UML for Embedded Systems

II. Design

Ludovic Apvrille
ludovic.apvrille@telecom-paristech.fr
Eurecom, Office 470

Memo on Methodology

I. Analysis
Use case
First class diagram
Relevant scenarios

II. Design
Classes of the system
Architecture of the system

III. Detailed design
Behavior of the system

IV. Validation of the system
Simulation
Code generation
Design stage

- **Purpose**
  - Structure the system under the form of classes and communication among those classes

- **Steps**
  - Identification of secondary classes
    - Class diagram
  - Identification of relations between classes
    - Association, aggregation, specialization, etc.
    - Class diagram
  - Modeling of class hierarchy
    - Composite structure diagram
  - Modeling message exchange between classes
    - Composite structure diagram

Design

- **UML class diagrams**
- **UML composite structure diagrams**
- **Exercises**
UML Class diagrams

- Set of classes
- Definition of classes
  - Identifier
  - Stereotype (abstract, interfaces, etc.)
  - Attributes
  - Operations
- Relations among classes
  - Association
  - Aggregation
  - Multiplicity

Definition of a Class

BankAccount

- value : Integer
- owner
- amount
- static nextAvailableAccountNb

+ credit(Real)
+ debit(Real)
+ static getNextAvailableAccountNb () : Integer
Association

- Association name
  Client ----> purchase ----> Product

- Association with navigability
  Student ----> studyIn ----> University
  Student ----> admits ----> University

Association (Cont.)

- Role
  Company
  Boss ----> worker ----> Person
  worksIn
### Multiplicity

- **1** Exactly one
- **0..*** Zero or more
- **1..*** One or more
- **0..1** Zero or one
- **5..8** Specific range (5, 6, 7, or 8)
- **4..7,9** Combination (4, 5, 6, 7, or 9)

### Exercise

**What do you think about this class diagram?**

- **Company**
  - Boss
  - 1
  - worksIn

- **Person**
  - worker
  - 0..1
  - 1..*
More on Associations

- Multiples associations between the same two classes

- Common error: confusion between message and association name
  - Only one association is sufficient to drive the car

N-ary Associations

- N-ary association
  - Associative class
**Reflexive Association**

CourseManager

Professor

1

manages

Pre-requisite

Course

0..*

0..*

**Constraints on Associations**

Shop

gets a rebate \((sale = \text{true})\)

Buyer

\(1..*\)

\(0..n\)

sale: boolean
Exercise

- During its studies, a student stays in a room with another student (roommate). A student has some homework to do and some courses and exams to attend. For his/her exam and homework, he/she gets a grade. If this grade is more than 10, the student gets its diploma. Moreover, if he gets a grade over 12, it gets its diploma with distinction.
- Note: use only basics associations

Association / Aggregation / Composition

- **Association**
  - Objects are aware of one another so they can work together
- **Aggregation**
  - Control through one object – propagation downward
  - Protects the integrity of the configuration
  - Function as a single unit
- **Composition**
  - A part may be a member of only one configuration
  - The part cannot exist outside the configuration
Aggregation: Example

- An agent may be a part of no more than one agency, but an agent does not have to be part of an agency (0..1)
- The agency is always comprised of at least one agent but there may be any number of agents
- An agent is considered as an employee of an agency
- Every agent is constrained by the fact that he/she must have a current contract to be an employee of the agency

Composition

- Performance is a performance of an event (1..* 0..1)
Inheritance

- Inheritance
- Polymorphism
- Abstract / concrete classes

Class Diagram / Object Diagram

<table>
<thead>
<tr>
<th>Produit</th>
<th>1..*</th>
<th>1..*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Achété</td>
<td>#Acheteur</td>
</tr>
</tbody>
</table>

Cavie : Produit

Chocolat : Produit

Cheaussures : Produit

Dupont : Client

Durant : Client
Problematic

Are driven wheels associated to the engine of the car? (and not passive wheels...)
Are the driven wheels of a car associated to the engine of the same car?

Internal structure of Car
Composite Structure Diagrams

- **Purpose**: accurate description of complex systems
  - Structure of interconnected parts
  - Run-time structure of interconnected instances

- **Basics**
  - Illustrates the internal architecture of classes, and the interconnections of those classes
    - Parts
    - Ports
    - Signals
    - Interface
      - required
      - offered
    - Connectors between ports

Nested Notation vs. Composition Notation

architecture
class Car {1/1}

package coursStructureCompositeDiagram {1/1}

package part

parts

 Wheel {4}  
  
  Engine {1}  
  
  Car {1/1}  

Architecture Diagram

class Car {1/1}

w : Wheel [4]

e : Engine [1]
### Parts

**Difference between a part and an object**
- An object is a specific instance of a given class
  - An object has a kind of serial number 9983-3387-00237
- A part is abstraction of roles played by a set of objects
- At execution time, the specific object plays the role specified by the part

**Difference between a part and a class**
- A class specifies a structure independently from the context in which it may be used
- A part is a role played by an instance of a class in a given context

### Parts Multiplicity

<table>
<thead>
<tr>
<th>Initial number of instances</th>
<th>Maximum number of instances</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>aircraft : A380</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>display : LCD[1]</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>ap : AutoPilot[2]</strong></td>
<td>2</td>
</tr>
<tr>
<td><strong>call : Call[0..255]</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
Example of Parts / Objects

myNet : Net

ObjectRole Class

Recursive Nested Notation

package coursStructureCompositeDiagram

Architecture Diagram

class Car

Architecture Diagram

class Engine

Architecture Diagram

class Piston
Ports

- Represent a communication point
  - Junction between a class / object with a connector
  - No knowledge of what class / object is connected at the other end of the port

- Two types of communications
  - Between a class / object and its external environment
  - Between a class / object and its internal structure

Ports Visibility

- Port = rectangle put along its corresponding class
  - If it covers the rectangle boundary of its class
    - public port
  - If it is inside the rectangle of its class but adjacent to the boundary
    - protected port
**Interfaces**

- A port may
  - Provide an interface
    - Circle
    - Input
  - Require an interface
    - Semi circle
    - Output

- Interfaces should be declared at class diagram level
  - Declaration of a list of signals

- Signals should also be declared at class diagram level

---

**Example: At Class Diagram Level**

```
package example1

class CarSystem
  Car
  Driver

package toCar
  interface toCar
    startCar()
    stopCar()
    gearUp()
    gearDown()
    turnRight()
    turnLeft()

interface fromCar
  failure(Integer)
```

---
Example: At Composite Structure Diagram Level

Connectors

- **Connecting ports**
  - Interfaces must be “compatible”

- **Multiplicity**
  - Not specified -> peer to peer connection
  - Specified -> star connection
Example of Connector Multiplicity

\[ \text{Net} \]

\[ \begin{array}{c|c|c}
  & tx : TX[2] & c1 \\
\hline
  & 2 & 2 \\
\end{array} \]

\[ \begin{array}{c|c|c}
\hline
  & & \\
\end{array} \]

\[ \text{myNet} : \text{Net2} \]

\[ \begin{array}{c|c|c}
  & tx1 / tx : TX & c1 \\
\hline
  & c1 & \\
\end{array} \]

\[ \begin{array}{c|c|c}
  & rx1 / rx : RX & c1 \\
\hline
  & c1 & \\
\end{array} \]

\[ \begin{array}{c|c|c}
  & tx2 / tx : TX & c1 \\
\hline
  & & c1 \\
\end{array} \]

\[ \begin{array}{c|c|c}
  & rx2 / rx : RX & \\
\hline
  & & \\
\end{array} \]

Delegation

The composite class must delegate its request to one of its part

Behavior delegation. theDisplay knows that it communicates with a port, but has no knowledge about the destination part
Design

- UML class diagrams
- UML composite structure diagrams
- Exercises

Mobile Phone System

- Apart from its regular components, the mobile phone we wish to design has an integrated camera to take pictures. Pictures can be taken, visualized, deleted, sent via MMS or transferred, via a USB port, to a PC
- Make the system analysis
- Design the whole system
  - Class diagram
  - Composite structure diagram