Business Process Modeling
BP Aspects

- Process flow
  - Process modeling
    - UML Activity Diagrams
    - BPMN

- Information
  - Conceptual modeling
    - UML Class diagrams
    - (Entity–Relationships)

- Interaction
  - Interaction modeling
    - Use cases
UML

- Unified Modeling Language
- Standardized by OMG
- Several diagrams
  - Class diagrams
  - Activity diagrams
  - Use Case diagrams
  - (Sequence diagrams)
  - (Statecharts)

Conceptual modeling
Process modeling
Functional modeling
Conceptual Modeling

CLASS DIAGRAM
Conceptual Modeling

- Construction of model,
  - Providing an optimal description
  - From the stakeholder perspective

- Is the formalization phase after
  - Requirements elicitation and collection
  - Requirements analysis
Goal

- Capture
  - Main (abstract) concepts
  - Characteristics of the concepts
    - Data associated to the concepts
  - Relationships between concepts
## Abstrcction levels

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Concept</th>
<th>Entity</th>
<th>Class</th>
<th>Category</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concrete</td>
<td>Instance</td>
<td>Item</td>
<td>Object</td>
<td>Example</td>
<td>Occurrence</td>
</tr>
</tbody>
</table>
Class

- They represent set of objects
  - Common properties
  - Autonomous existence.
  - E.g. facts, things, people

- An instance of a class is an object of the type that the class represents.
  - In an application for a commercial organization CITY, DEPARTMENT, EMPLOYEE, PURCHASE and SALE are typical classes.
Class – Examples

Employee

City

Sale

Department
Object

- Model of item (physical or within the software system)
  - ex.: a student, an exam, a window

- Characterized by
  - identity
  - attributes (or data or properties)
  - operations it can perform (behavior)
  - messages it can receive
Association

- Represent logical links between two classes.
- An occurrence of an association is an couple made up of occurrences of entities, one for each involved class
  - Residence can be an association between the classes City and Employee;
  - Exam can be an association between the classes Student and Course.
Associations

Class
Student

Association between classes

Class
Course

Link between objects
Association – Examples

- Student attends Course
- Employee works in City
- Residence
Recursive association–Samples

- Student
- Employee
- Friend
- manager
- employee

Supervise
Link

- Model of association between objects
Attribute

- Elementary property of classes
  - Name
  - Type

- An attribute associates to each object (occurrence of a class) a value of the corresponding type
  - Name: String
  - ID: Numeric
  - Salary: Currency
Attribute – Example

- **Student**
  - ID : int
  - Name : String

- **Course**
  - Code : String
  - Year : int

- **Employee**
  - Name : String
  - Age : int
  - Salary : Currency

- **City**
  - Name : String
  - Inhabitants : int

Relationships:
- Exam between Student and Course
- Works_in between Employee and City
- Residence between City and Employee
Attribute – Example

Is everything ok?
Multiplicity

- Describe the maximum and minimum number of links in which a class occurrence can participate
  - Undefined maximum expressed as *
- Should be specified for each class participating in an association
A car can mount none, up to four wheels
A wheel can be mounted on none or at most one car.
Multiplicity

- Typically, only three values are used: 0, 1 and the symbol * (many)

- Minimum: 0 or 1
  - 0 means the participation is optional,
  - 1 means the participation is mandatory;

- Maximum: 1 or *
  - 1: each object is involved in at most one link
  - *: each object is involved in many links
**Multiplicity**

- $n$: Exactly $n$
- $*$: Zero or more
- $m..n$: Between $m$ and $n$ ($m,n$ included)
- $m..*$: From $m$ up
- $0..1$: Zero or one (optional)
Multiplicity

Order

Invoice

1

Sale

0..1

Person

City

0..*

Residence

1

Tourist

Trip

1..*

Reservation

0..*
Aggregation

- B *is-part-of* A means that objects described by class B can be attributes of objects described by A
Example

Car

CD player

Engine
  power

Tyre

1

4

1
Association Class

- The association class define the attributes related to the association
- A link between two object includes
  - The two linked objects
  - The attributes defined by the association class
Association Class Limitations

- Association class
  - Fee is a function of consultant and company
    - fee (Consultant, Company)

- Intermediate class
  - Fee is a function of the contract
    - fee (Contract)
Association class limitation

- Case
  - Consultant working several times for the same Company
- Cannot be represented by association class
- Only representable through intermediate class
Specialization / Generalization

- B *specializes* A means that objects described by B have the same properties of objects described by A.
- Objects described by A may have additional properties.
- B is a special case of A.
- A is a generalization of B (and possible other classes).
Generalization

Person
- First : String
- Last : String
- SSN : String

Employee
- Salary : Currency

Student
- ID : int
Inheritance terminology

- Class one above
  - Parent class
- Class one below
  - Child class
- Class one or more above
  - Superclass, Ancestor class, Base class
- Class one or more below
  - Subclass, Descendent class, Derived class
Example of inheritance tree

- **Living species**
  - **Animal**
    - **Human being**
      - **salesman**
    - **Customer**
  - **vegetal**
    - **Flower**
  - **Flower seller**
NL Requirements Specification

- Requirements specifications are often written in natural language (NL)
  - At least in the first draft.
- NL is, by nature, subject to ambiguity and misinterpretation.
- Inaccuracies and ambiguous terms must be removed
  - Necessary an in-depth analysis of the specification document
We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

For each course participant (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, employer’s name, address and telephone number, previous employers (and period employed), the courses attended (there are about 200 courses) and the final assessment of each course.

We need also to represent the seminars that each participant is attending at present and, for each day, the places and times the classes are held. Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.

If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Requirement analysis

- Choose the appropriate level of abstraction
  - Identify the main concepts
- Construct a glossary of terms
- Identify synonyms and homonyms, and standardize terms
- Make cross-references explicit
- Standardize sentence structure
- Avoid complex phrases
We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

For each course participant (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, employer’s name, address and telephone number, previous employers (and period employed), the courses attended (there are about 200 courses) and the final assessment of each course.

We need also to represent the seminars that each participant is attending at present and, for each day, the places and times the classes are held. Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.

If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
<th>Synonym</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course</td>
<td>Course offered. Can have various editions.</td>
<td>Seminar</td>
<td>Instructor, Trainee</td>
</tr>
<tr>
<td>Trainee</td>
<td>Participant in a course. Can be an employee or self-employed.</td>
<td>Participant</td>
<td>Course, Employer</td>
</tr>
<tr>
<td>Instructor</td>
<td>Course tutor. Can be freelance.</td>
<td>Tutor</td>
<td>Course</td>
</tr>
<tr>
<td>Employer</td>
<td>Company by which a trainee is employed or has been employed.</td>
<td></td>
<td>Trainee</td>
</tr>
</tbody>
</table>
We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

For each course participant (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, employer’s name, address, and telephone number, previous employers (and period employed), the courses attended (there are about 200 courses) and the final assessment of each course.

We need also to represent the seminars that each participant is attending at present and, for each day, the places and times the classes are held. Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.

If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Example

- We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.
- For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer’s, previous employers (and start date and end date of the period employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.
- For each employer we store the name, address, and phone number
- Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. We need also to record for each day, the places and times the classes are held. For each edition, we represent the start date, the end date, and the number of participants.
- If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.
- For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
We wish to create a IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

- For each employer we store the name, address, and phone number.
- We need also to represent course editions and, for each day, the places and times the classes are held. Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.
- If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.
- For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Example

Statements about Trainees

For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer’s, previous employers (and start date and end date of the period employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.

For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Example

- We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

- For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer’s, previous employers (and start date and end date of employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.

- For each employer we store the name, address, and phone number.

- If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

- For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Example

- We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

  - For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer's, previous employers (and start date and end date of the period employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.

  - For each employer we store the name, address, and phone number.

  - Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. We need also to record for each day, the places and times the classes are held. For each edition, we represent the start date, the end date, and the number of participants.

  - If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

  - For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Example

- We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.
- For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer’s, previous employers (and start date and end date of the period employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.
- For each employer, we store the name, address, and phone number.
- We need also to represent course editions and, for each day, the places and times the classes are held. Each course has a code and a title and can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.
- If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.
- For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors' telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
We wish to create an IS for a company that runs training courses. For this, we must store data about the trainees and the instructors.

For each trainee (about 5000), identified by a code, we want to store the social security number, surname, age, gender, place of birth, current employer’s, previous employers (and start date and end date of the period employed), the courses editions attended (there are about 200 courses) and the final assessment of each course edition.

For each employer we store the name, address, and phone number.

We need also to represent course editions and, for each day, the places and times the classes are held. Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an ‘edition’ of the course. For each edition, we represent the start date, the end date, and the number of participants.

If a trainee is a self-employed professional, we need to know his or her area of expertise, and, if appropriate, his or her title. For somebody who works for a company, we store the level and position held.

For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors’ telephone numbers are also stored. An instructor can be permanently employed by the training company or can be freelance.
Essential guidelines

- If a concept has significant properties and/or describes types of objects with an autonomous existence, can be represented it by a class.
- If a concept has a simple structure, and has no relevant properties associated with it, it is likely an attribute of a class.
- If the requirements contain a concept that provides a logical link between two (or more) entities, it is convenient to represent this concept by an association.
- If one or more concepts are particular cases of another concept, it is convenient to represent them by means of a generalization.
Modeling strategies

- **Top–down**
  - Start with abstract concepts and perform successive refinements

- **Bottom–up**
  - Start with detailed concepts and proceed with integrating different pieces together

- **Inside–out**
  - Like bottom–up but beginning with most important concepts first

- **Hybrid**
Conceptual model quality

- Correctness
  - No requirement is misrepresented
- Completeness
  - All requirements are represented
- Readability
  - Is easy to read and understand
- Minimality
  - There are no avoidable elements
References