# Designing data-intensive applications with WebML

(based on slides by

Marco Brambilla
marco.brambilla@polimi.it)

http://www.slideshare.net/mbrambil

## Outline

- Motivation
- WebML models and concepts
- WebML development process

## The WebML models

- WebML: a conceptual language for high-level design of data-intensive web sites
- Graphical notation to express all constructs of WebML
- Even if the designer can make use only of graphical notation, the WebML model is defined by an XML DTD; therefore, each project is stored as an XML file.



## Model-driven development

- Data-intensive Web site design today is founded on methodologies stolen from different sectors (DB, software eng., …)
- Lack of model-driven support of data-intensive Web Sites
  - Navigation equal to database structure
  - Lot of hand-written code
- Big efforts are requested even for prototyping

# Motivation

Maintenace and evolution of data-intensive sites is going to experience a substantial increase in complexity and cost.

→ **3 factors:**

- application will need to serve content across a wide variety of devices with different capabilities, ranging from PC to PDAs, to cellular phones, to the digital television and videotext
- competition among providers will more and more require that content and services be tailored to the needs of individuals, as it si already customary in such portal sites as myYahoo, myCDNow, etc
- lack of IT personnel is foreseen as a major bottleneck of the IT industry in the short term.

# Site complexity

- Increased complexity of modern Web sites impacts on cost of:
  - site development
  - site evolution

## shortage of IT & art personnel

- Need to rise the level of abstraction
- Available skills can address analysis and high-level design instead of huge time-wasting manual coding of ASP pages
- IT technicians can focus on optimization and performance analysis
- Art people can focus on “creation”



## A scrupolous Modeling approach

- Can reduce development efforts (cost and time)
- Allows a more structured development process
- Produces more usable and coherent final results
- Design models are self-documenting and always up-to-date projects

Immediate prototyping can be achieved

## Requirements for Web modeling

- Expressivity
  - Real-life cases should be expressible
  - Frequently used design patterns should be captured
- Ease of use
  - Intuitive notation
  - Clear semantics
  - Consistency checks
- Implementability
  - Natural and efficient mapping to physical data structures
  - Flexible code generation from behavioral specifications

9

## WebML purpose

- WebML aims at providing a structured approach to the design of Data-intensive Web sites
- A set of integrated Models should help designers in high-quality Web sites production
- All the facets of Web design should be addressed
- Use of old or uncoherent methodologies becomes deprecated

10

# Target of WebML

- Target: data intensive Web sites
  - large amount of data
  - interfaces directed to general public
    - exploratory
    - browsing-oriented
    - personalized (1 to 1)
  - volatile content, structure, navigation, presentation
- WebML is not the right approach for:
  - Small Web sites (Homepages, …)
  - Static Web sites



# WebML models and concepts

## The WebML models

- WebML: a conceptual language for high-level design of data-intensive web sites
- Graphical notation to express all constructs of WebML
- Even if the designer can make use only of graphical notation, the WebML model is defined by an XML DTD; therefore, each project is stored as an XML file.

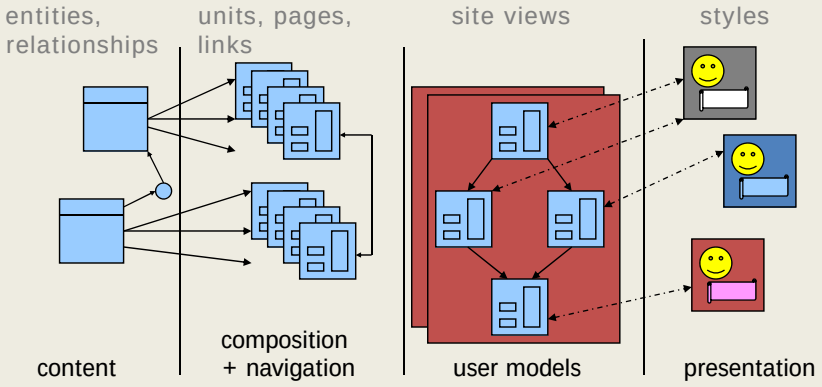


## The WebML models

- Content model: data organization
- (Content) Derivation model: redundant data definition
- Composition model: definition of site pages as set of subpages and elementary publishing units
- Navigation model: definition of links between pages and between units
- Presentation model: positioning of the units in the page and definition of graphical appearance

# Preview of WebML concepts

- Site = Content + Composition + Navigation + Presentation



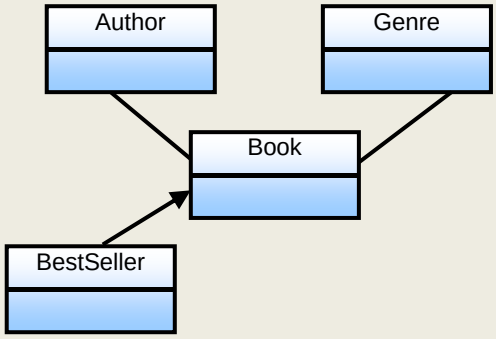



# Structure Model (1)

- The objects published in the site and their relationships:
  - Entity: an object type in the application domain
  - Attribute: scalar property of an entity
  - Relationship: A connection between entities
  - IS-A hierarchy: classification and grouping
- Compatible with Entity-Relationship and UML class diagrams

## Structure Model (2)

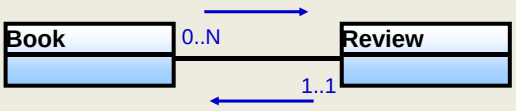
- Simplified Entity-Relationship model
  - Binary relationships between entities
  - IS-A hierarchies
  - Simple typed attributes in entities
  - Derivation model can be applied for redundant data






## Relationship cardinality

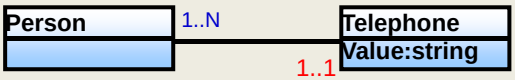
- Relationship Book_Review
  - book2review minCard: 0 maxCard: N
  - review2book minCard: 1 maxCard: 1




- It reads: a book may have zero or more reviews, a review deals with one and only one book

# Multi-valued attribute

- A multi-valued attribute is an attribute for which an object may have a set of values (e.g., a person may have a set of telephone numbers)
- Multi-valued attributes are represented by means of an entity plus a relationship
- Example: a person has a set of telephone numbers

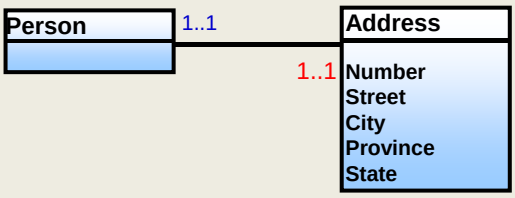



NOTE: it often happens that such an entity is a "weak" entity, it cannot exist independently of the associated "strong" object

19

# Structured attributes

- A structured attribute is an attribute with an internal structure (e.g., an address may consist of several fields, like civic number, street, city, province, state, etc..)
- Structured attributes are represented by means of an entity plus a relationship
- Example: a person has an address consisting of number, street, city, province, state

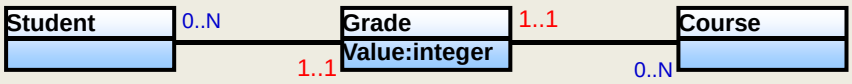



NOTE: often happens that such an entity is a "weak" entity, it cannot exist independently of the associated "strong" object

20

# Relationship with attributes

- An attribute of a relationship is a description of a property that refers to a pair (more generallly to sets) of objects (e.g., the grade taken by a student in a course)
- A relationship with attributes is represented by an entity and two (more generally, several) relationships

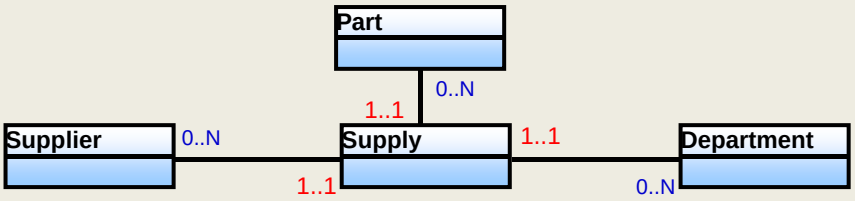



NOTE: the "grade" entity is a "weak" entity, it cannot exist independently of the associated objects



# N-ary Relationship

- An N-ary relationship is a relationship involving more than two entities (N>2)
- An N-ary relationship is represented by an entity plus N binary relationships
- Example: a supplier supplies a part to a department



TECHNIQUES

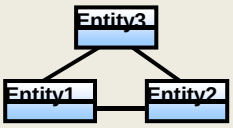
## Structure patterns

- Structure pattern: typical structure design solution, found very frequently in real Web sites
- Patterns can be combined and used together to achieve complete information access

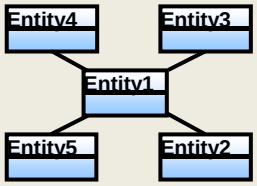


## Structure Patterns

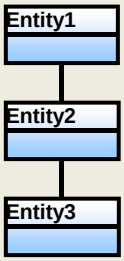
- **Network skeleton:**



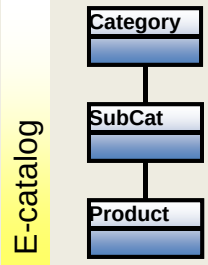

- **Star skeleton:**




- **Hierarchy skeleton:**



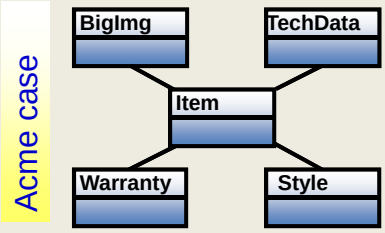
# 1. Hierarchy Pattern




- A categorization hierarchy
- User can browse the hierarchy from top to bottom (and back)
- Often used in e-commerce applications
- BEWARE: it is NOT a generalization (ISA) hierarchy !!!

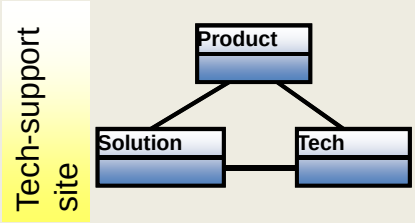


# 2. Star Pattern




- A main object can be recognized
- A set of informative elements are related to the object
- Users can jump from main object to information pages and back

# 3. Network Pattern

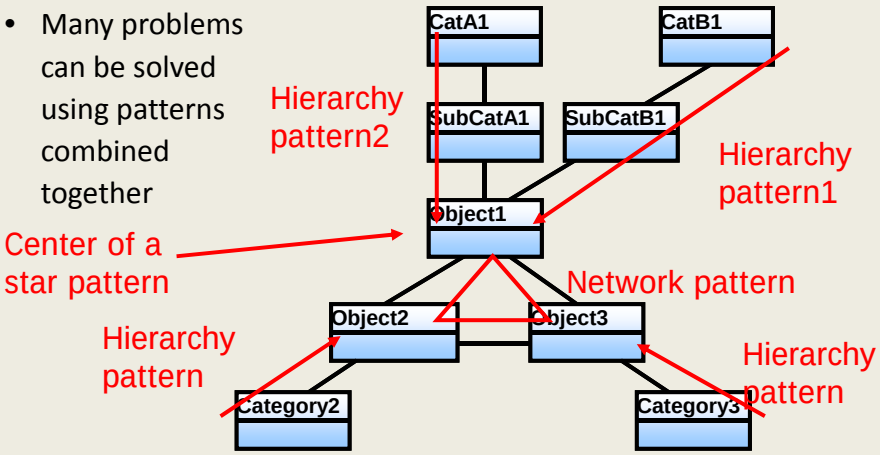



- A set of elements to be published
- All at the same level of importance
- Every object has a detailed description
- Users can browse from an object to the related ones



# Pattern combination

- Real web sites are much more complex
- Many problems can be solved using patterns combined together



## Derivation Objectives

- Derivation is a modelling phase that makes it possible:
  - To avoid redundancy
  - To augment the content of an entity by adding attributes, either imported or computed from related objects.
  - To define the population of entities or of relationships, based on some property of the involved objects.

29

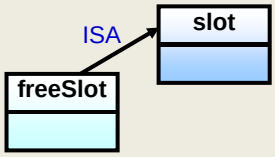
## WebML OCL

- Derivation consists in writing expressions called "*derivation queries*" (→ views!)
- Queries can be expressed using the WebML OCL (Object Constraint Language)
- Derivable concepts:
  - Entities, relationships, attributes
- Derivation queries can be automatically transformed into SQL views installed in the database

30

# Derived entities

- Sub-Entity population in ISA hierarchies can be specified by means of OQL or OCL queries
- "A free slot is a slot having 0 reservations"
- WebML OQL: "SuperEntity where count(reservation)=0"



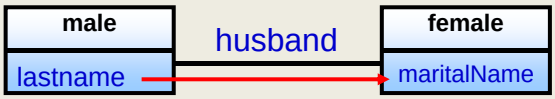



# Derived attributes

- Attributes of an entity can be derived by associating a query to them .
- Three types of derived attributes:
  - Imported attributes: maritalName: Self.husband.lastname
  - Aggregate attributes: reservation#: count(Self.reservation)
  - Computed attributes: lastPrice: Self.price*discount
- The ***Self*** keyword identifies the current entity in which the attribute is defined

## Imported attribute

- Import a piece of external information into an entity






## Computed and aggregate attributes

- Total price of order as sum of price of order lines


- Derivation rules:
  OL.TotalPrice = Self.Price * Self.Quantity
  Order.TotalPrice = Sum (Self.Order2OL.Price)
  Order.NumLines = Count (Self.Order2OL)

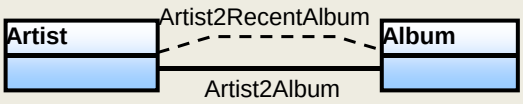
# Derived relationships

- Relationships can be derived by means of a WebOQL query in two ways:
  - by subsetting and/or concatenating existing relationships
  - by matching pairs of objects by means of a condition
- Variable Self denotes the instance of the source entity of the relationship



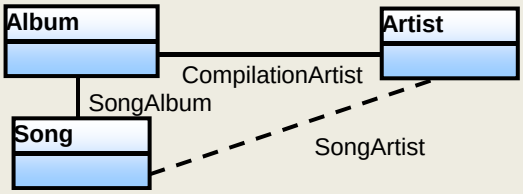
# Derived relationships: subsetting

- Subsetting existing relationships:



## Derived relationships: concatenation

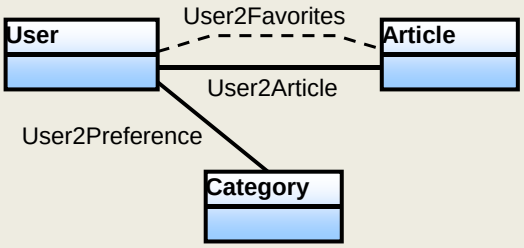
- Concatenating existing relationship:




37

## Derived relationships: matching pairs

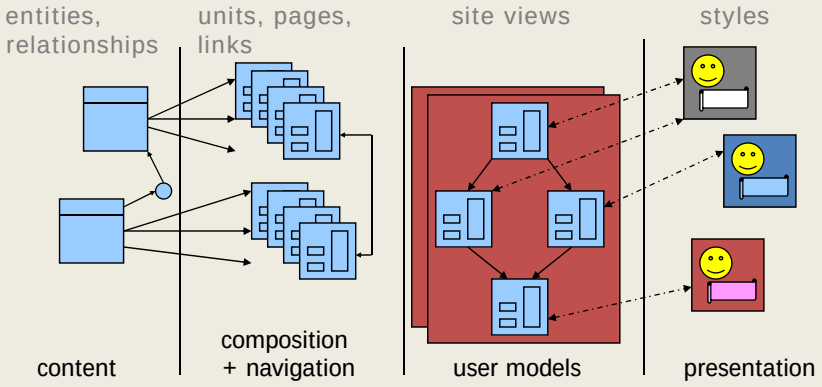
- User2Favorites relates User to all the Articles whose category appears in the set of the preferred categories in the user profile.




38

# Preview of WebML concepts

- Site = Content + Composition + Navigation + Presentation



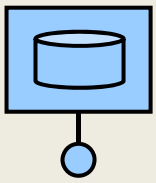



# Hypertext Model

- The *content units* are the actual information published in the hypertext nodes
- The *links* connect the hypertext nodes, for *navigation* purposes
- The experience the user will achieve is obtained by means of *siteviews*
- The hypertext is divided into *pages* served to users

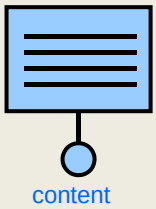
## **Composition:** examples of Content Units

DATAUNIT




To publish information about
A SINGLE object
(e.g. AuthorDetail)

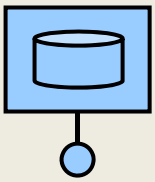
INDEXUNIT

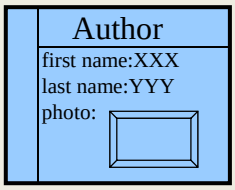



To publish a list of objects
(e.g. IndexOfAuthors)

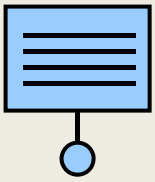
## Composition: examples of Unit rendering

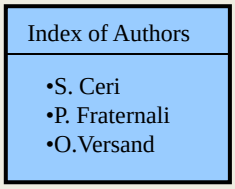
DATAUNIT




INDEXUNIT



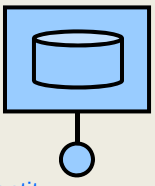



# Basic Content Units

Content:
- instances of an entity

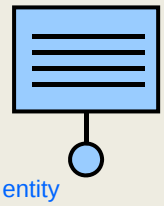
Selector:
- set of conditions

DATAUNIT



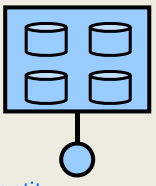
entity
[Selector]

INDEXUNIT



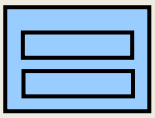
entity
[Selector]

MULTIDATAUNIT

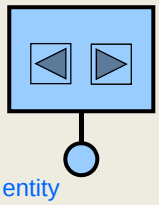


entity
[Selector]

ENTRYUNIT

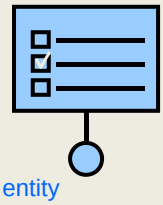


SCROLLERUNIT



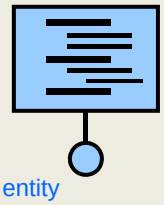
entity
[Selector]

MULTICHOICE



entity
[Selector]

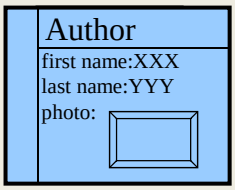
HIERARCHICAL



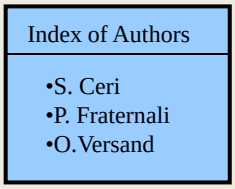
entity
[Selector]

# Meaning of Content Units

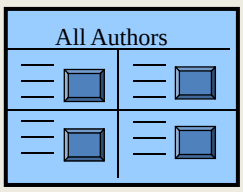
DATAUNIT



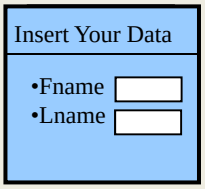

INDEXUNIT



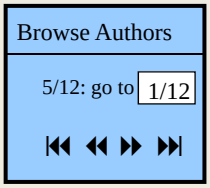

MULTIDATAUNIT




ENTRYUNIT

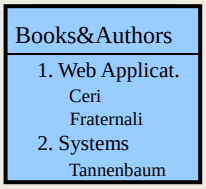



SCROLLERUNIT



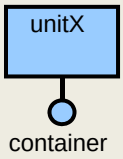

MULTICHOICE


HIERARCHICAL



## Content Units

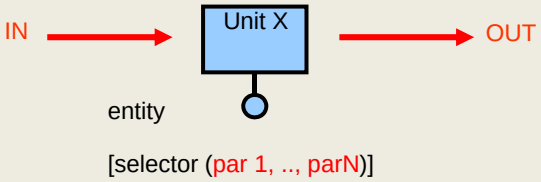
- A WebML unit is the atomic *information publishing element*




- It is a "view" defined upon a container of objects:
  - All the instances of an entity
  - Instances of an Entity that meet a selection condition called selector

45

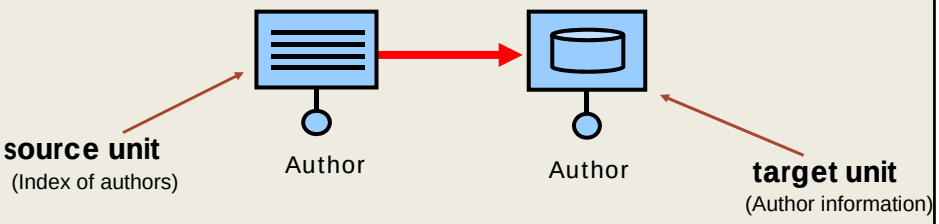
## Unit input and output




- A unit may need some "context" to be computed
- Each unit exposes input and output parameters
- Input is required to compute the unit itself:
  - Parameters pre-defined for the unit
  - Other parameters required by the selector of the unit
- Output can be used to compute other unit(s) depending on the current unit

46

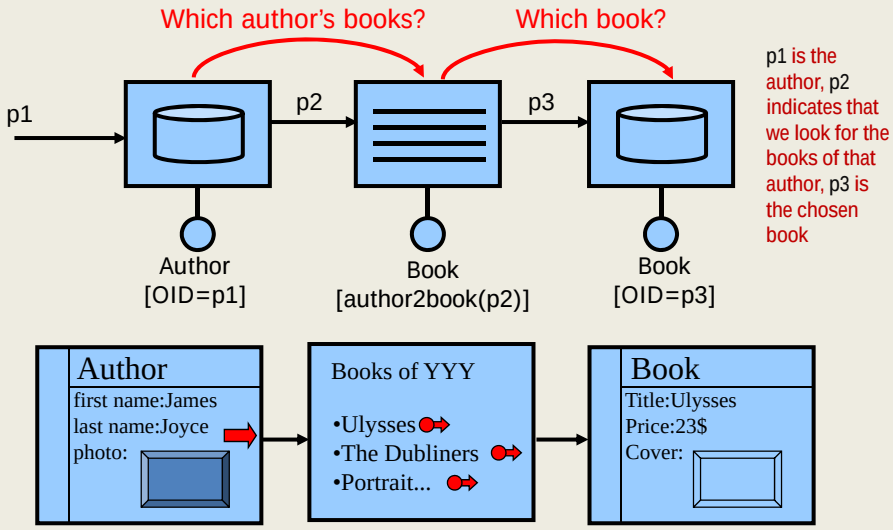
# Navigation: contextual links




- A contextual link is an oriented connection between two units (source unit and target unit), normally rendered by means of anchors or submit buttons
- Purpose of a contextual link:
  - Allowing the user to move from one place to another
  - Transporting information from one place to another
  - Activating a computation (side effect)

47

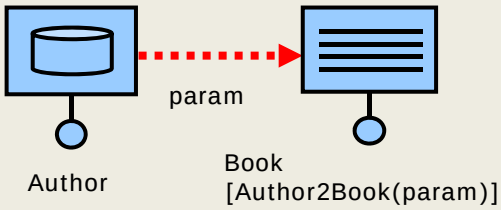
# Example of links




p1 is the author, p2 indicates that we look for the books of that author, p3 is the chosen book

48

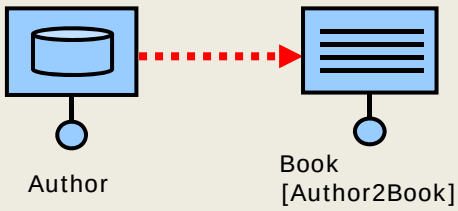
# Link Parameters




- Context is transported through links using link parameters
- A link parameter is defined by:
  - Name
  - Source: one of the output parameters of the source unit
  - Target: one of the input parameters of the target unit

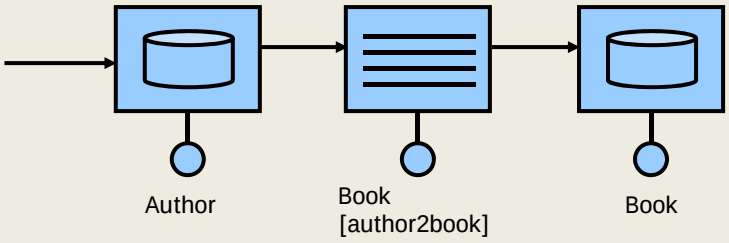
# Default Link Parameters

- Whenever possible, link parameters are inferred from the diagram and need not be explicitly specified
- Diagrams become simpler and more readable
  - Example:

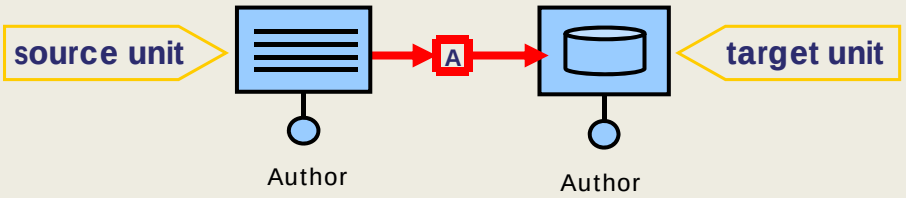


# Defaults for selectors

- Whenever possible, selectors and their parameters are inferred from the diagram and need not be explicitly specified
- Example:

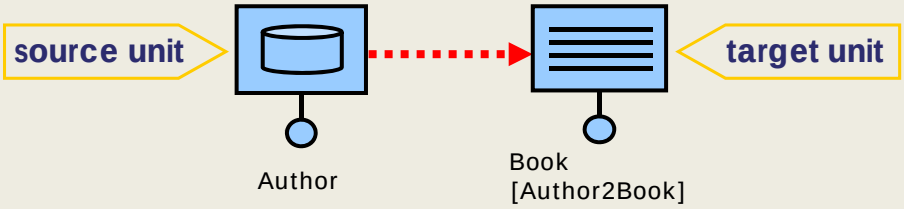



# Automatic Links



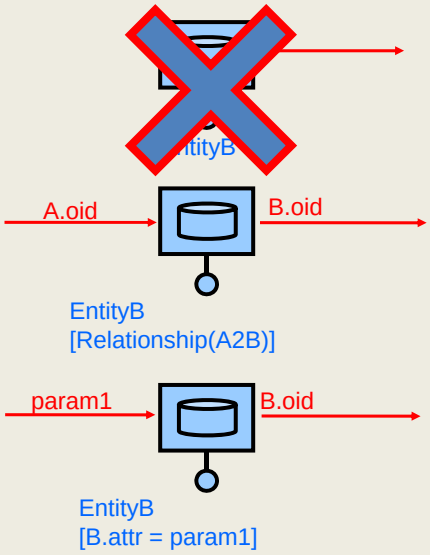

- An automatic link passes some **default context** to the destination unit immediately after the display of the source unit, **without the user intervention**
- Subsequently, the user can change the passed context by choosing a different object, using the anchor(s) representing the link

# Transport Links



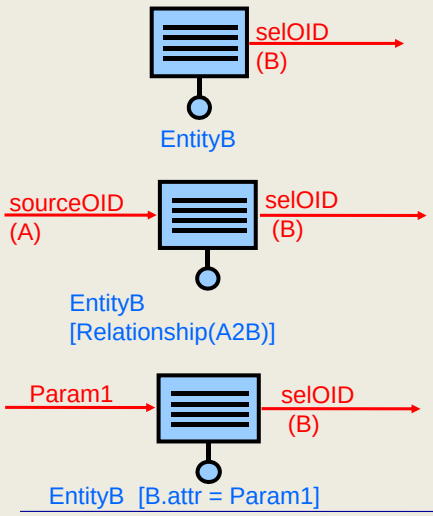

- A transport link has a default context that is passed to the target unit immediately after the display of the source unit, without the user intervention
- The user cannot change the default context and therefore the link is not rendered with an anchor

# Examples: DataUnit




- No input links and no selector: wrong unit!
  - Input parameter: OID of a connected object: the related object is shown
  - Input parameter: value to be used in the selector: the matching object is shown

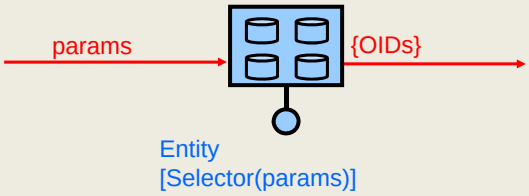
# Examples: IndexUnit




- No input links: all objects shown
  - Input parameter: OID of the source object of the relationship role, all related object are shown (e.g. books written by J. Joyce)
  - Input parameter: value to be compared with the attribute, matching objects shown (e.g. books that cost less than 15 euros)



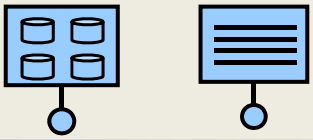
# Examples: MultiDataUnit




- Multidata units
  - Present multiple instances of an entity (SET OF OBJECTS)
- The container is an entity with (optional) selector
  - Input parameters requested for the computation of the selector:
    - Values for attribute comparisons
    - OIDs of participants to relationships
  - Output parameter:
    - The set of OIDs of the published objects

# MultiData vs. Index

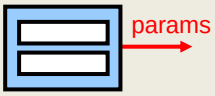
- MultiData or Indexes?
  - Indexes can be used as access mechanisms to show detailed information about one object
  - Multidata Units publish information about many objects at a time
- The difference is in the output parameter:
  - Indexes output the OID of one selected object (e.g. choose one author of the list)
  - Multidata units output the OIDs of all the displayed objects (e.g. choose the authors born after 1900)



57

# Examples: Data Entry Unit

- Unit for describing input forms that allow information submission by the user
- Content is shipped to other units via outgoing links (normally one), using link parameters
- Typically translated into HTML using the <form> tag and the associated submit button




- The data entry unit permits the designer to specify a device for collecting user input

58

## Entry Fields and Selection Fields

- Entry units contains two types of widgets for data entry:
  - **Fields** to insert new values
  - **Selection fields** to select a value from a list
- Fields have properties:
  - Preloading (of an existing value in the field)
  - Modifiability (Y/N)
  - Visibility (hidden/shown)
  - Data type (string, integer, text, etc..)
  - Input validation rules

59

## Entry Fields and Selection Fields

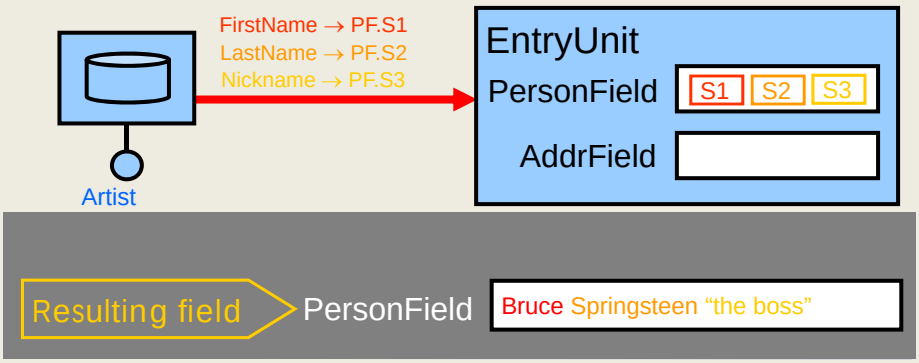
To insert data we define three field types, which can be included into entry units:

- The “in” field type is used when the user inserts data in a blank form field
- The “out” field is used when data is preloaded in the form fields and the user cannot modify it
- The inout field is used when the form is preloaded but the user can modify its content.

60

# Preloaded Fields

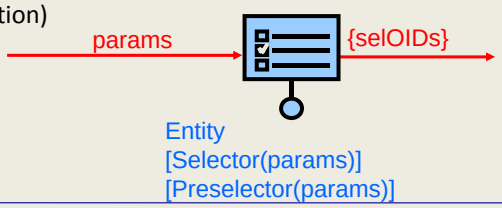
- A field can be preloaded with value(s)
  - Field slots allow the concatenation of multiple values in the same field
  - Slots can contain dynamic values taken from the database or static values defined in the model
  - In this example we have 3 fields of type "in". When the user presses the ok button the input data are sent to the unit following the outgoing link (typically an operation unit).






# Examples: Multichoice Unit

- Multichoice units publish indexes of elements (SET OF OBJECTS) among which the user can select one or more elements (using checkboxes)
- The container is an Entity (with optional selector & pre-selector)
  - Input parameters:
    - Those requested for the computation of the selectors (values for attribute comparisons and OIDs of participants to relationships)
  - Output parameters:
    - OIDs of the objects checked by the user
  - Preselector: allows one to define a sub-set of elements as pre-checked (before any user interaction)



# Examples

- No input links: all objects shown, no pre-checked elements



[PRE:Relationship(A2B)]

- Pre-selector: all objects shown, pre-checked elements related to A

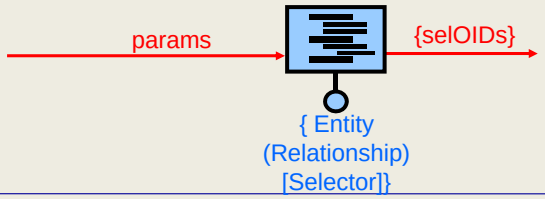

[Relationship(A2B)]

- Selector: only elements related to A shown, no pre-checked elements

63

# Examples: Hierarchical Unit

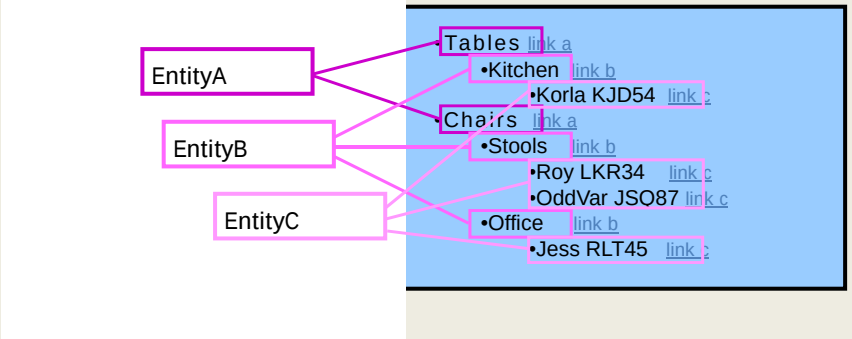
- Hierarchical units:
  - Publish an index of elements, with entries organized hierarchically using entities connected by relationships
  - Allow the user to select one element from any level of the hierarchy
- The container is a set of entities and connecting relationships (with optional selectors at all levels)
  - Input param: requested for the computation of the selectors :
    - Values for attribute comparisons
    - OIDs of partecipants to relationships
  - Output parameter: OID of the object selected by the user




64

# Hierarchical Unit outgoing links

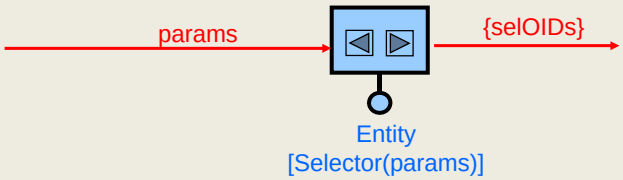
- Each link is rendered as an anchor at the proper level of the hierarchy
- The level where the link is placed depends on the type of the link parameter(s)





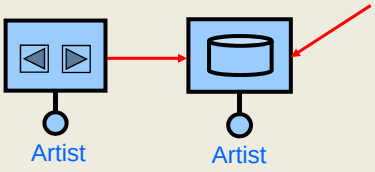

# ScrollerUnit

- Adds browsing capabilities to a set of objects
  - Publish links to the first, previous, next, last object of the set
  - Used in conjunction with data, index and multidata units
  - Number of scrolled objects = block factor
  - Input: values for attribute comparisons, OID of participants to a relationship
  - Output: the set of OIDs (possibly 1) of the current block of objects
- Always placed in the same page as another content unit that publish the current (block of) object(s)



# Scroller + data unit

- The entity is the same for the scroller and the data unit
- It is possible to access the data unit also from another page and the scroller is automatically syncronized

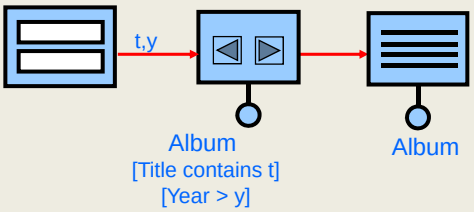




67

# Scroller + index

- Paging the result of a search

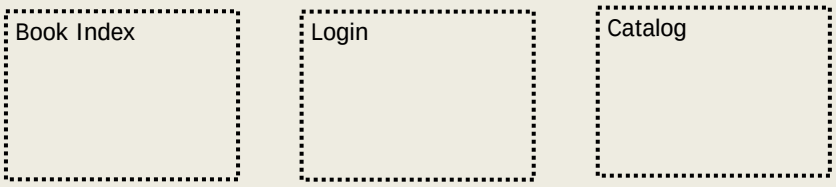





68

## Pages

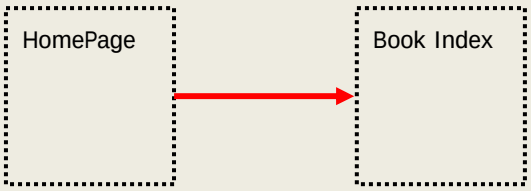
- A page is a container of one or more pieces of information shown to the user at the same time
- Nesting of pages is allowed: a page can have sub-pages
- The user navigates a site made of pages






## Non-contextual links

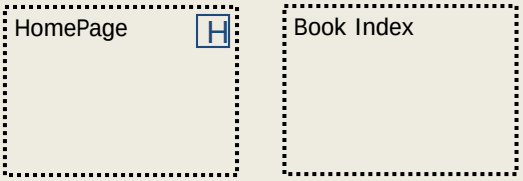
- A non-contextual link is a link between pages
- No context (information) is transported




- The user can browse from a page to another one via an anchor (e.g., *>>Books*)

# Home Page

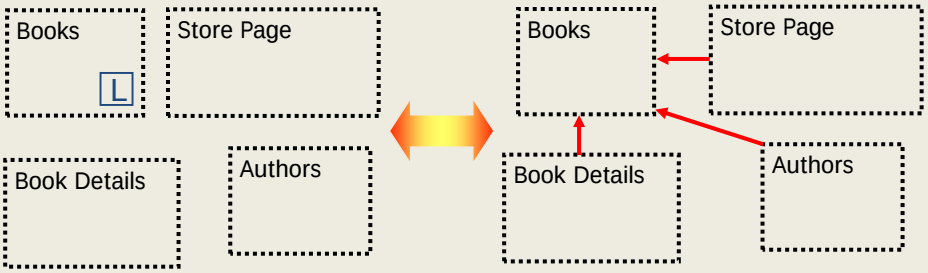
- The Home Page is the main page of a site
- It is the first page of the site that the user should see
- Each siteview must contain a page marked as "Home"





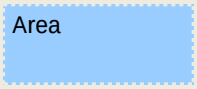

# Landmark pages

- *Landmark pages:* globally visible pages. The user can jump to them from everywhere in the site view
- It is equivalent to a non contextual link implicitly defined from every other page in the site view to the *landmark page*



# Areas

- An area is a set of logically homogeneous pages
- Examples of areas:
  - Sections of a portal: Sport, Music, Technology, …
  - Elements of a data-management system: Products management, News management, …
- Areas can be nested, so that sub-areas can be defined inside areas
- Each area should have a DEFAULT PAGE or a DEFAULT SUB-AREA, to give a meaning to landmark areas and non-contextual links pointing to areas






# Site Views

- A siteview is a set of pages and/or areas forming a coherent view of the site
- Multiple site views can be defined on the same data model
- Different site views can be published for different types of users and for different types of output devices
- Site views can be
  - Public: everyone can enter
  - Private: access control with password protection is enforced

marco.brambilla@polimi.it
http://home.dei.polimi.it/mbrambil