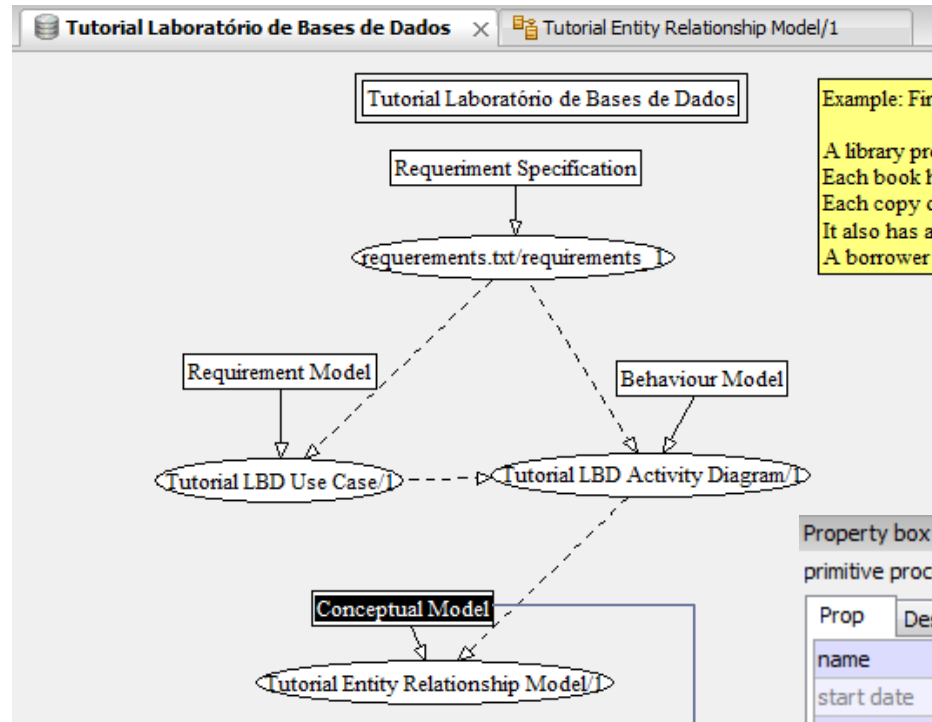
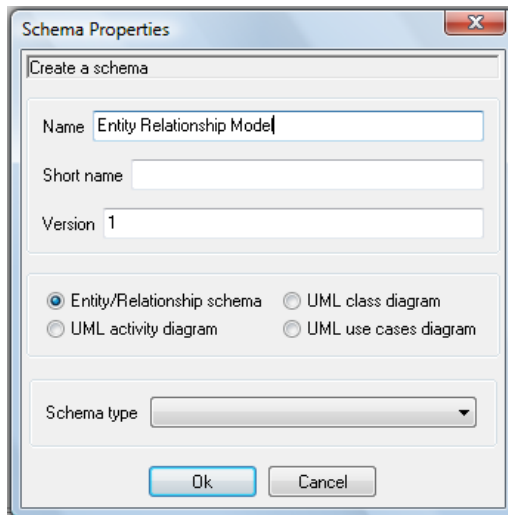
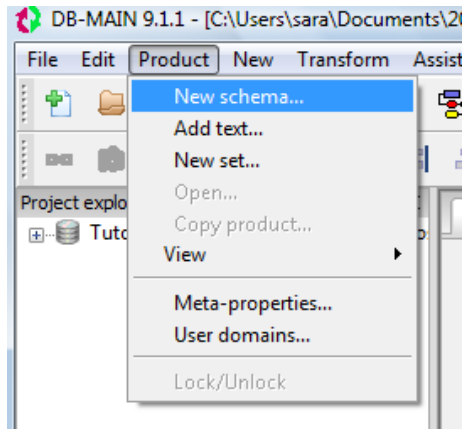
- ▶ *Signal sending* shows a transition sending a signal.
- ▶ *Signal receipt* shows a transition receiving a signal.
- ▶ *Control Flow*: to open a dialog box for selecting the action state(s) implied in a control flow;
 - ▶ a *control flow* indicates the order of execution of action states;
- ▶ *Object Flow*: to open a dialog box for selecting the object(s) implied in a object flow;
 - ▶ an *object flow* indicates input object states or output objects (internal or external) of an action state;



Activity Diagram - elements



Building an Entity-Relationship Model

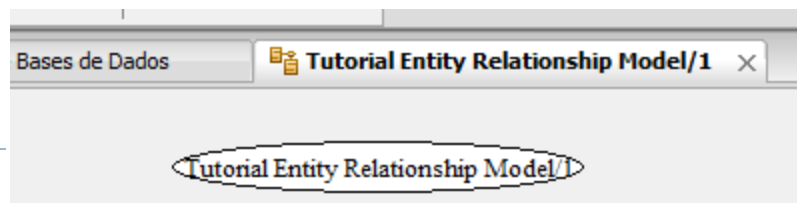


Labeling the schema

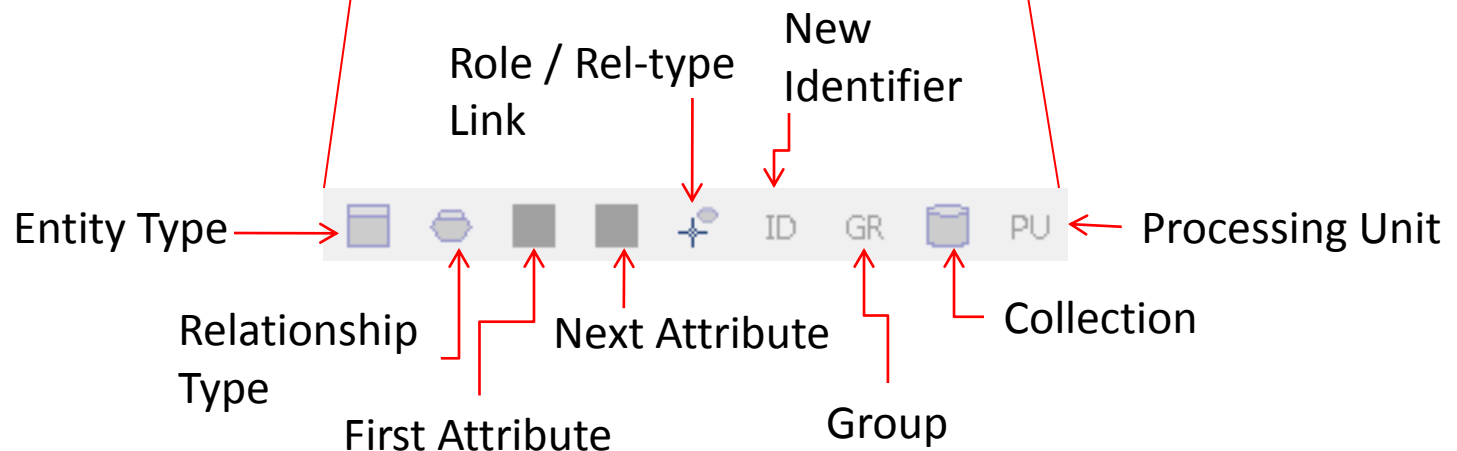
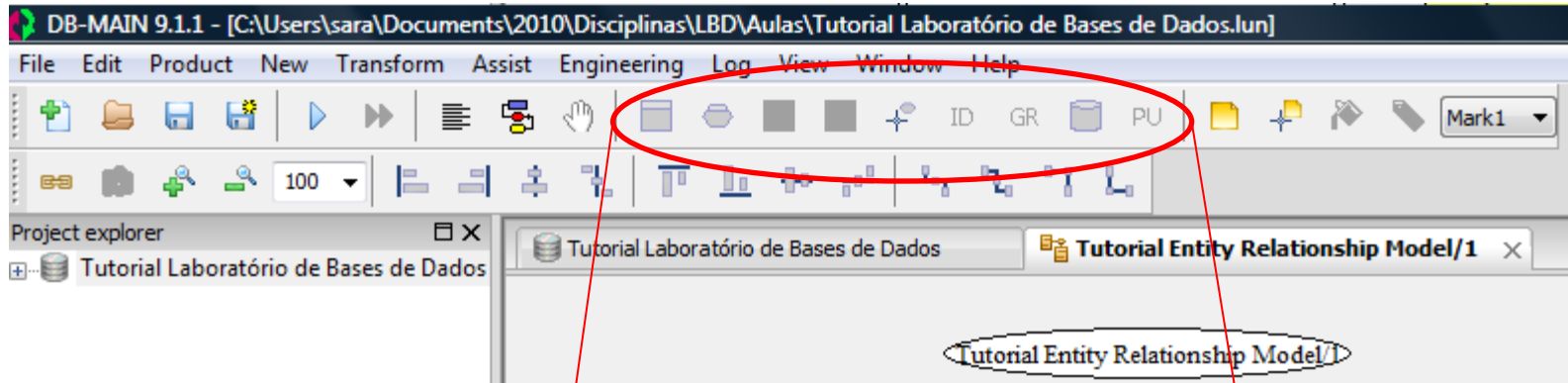
Property box

primitive process: Conceptual Model

Prop	Desc
name	Conceptual Model
start date	03/04/2010
start time	22:24:25
end date	03/04/2010
end time	22:24:25
in progress	<input type="checkbox"/>
products	Tutorial Entity Relationship Model/1



Entity Relationship Model – Toolbar



Entity Relationship Model - definitions

- ▶ *Entity type* materializes a class of entities that represent objects. These objects can be real world abstract or concrete entities. They can also be abstract or concrete data structures, such as records, tuples or segments. An entity type can have any number (including zero) of attributes.
 - ▶ An entity type can be a subtype of one or several other entity types.
- ▶ *Rel-type* (relationship type) represents a class of associations between entities. It has two or more roles and any number (including zero) of attributes.
 - ▶ A role is the partner of an entity type.



Entity Relationship Model - definitions

- ▶ An attribute represents a property of entities or associations of the same type. It is either atomic or compound; an atomic attribute has a domain of values; each attribute is subject to a cardinality constraint [min-max]. This constraint allows to specify optional/mandatory (min = 0 or 1) attributes as well as single-valued/multivalued (max = 1 or > 1) attributes.
- ▶ The possible domains of values are listed below:
 - ▶ *boolean; char; compound; date; float; index; numeric; sequence; varchar; object type;*
 - ▶ *user-defined*: can be atomic or compound, it can be associated with several attributes (in the attribute properties dialog box, select user-defined type in the *type* combo-box and then the user-defined domain in the new combo-box). A user-defined domain is defined for the current project.



Entity Relationship Model - definitions

- ▶ A multivalued attribute has a collection type. The possible collection types are listed below:
 - ▶ *set*: the values of the attribute are distinct and there is no ordering relation between them.
 - ▶ *bag*: the values are not necessarily distinct and there is no ordering relation between them.
 - ▶ *unique list*: the values are distinct and ordered.
 - ▶ *list*: the values are not necessarily distinct but they are ordered.
 - ▶ *unique array*: the values are distinct and ordered. Each value is stored into a cell and a cell can be empty.
 - ▶ *array*: the values are not necessarily distinct but they are ordered. Each value is stored into a cell and a cell can be empty.



Entity Relationship Model - definitions

- ▶ *Role* is a place holder in a rel-type. It is played by one or several entity types (mono-ET role or multi-ET role) and is given a cardinality constraint that states the minimum and maximum number times connected entities can play this role.
- ▶ *Group* is associated to a parent object (entity type, rel-type or multivalued compound attribute). A group is a set of attributes and/or roles and/or other groups that play some functions together for the parent object. The possible functions of a group include: identifier, coexistence, exclusive, at-least-one, user constraint and access key.

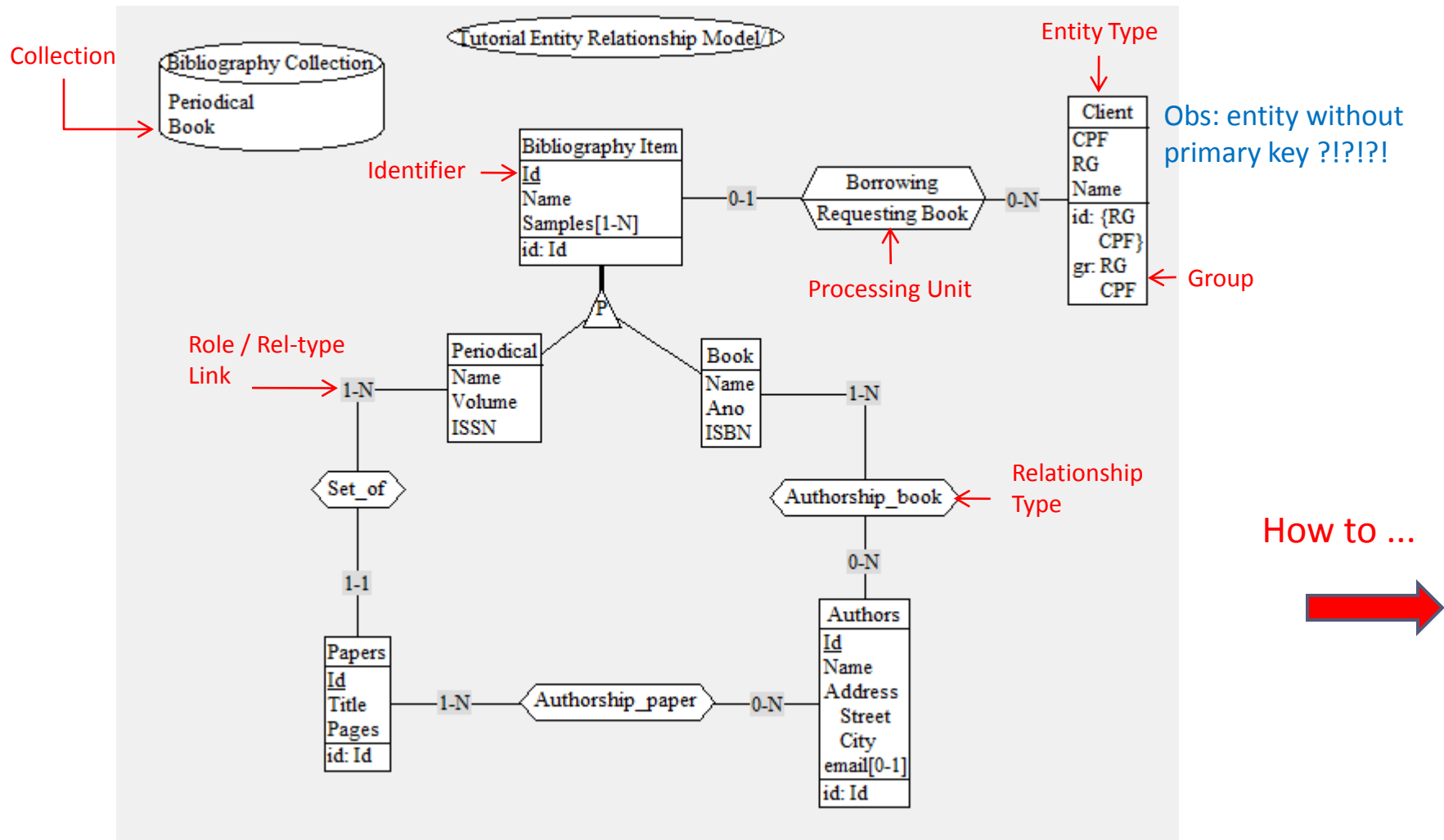


Entity Relationship Model - definitions

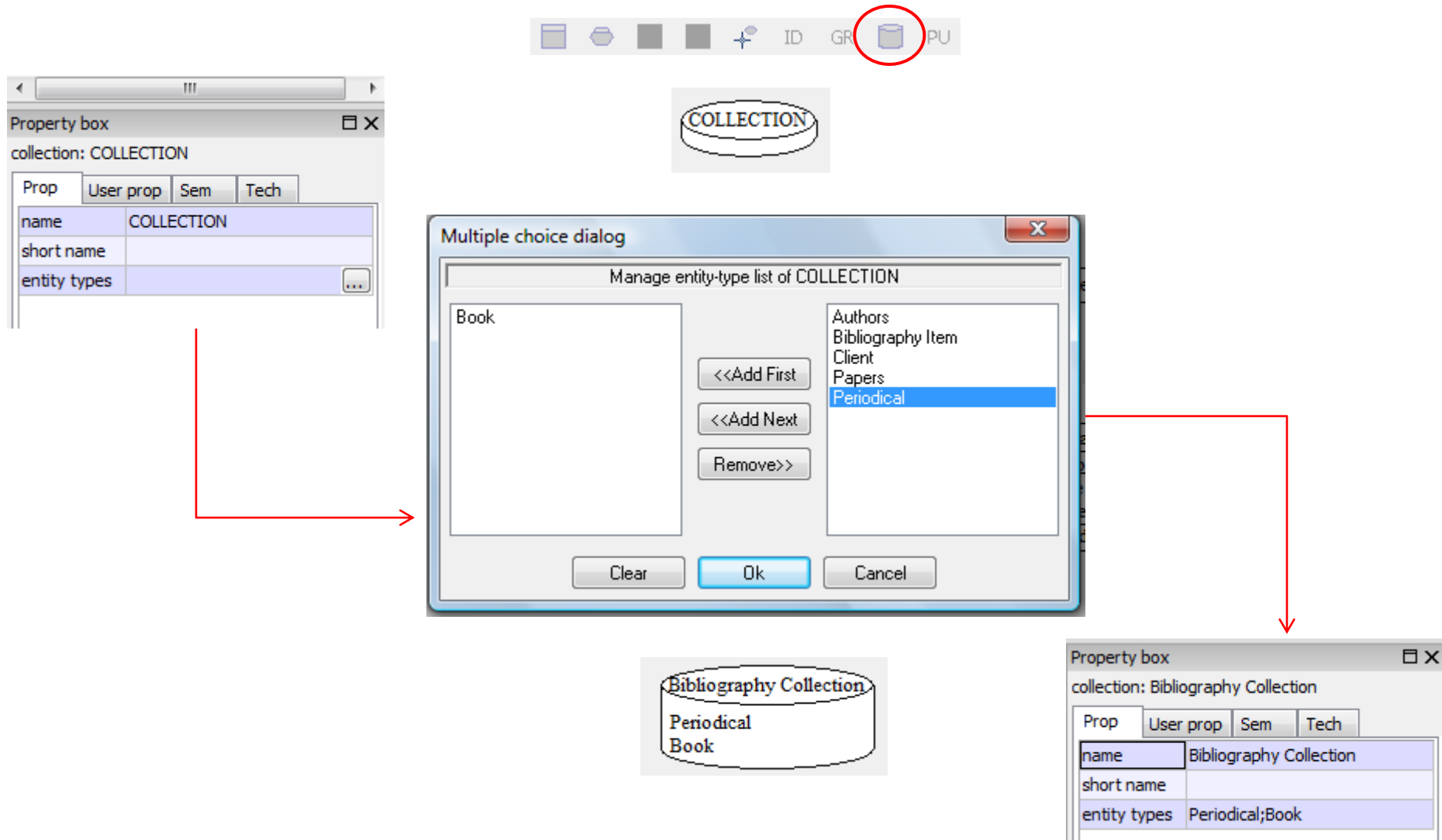
- ▶ *Collection* is a repository of entity types. In logical and physical schemas, collections can be used to represent files and the like.
- ▶ In a data schema (ER or UML Class Diagram), an *anchored processing unit* is any dynamic or logical component of the described system that can be associated with a schema, an entity-type or a relationship type.



Entity Relationship Model - elements



Entity Relationship Model – how to



Entity Relationship Model – how to

The screenshot displays an Entity Relationship Model software interface. At the top center is a menu bar with 'View', 'Window', and 'Help'. Below it is a toolbar with icons for saving, opening, adding, and deleting, with the 'ID' icon circled in red. The central diagram shows a 'Bibliography Item' entity with attributes 'Id_bib' (underlined), 'Name', 'Samples[1-N]', and 'id: Id_bib'. A red arrow points from the 'ID' icon to 'Id_bib'. Below 'Bibliography Item' is a partition symbol 'P' with two sub-entities: 'Periodical' (attributes: Name, Volume, ISSN) and 'Book' (attributes: Name, Ano, ISBN). Red arrows point from the 'total' and 'disjoint' checkboxes in the 'Bibliography Item' property box to the partition symbol 'P'. To the left of the central diagram are two 'Property box' windows. The top one is for 'Bibliography Item' with 'total' and 'disjoint' checked. The bottom one is for 'Periodical' with 'Bibliography Item' listed under 'supertypes'. A red arrow points from the 'supertypes' list to the 'Multiple choice dialog' window at the bottom left. To the right of the central diagram is another 'Property box' window for 'Bibliography Item.Samples' with 'cardinality' set to '1-N' and 'collection type' set to 'set'. Red arrows point from the 'Samples[1-N]' attribute in the central diagram to the 'cardinality' and 'collection type' fields in this property box.

Property box
entity type: Bibliography Item

Prop	User prop	Sem	Tech
name	Bibliography Item		
short name			
total	<input checked="" type="checkbox"/>		
disjoint	<input checked="" type="checkbox"/>		
supertypes			
length	N		

Property box
entity type: Periodical

Prop	User prop	Sem	Tech
name	Periodical		
short name			
total	<input type="checkbox"/>		
disjoint	<input type="checkbox"/>		
supertypes	Bibliography Item		
length	N		

Multiple choice dialog
Manage supertype list of Periodical

Bibliography Item	Authors
	Book
	Client
	Papers

Property box
attribute: Bibliography Item.Samples

Prop	User prop	Sem	Tech
name	Samples		
short name			
cardinality	1-N		
collection type	set		
type	varchar		
length	1		
decimal length			
domain			
stable	<input type="checkbox"/>		
non recyclable	<input type="checkbox"/>		

Bibliography Item

Id_bib
Name
Samples[1-N]
id: Id_bib

Periodical

Name
Volume
ISSN

Book

Name
Ano
ISBN

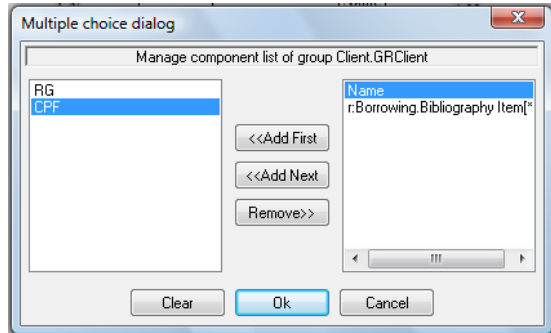
P

If the cluster has the disjoint property, the new group is submitted to the exclusive constraint.

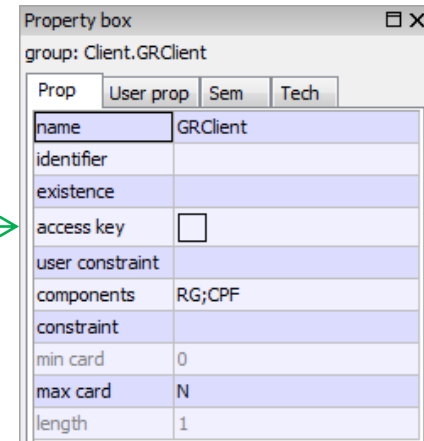
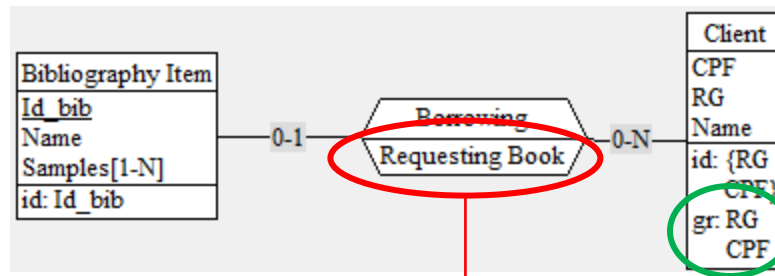
If the cluster has the total property, the new group is submitted to the at-least-one constraint.

If the cluster has the partition property, the new group is submitted to both the exclusive and the at-least-one constraints (i.e. the exactly-one constraint).

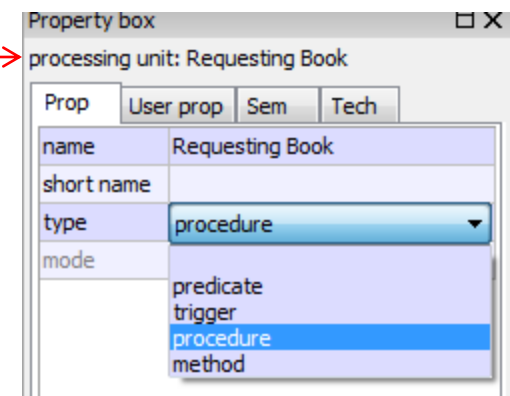
Entity Relationship Model – how to



Choose an attribute or a relationship type and create the group.

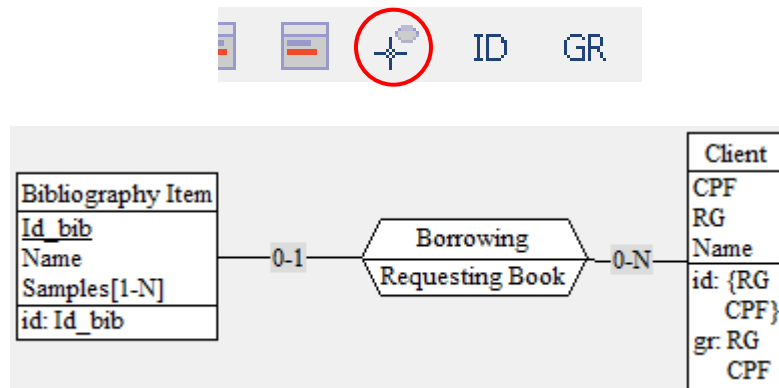


Choose an entity or a relationship type and associate the unit processing.



Entity Relationship Model – how to

Link/association between entity and relationship types.



Property box

role: Borrowing.Bibliography Item

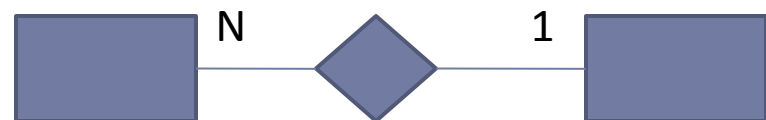
Prop	User prop	Sem	Tech
name			
cardinality	0-1		
entity types	0-1		
aggregation	1-1		
	0-N		
	1-N		
	0-5		
	1-5		
	5-5		

Reading:

A bibliography item can appear 0 or 1 time in the borrow event (relationship), i.e., a bibliography item can be borrowed or not.

A client can appear 0 or N times in the borrow event (relationship), i.e., a client can borrow some (or several) items or not.

Classic view



Entity Relationship Model – how to

Property box

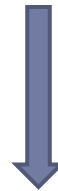
attribute: Authors.Address

Prop	User prop	Sem	Tech
name	Address		
short name			
cardinality	1-1		
collection type			
length	51		

Authors	
<u>Id_author</u>	
Name	
Address	
Street	
City	
email[0-1]	
id: Id_author	



Mark the parent attribute and choose, in the toolbar, the option “First Attribute”.



Property box

attribute: Authors.Address.Street

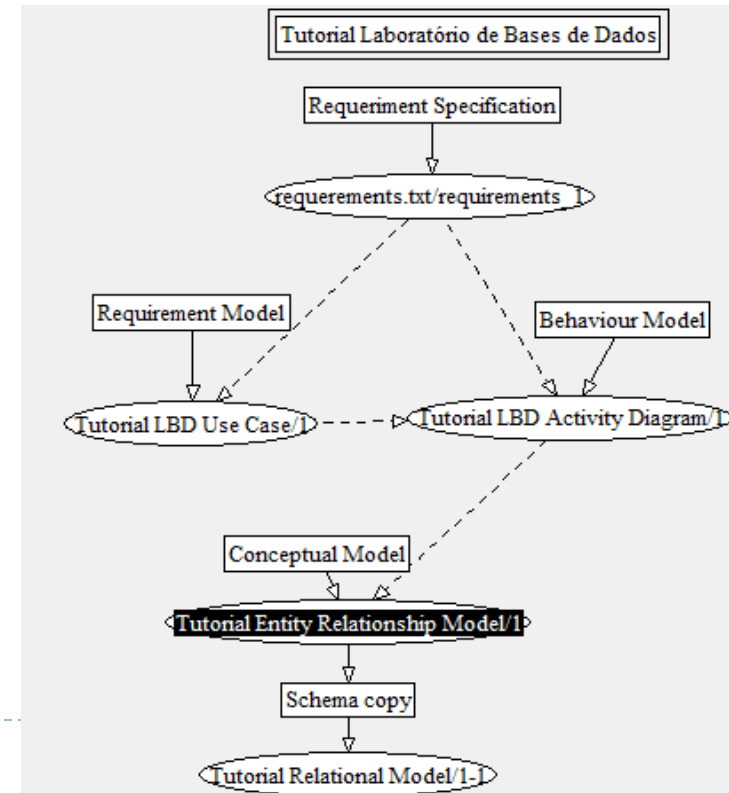
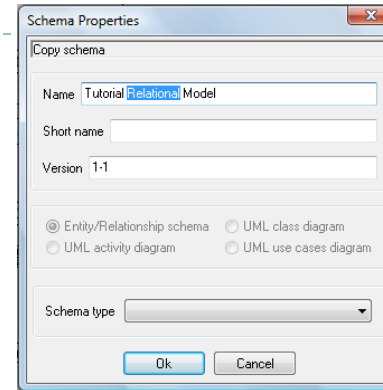
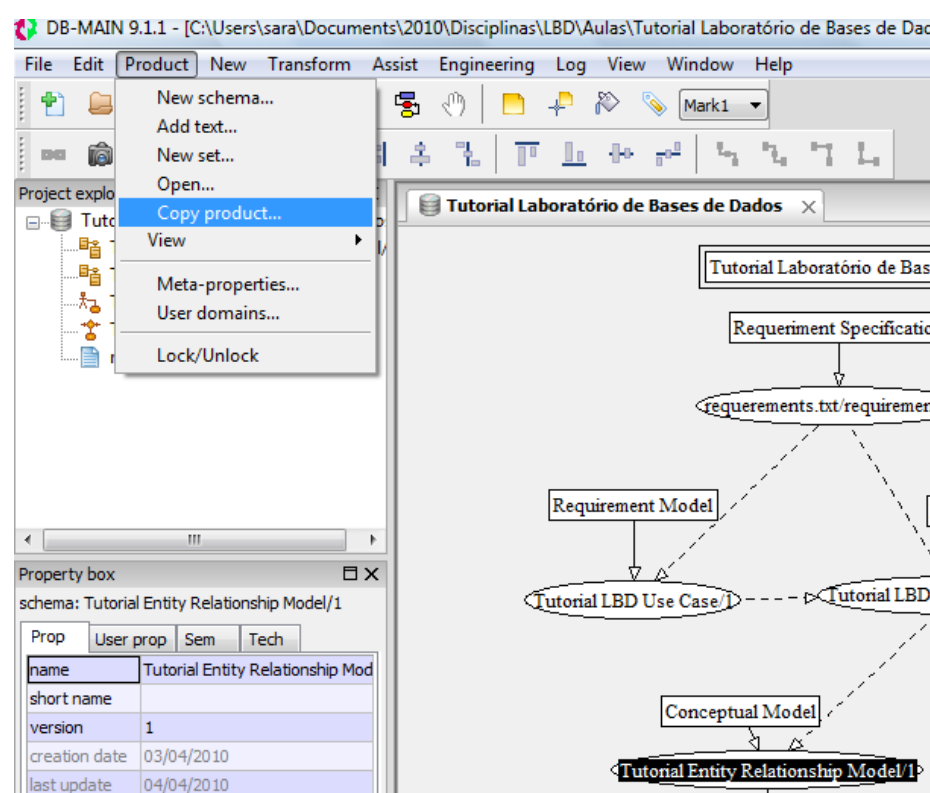
Prop	User prop	Sem	Tech
name	Street		
short name			
cardinality	1-1		
collection type			
type	varchar		
length	50		
decimal length			
domain			
stable	<input type="checkbox"/>		
non recyclable	<input type="checkbox"/>		

Authors	
<u>Id_author</u>	
Name	
Address	
Street	
City	
email[0-1]	
id: Id_author	



Mark the attribute and choose, in the toolbar, the option “Next Attribute”.

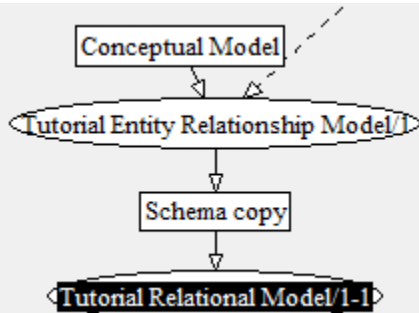
Building an Relation Model through an automatic mapping (1)



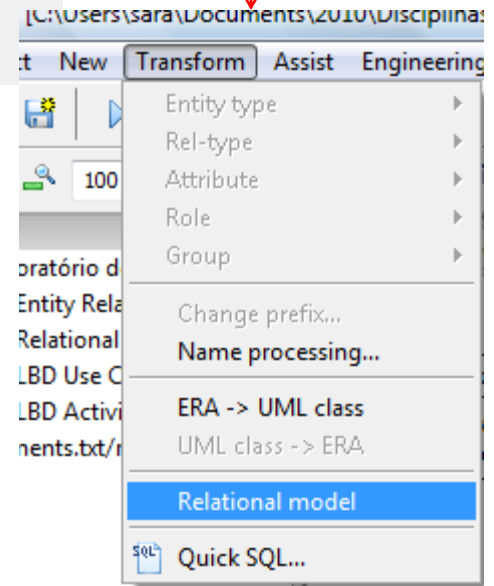
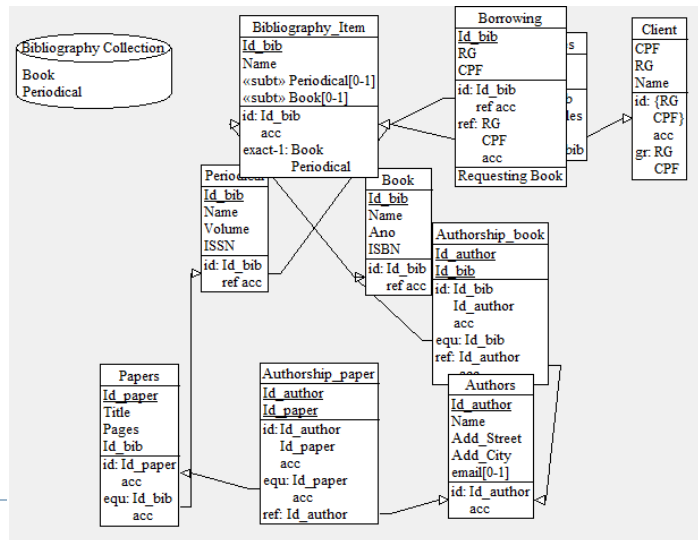
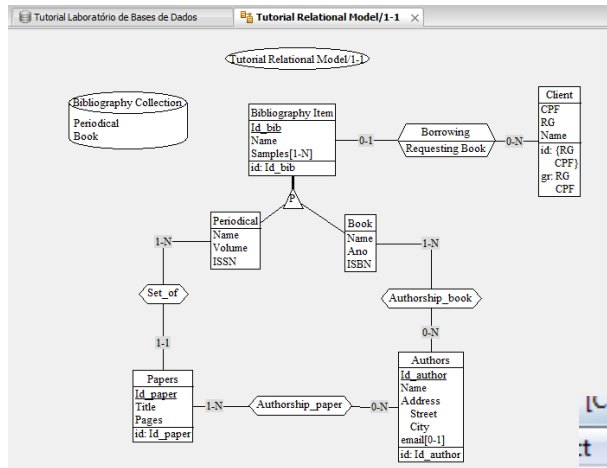
Do not do the mapping with the original ER model. The mapping will replace the ER model by the relational model. We cannot redo!!!!!!

Create a schema copy!!!!!!!!!!!!!!!!!!!!!!

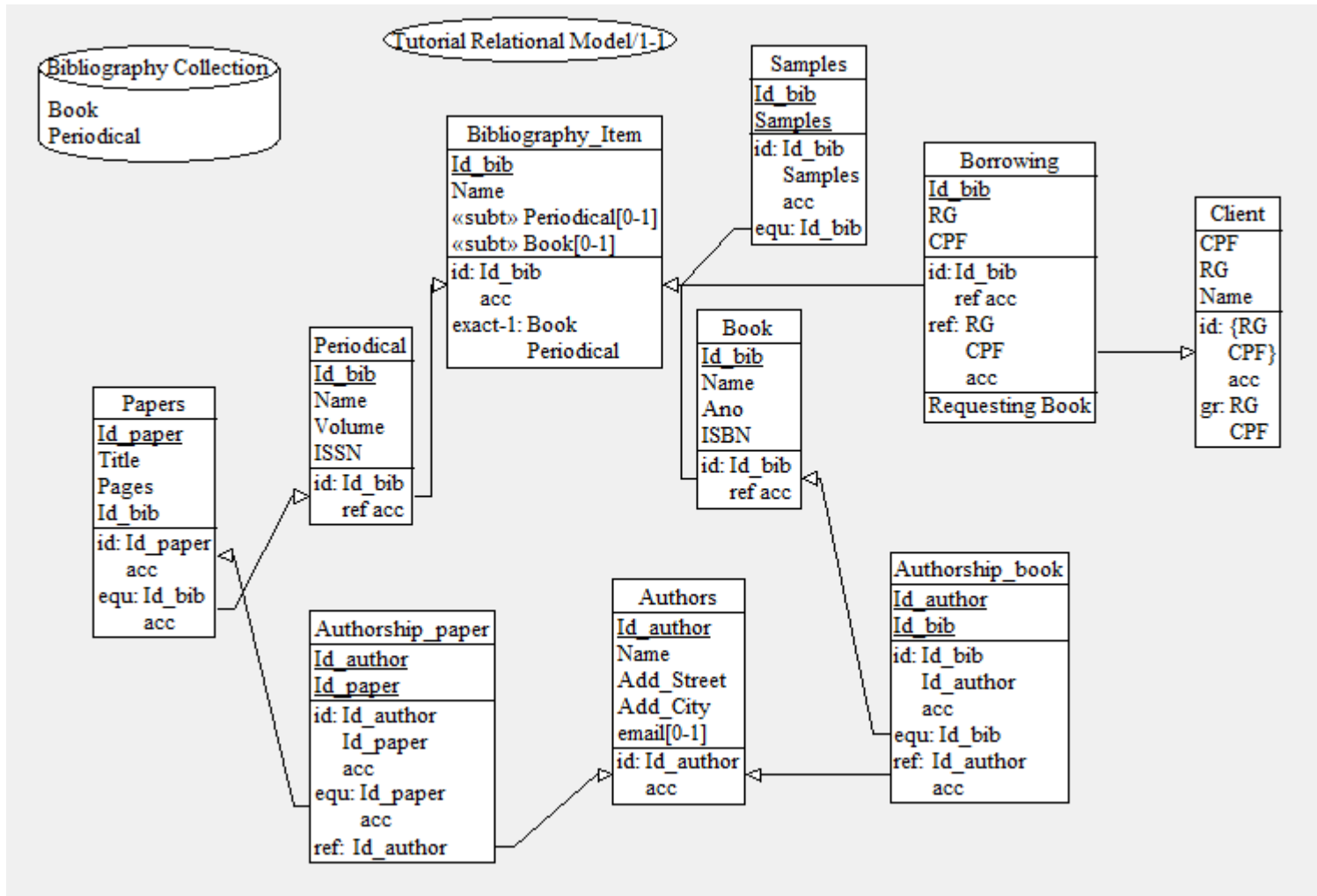
Building an Relation Model through an automatic mapping (2)



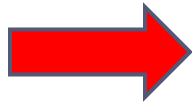
Double Click!



Building an Relation Model through an automatic mapping (3)



Analysing!!!!

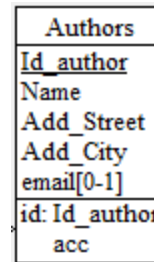
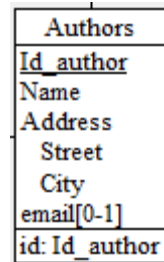


Relation Model -automatic mapping – analysing ...

ER Model

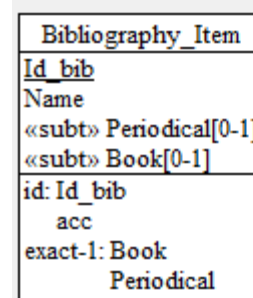
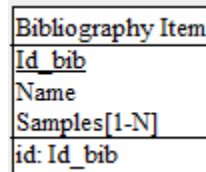
Relational Model

Compound Attribute



→ Primary key = index!

Multi-valored Attribute



Foreign key “equ” – total: All Bibliography items have associated Samples.

attribute: Bibliography Item.Samples

Prop	User prop	Sem	Tech
name	Samples		
short name			
cardinality	1-N		
collection type	set		
type	varchar		
length	1		

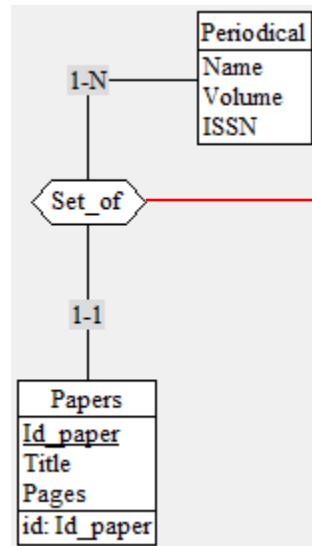
Property box
attribute: Samples.Samples

Prop	User prop	Sem	Tech
name	Samples		
short name			
cardinality	1-1		
collection type			
type	varchar		
length	1		

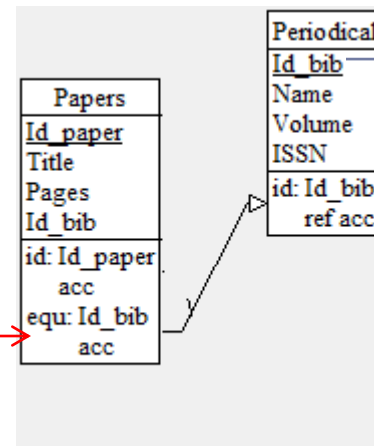
Relation Model -automatic mapping – analysing ...

Relationship 1-N

ER Model



Relational Model

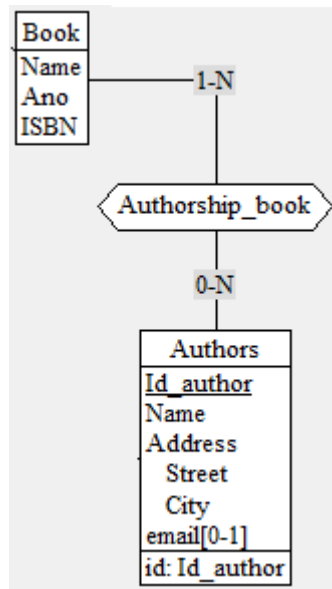


From
Generalization
Relationship.

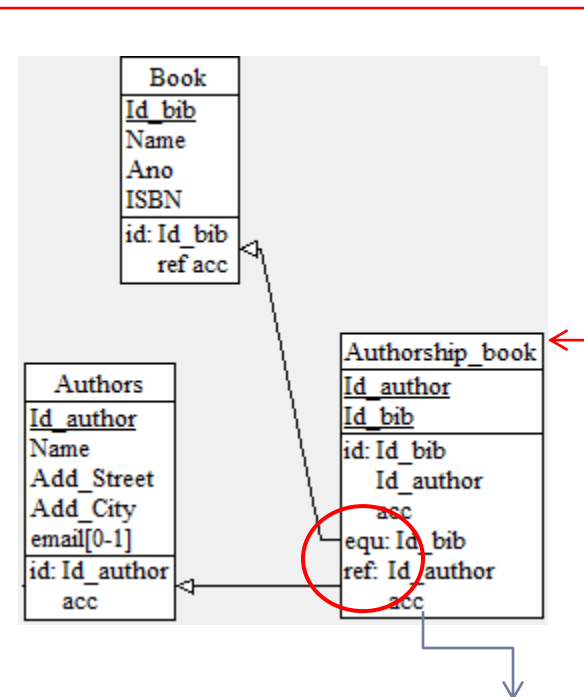
Relation Model -automatic mapping – analysing ...

Relationship N-M

ER Model



Relational Model



equ → 1-N → the participation is required.

ref → 0-N → the participation is not required

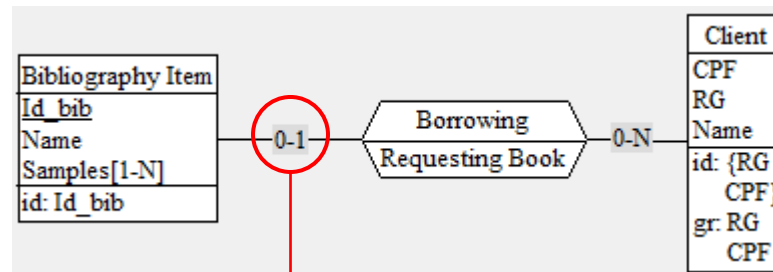
The symbol **acc** (for *access key*) is associated with each identifier and each foreign key.

Relation Model -automatic mapping – analysing ...

Relationship 1-N (0-N)

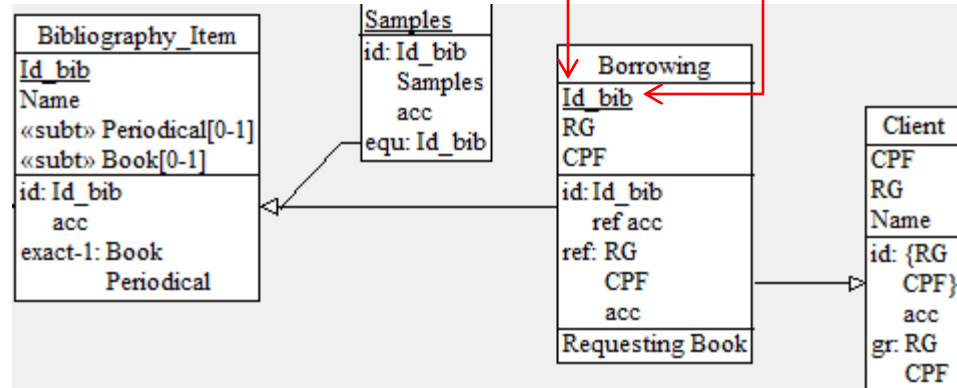
With specific feature – unit processing.

ER Model



The primary key is only one of the foreign keys.

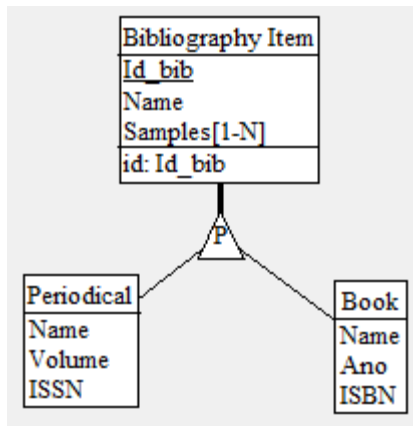
Relational Model



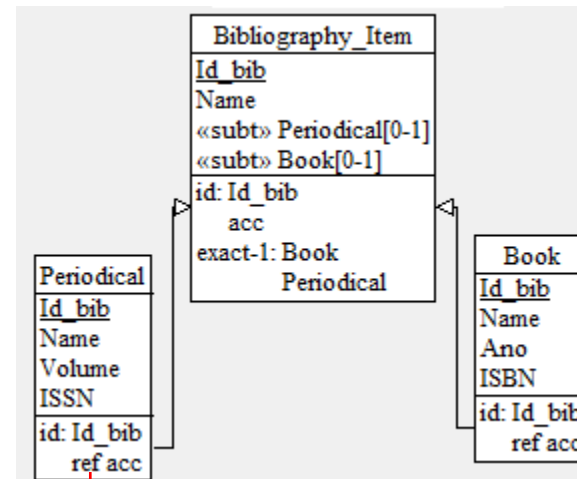
Relation Model -automatic mapping – analysing ...

Generalization - Specialization.

ER Model



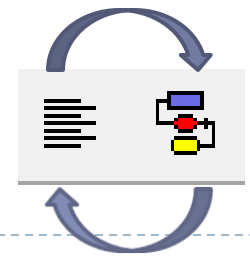
Relational Model



Ref: Foreign key – not total: Some Bibliography Items are Periodicals, others are Books.

If the cluster has the disjoint property, the new group is submitted to the exclusive constraint.
If the cluster has the total property, the new group is submitted to the at-least-one constraint.
If the cluster has the partition property, the new group is submitted to both the exclusive and the at-least-one constraints (i.e. the exactly-one constraint).

Text Standard



Schema Tutorial LBD Use Case/1

Borrow
 specialization of Internal_borrow
 specialization of External_borrow
External_borrow
Internal_borrow
COMUT
 extend to External_borrow
Special_borrowing
 include Internal_borrow

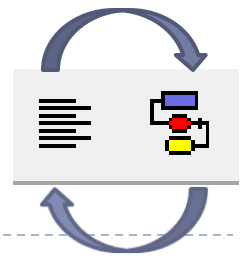
Client
 specialization of Staff
 specialization of Professor
 specialization of Student
External_Client
 specialization of Library_client
 specialization of Person_client
Library_client
Person_client
Pos_graduate_student
Professor
Staff
Student
 specialization of Undergraduate_student
 specialization of Pos_graduate_student
Undergraduate_student

COMUTing (
 [0..5]: COMUT

Schema Tutorial LBD Activity Diagram/1

INITIAL_STATE
 call Requesting Book
Requesting Book
 State Verification: call d
 in OBJECT [Book Name]
d
 call Reserving
 call SYNCHRONISATION_2
Borrowing
 call FINAL_STATE_1
 out OBJECT_1 [Not Free Book]
Reserving
 call SYNCHRONISATION
Book Reserving
 call Queue
Queue
 call SYNCHRONISATION_1
FINAL_STATE
SYNCHRONISATION
 call Book Reserving
 call Statistic Needs
Statistic Needs
 call Purchase Request
Purchase Request
 call SYNCHRONISATION_1
SYNCHRONISATION_1
 call FINAL_STATE
FINAL_STATE_1
External Authorization
 call SYNCHRONISATION_2

Text Standard



Schema Tutorial Entity Relationship Model/1

collection Bibliography Collection

Periodical

Book

Authors

Id_author

Name

Address

Street

City

email[0-1]

id: Id_author

Bibliography Item

Id_bib

Name

Samples[1-N]

id: Id_bib

Book

is-a Bibliography Item

Name

Ano

ISBN

Client

CPF

RG

Name

id: {RG, CPF}

group: RG, CPF

Papers

Id_paper

Title

Pages

Schema Tutorial Relational Model/1-1

collection Bibliography Collection

Book

Periodical

Authors

Id_author

Name

Add_Street

Add_City

email[0-1]

id: Id_author

access key

Authorship_book

Id_author

Id_bib

id: Id_bib, Id_author

access key

equ: Id_bib = Book.Id_bib

ref: Id_author -> Authors.Id_author

access key

Authorship_paper

Id_author

Id_paper

id: Id_author, Id_paper

access key

equ: Id_paper = Papers.Id_paper

access key

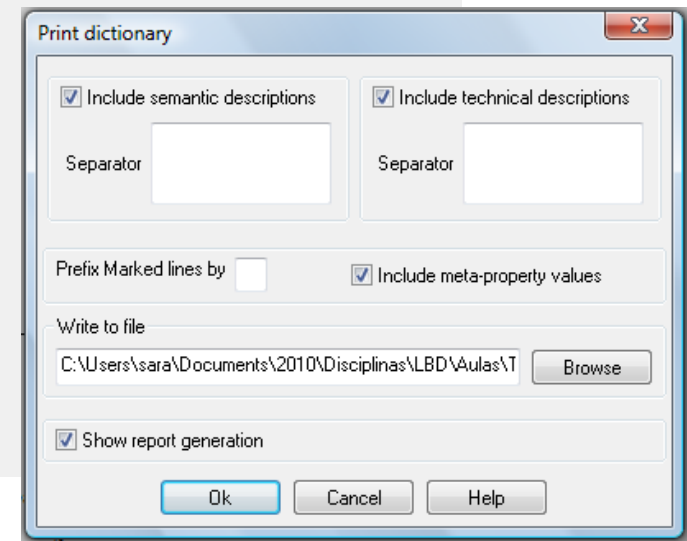
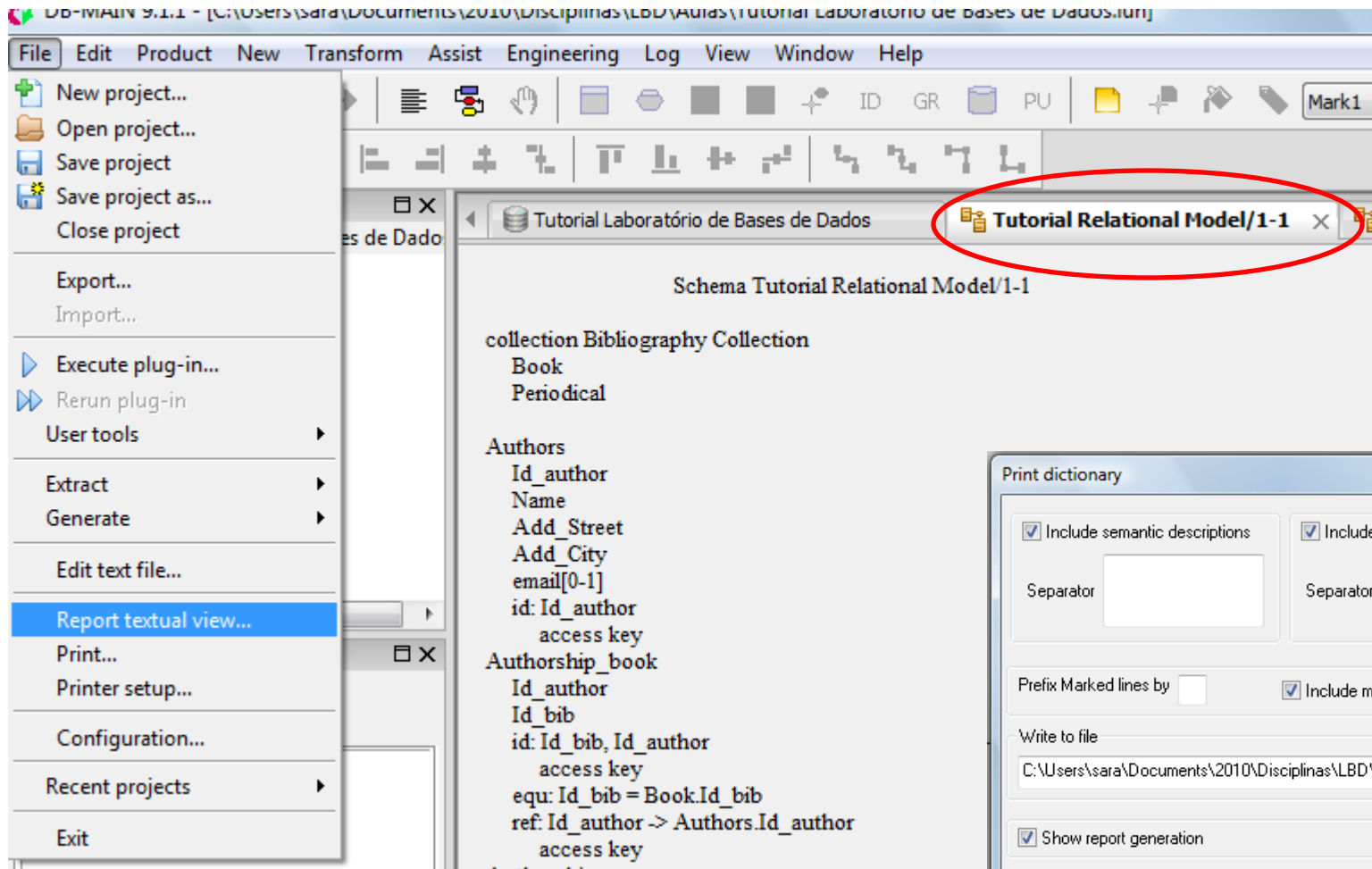
ref: Id_author -> Authors.Id_author

Bibliography_Item

Id_bib

→ Index!

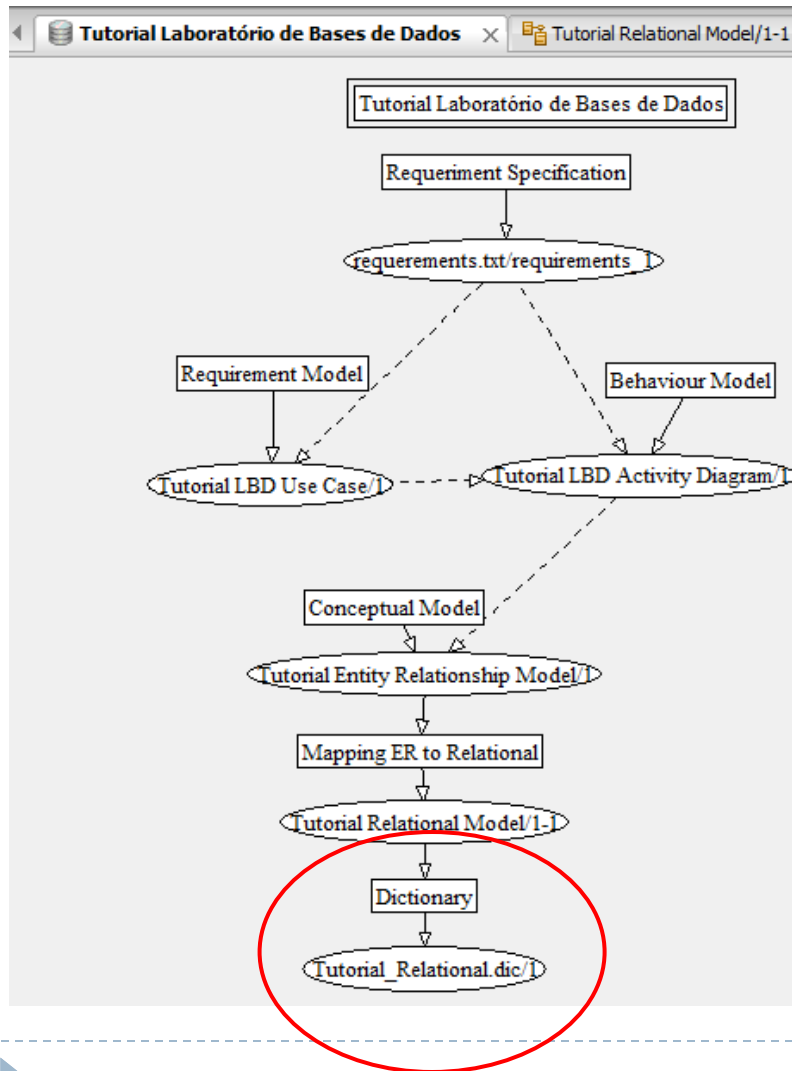
Data Dictionary



Only in the text view mode.



Data Dictionary



```

1
2
3      Dictionary report
4
5      Project          Tutorial Laboratório de Bases de Dados
6
7
8
9
10             Schema Tutorial Relational Model/1-1
11
12 collection Bibliography Collection
13   Book
14     MappingOID: 525
15   Periodical
16     MappingOID: 523
17
18 Authors
19   MappingOID: 558
20   Id_author
21     MappingOID: 560
22   Name
23     MappingOID: 562
24   Add_Street
25     MappingOID: 574
26   Add_City
27     MappingOID: 578
28   email[0-1]
29     MappingOID: 592
30   id: Id author

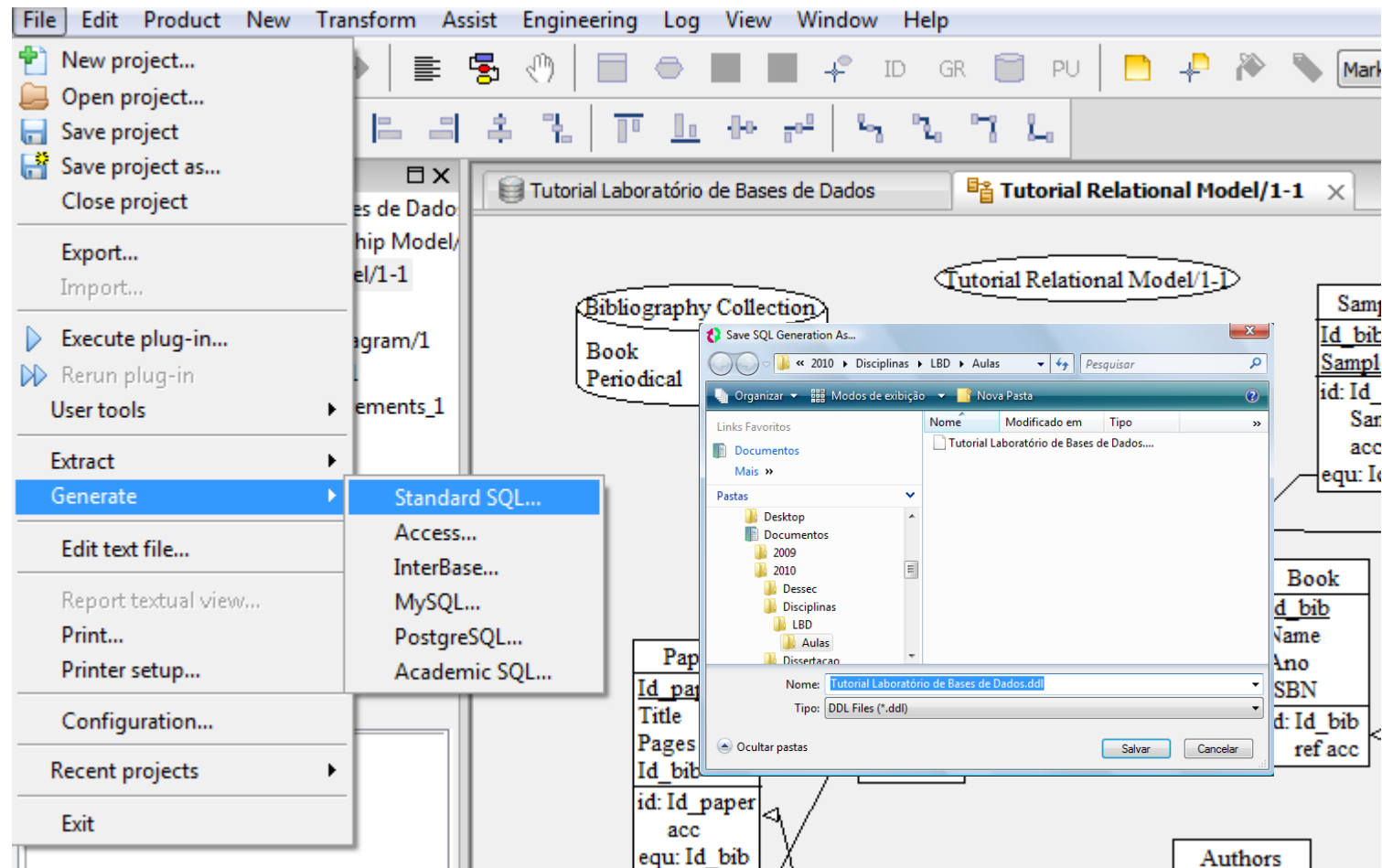
```

```

168      Supertype: Bibliography Item
169      MappingOID: 1093
170 Samples
171   MappingOID: 1143
172   Id_bib
173     MappingOID: 531
174 Samples
175   MappingOID: 600
176   id: Id_bib, Samples
177     MappingOID: 1150
178     access key
179     MappingOID: 1150
180   equ: Id_bib = Bibliography_Item.Id_bib
181     MappingOID: 1145
182
183

```

Generating the SQL Script



Generating the SQL Script

