AFI Region AIXM e-AIP Implementation Workshop
Dakar, Senegal, 3-5 October 2016

UML and XML - Introduction

Razvan GULEAC
EUROCONTROL

Aeronautical information "domains"

1. Aeronautical Information

2. MET

3. Flight object (Flight plan plus actual flight status)

4. Surveillance data (increasingly connected with the flight data)

5. Terrain data (relatively new, sometimes included in the aeronautical information domain)

6. Passenger data

7. Etc.

Data interchange - why necessary

- Computers and automated systems

  - air navigation
  - ATC
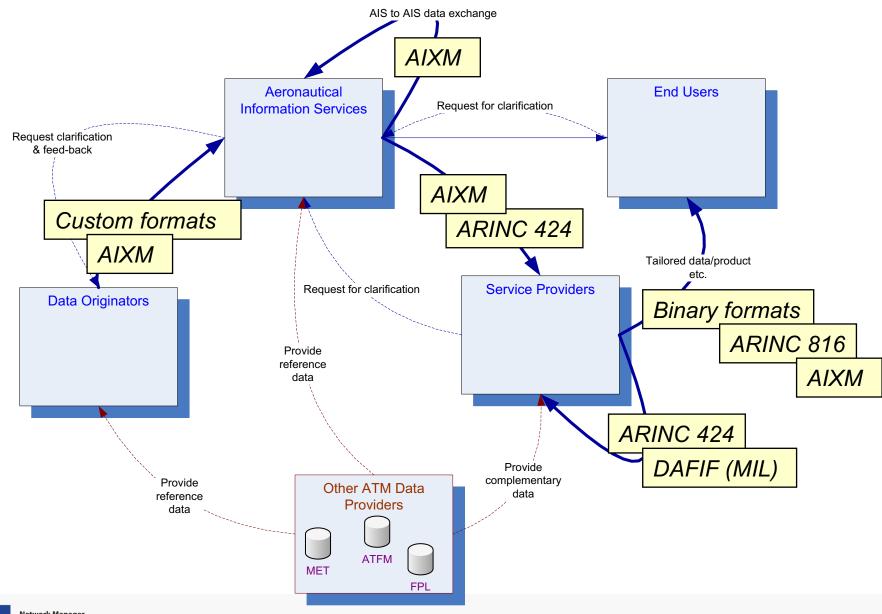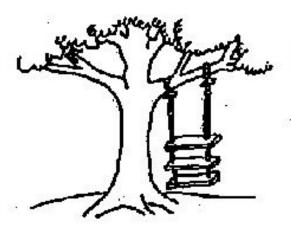 - flight preparation
  - aircraft maintenance
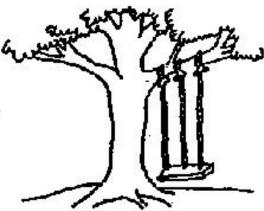
AIS Data interchange – standards

AIS to AIS data exchange

AIXM

Aeronautical Information Services

End Users

Request for clarification

Request clarification & feed-back

AIXM
ARINC 424

Custom formats

AIXM

Data Originators

Request for clarification

Provide reference data

Tailored data/product etc.

Service Providers

Binary formats

ARINC 816

AIXM

ARINC 424

DAFIF (MIL)

Provide reference data

Provide complementary data

Other ATM Data Providers

MET

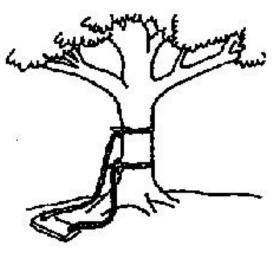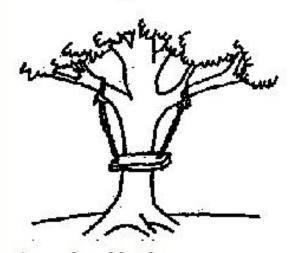ATFM

FPL

How system design works… *on a bad day* ☺

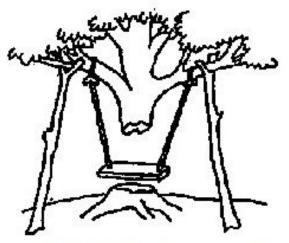As proposed by the project sponsor.

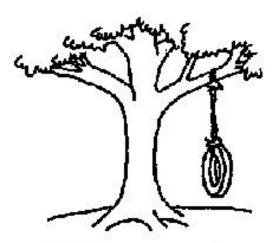As specified in the project request.

As designed by the senior analyst.

As produced by the programmers.

As installed at the user's site.

What the user wanted.

What is the solution?

- Model the Business Process

- Model the Use Cases

- Model the System

- Model the Data

Network Manager
nominated by
the European Commission

UML is a general-purpose modeling language in the field of object-oriented software engineering.

- Standard way to visualise the design of a system

- Created in 1994-1996 by Grady Booch, Ivar Jacobson and James Rumbaugh at Rational Software

- Adopted in 1997 as a standard by the Object Management Group (OMG)

- Today an ISO Standard (ISO/IEC 19501:2005)

- Keeps evolving in order to satisfy the need for managing the design and implementation of increasingly complex information management systems

What can you Model with UML?

- **Behavior Diagrams**
  - Use Case Diagram
    - frequently used during requirements gathering
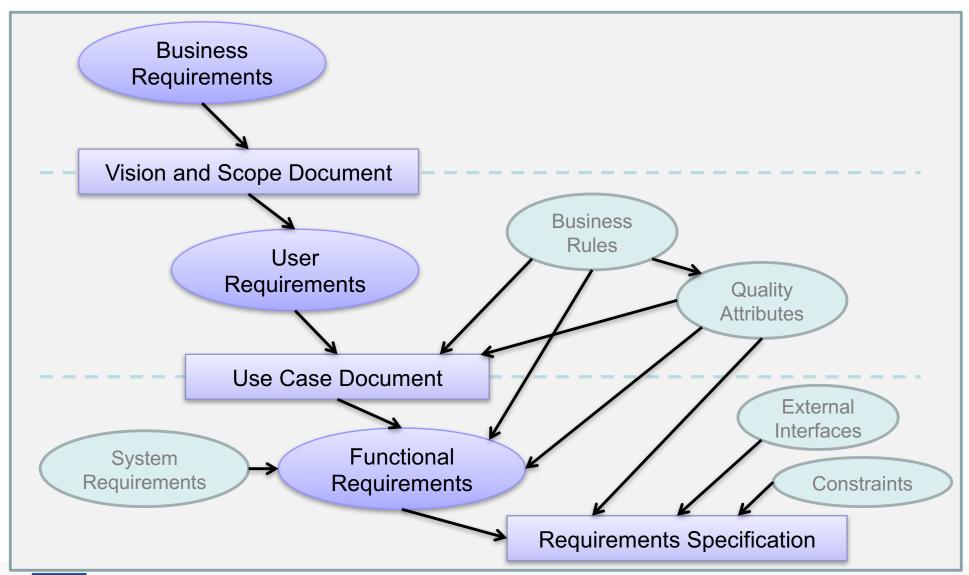 - Activity Diagram, State Machine Diagram

- **Structure Diagrams**
  - Class Diagram
  - Object Diagram, Component Diagram, Composite Structure Diagram, Package Diagram, Deployment Diagram

- **Interaction Diagrams** (derived from the more general Behavior Diagram)
  - Sequence Diagram, Communication Diagram, Timing Diagram, Interaction Overview Diagram
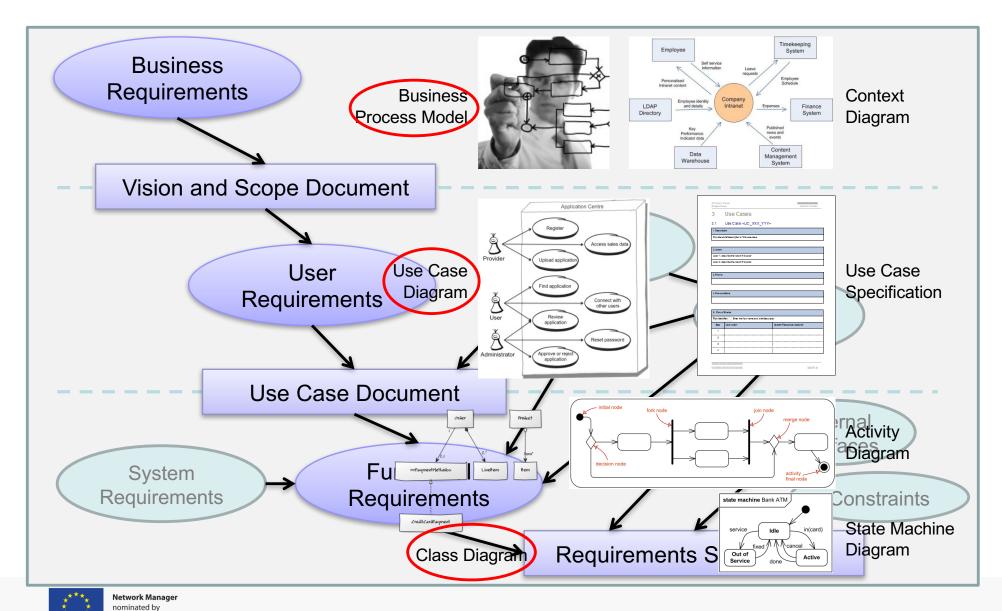
Requirements Engineering Process

Requirements Engineering Process

Business Requirements

Vision and Scope Document

Business Process Model

Context Diagram

User Requirements

Use Case Diagram

Use Case Specification

Use Case Document

System Requirements

Functional Requirements

Class Diagram

Activity Diagram

State Machine Diagram

Requirements S...
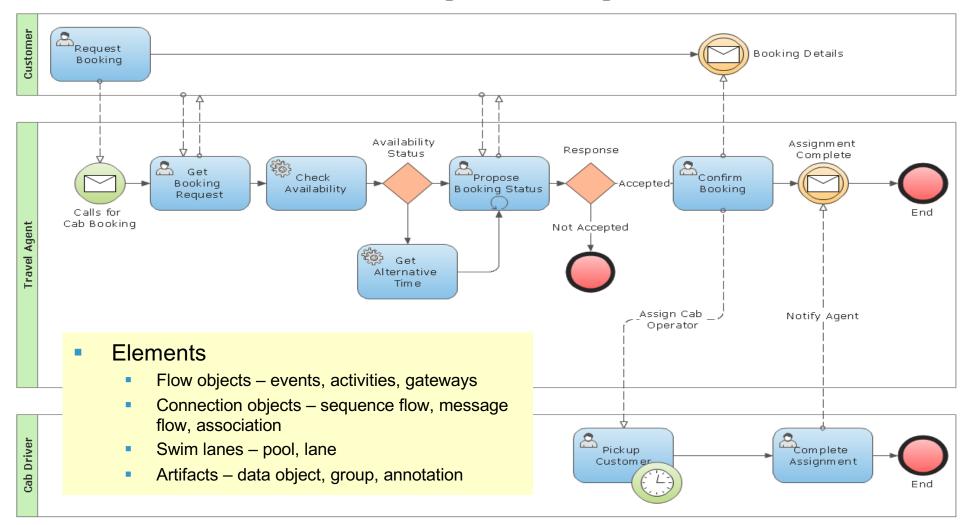
Business Process Modelling

- **Business Process Model and Notation (BPMN)**
  - a graphical notation for specifying business processes in a Business Process Diagram
    - very similar to activity diagrams from UML
    - notation that is intuitive to business users
    - able to represent complex process semantics

BPMN - example

Cab Booking Process Diagram

- Elements
    - Flow objects – events, activities, gateways
    - Connection objects – sequence flow, message flow, association
    - Swim lanes – pool, lane
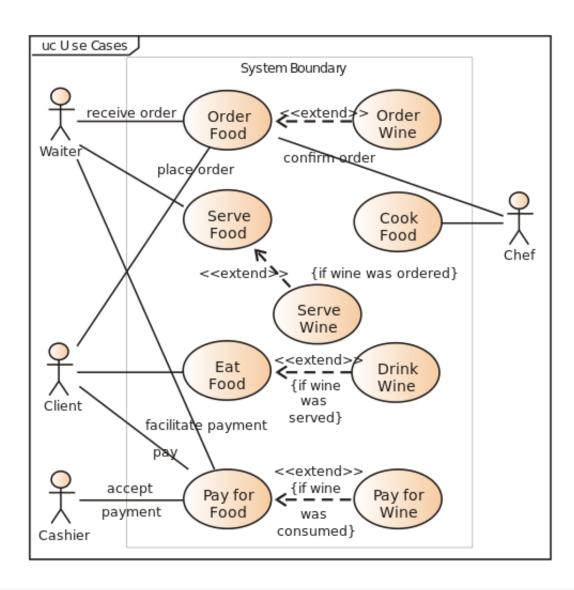   - Artifacts – data object, group, annotation

Source: http://www.conceptdraw.com/samples/business-process-diagrams-business-process-model-notation
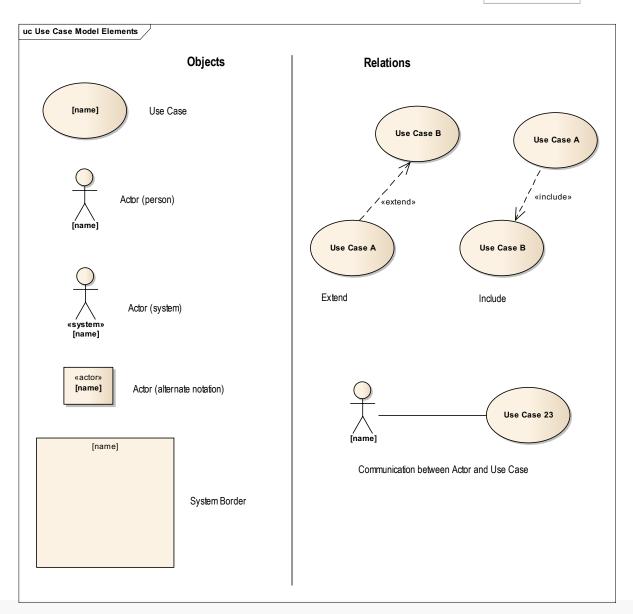
System model - Use Cases diagram

- Definition
  - "A use case is the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system" (UML 2).

- Pay attention!… people try to see some sequence in the Use Case diagrams (First I do that, then this…)
  - This is not the purpose of Use Case diagrams!

Use Case Modelling

- **Actors** are external entities (people or other systems) who interact with the system to achieve a desired goal.

- A **use case** describes how an actor uses a system to achieve a goal. It is a list of action or event steps, typically defining the interactions between a role and a system.

uc Use Case Model Elements

Objects

[name] — Use Case

[name] — Actor (person)

«system» [name] — Actor (system)

«actor» [name] — Actor (alternate notation)

[name] — System Border

Relations

Use Case B ← «extend» — Use Case A

Extend

Use Case A - - «include» → Use Case B

Include

[name] —— Use Case 23

Communication between Actor and Use Case

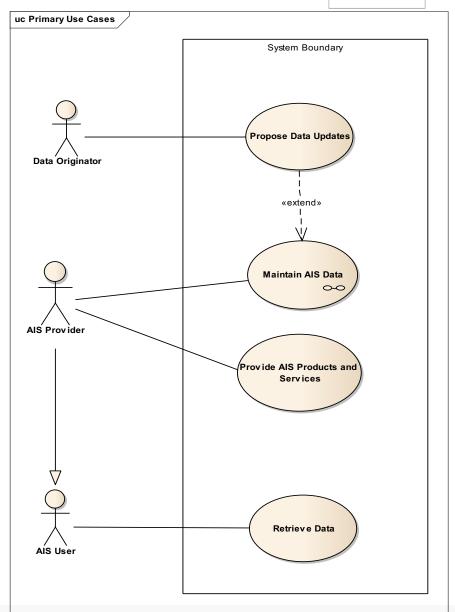EUROCONTROL

Use Case Modelling

- Almost universally accepted as a requirements gathering and documentation tool

- Captures system requirements through generalized, structured scenarios that convey how the system operates to provide value to at least one of the system's actors.

- Simple and intuitive way in which the system's behaviour is described.

- Serve as a bridge between stakeholders and the technical community.
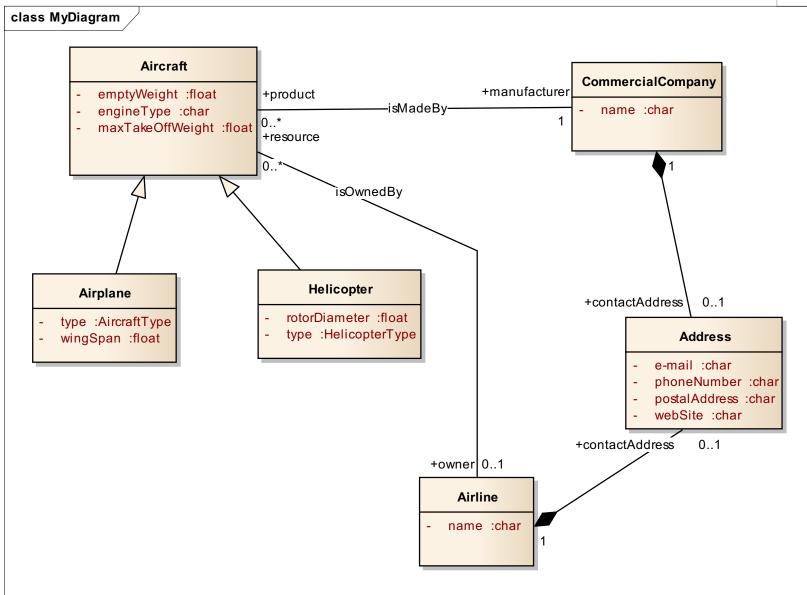
AIS system - high level UC

- The story of how the AIS system and its users interact

- Use cases are formal requirements with context and structure that clearly define the resultant value

  - Brief description...

  - Actors…

  - Basic flow (sub-Use Cases!) for example:
    - Log on
    - Find item to be updated
    - Encode update
    - Verify update
    - Submit to AIS Provider

uc Primary Use Cases

System Boundary

- Data Originator
- AIS Provider
- AIS User

Propose Data Updates

«extend»

Maintain AIS Data

Provide AIS Products and Services

Retrieve Data

Network Manager
nominated by
the European Commission

Data modelling – Class Diagrams

UML - class

Airplane

the <u>abstraction</u> of a <u>concept</u> in the <u>application domain</u>

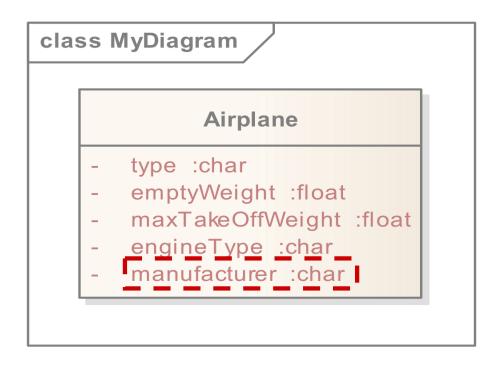# UML – class attributes

class MyDiagram

Airplane

- type :char
- emptyWeight :float
- maxTakeOffWeight :float
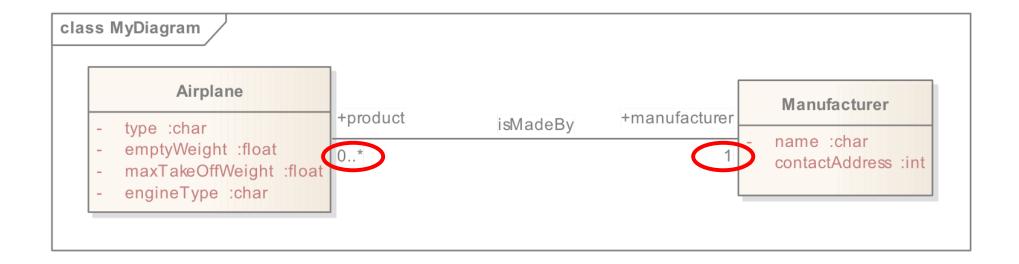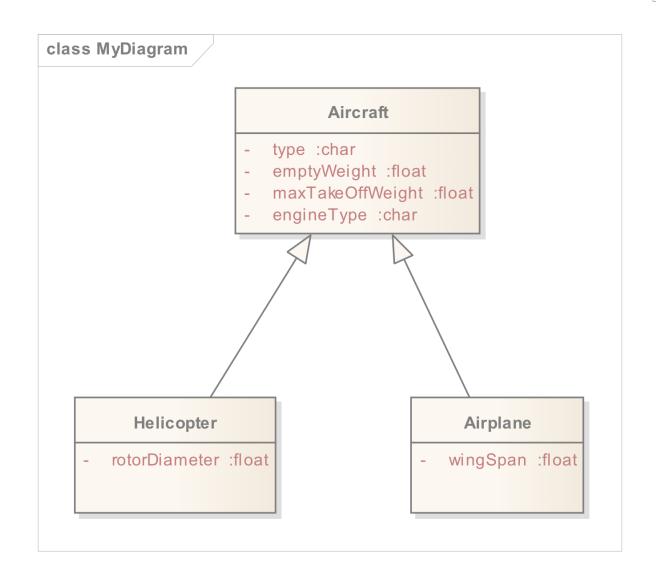- engineType :char
- manufacturer :char

UML – attribute vs association
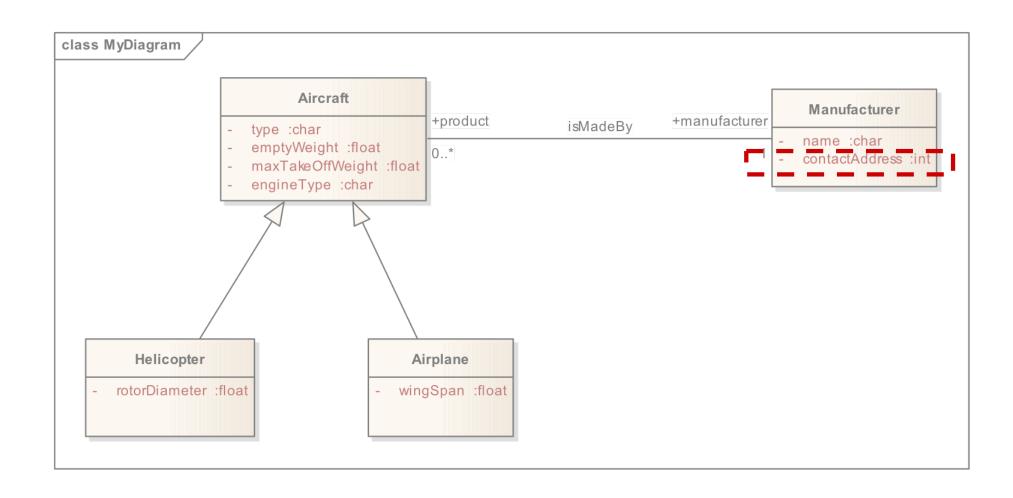
UML – class associations
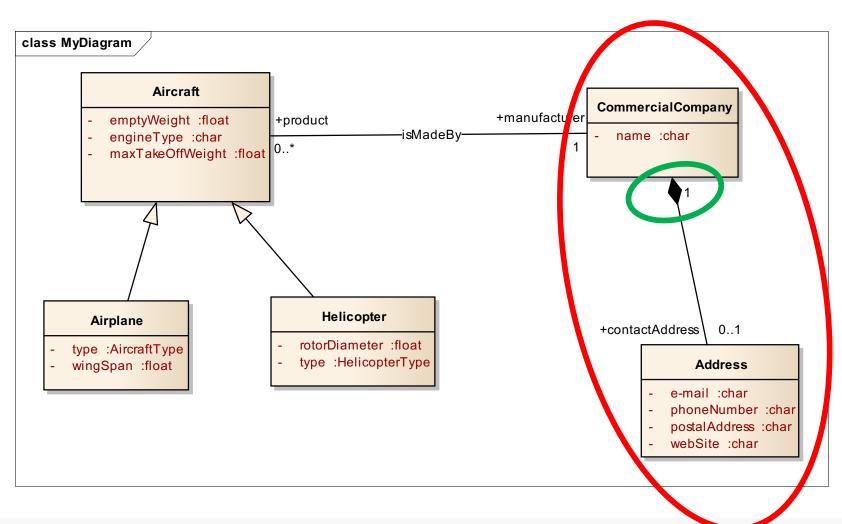
UML – Specialisation

UML – an attribute may hide an association
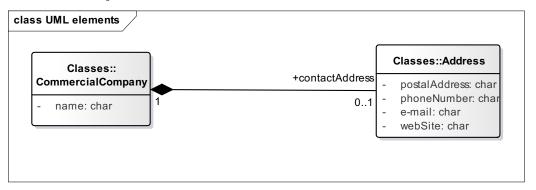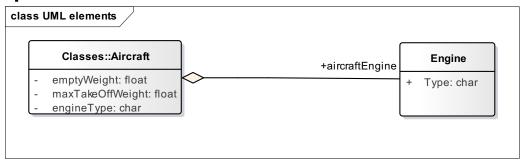
UML - Composition

Aggregation versus Composition

- Composition implies a relationship where the child cannot exist independent of the parent.
  - Address of Manufacturer
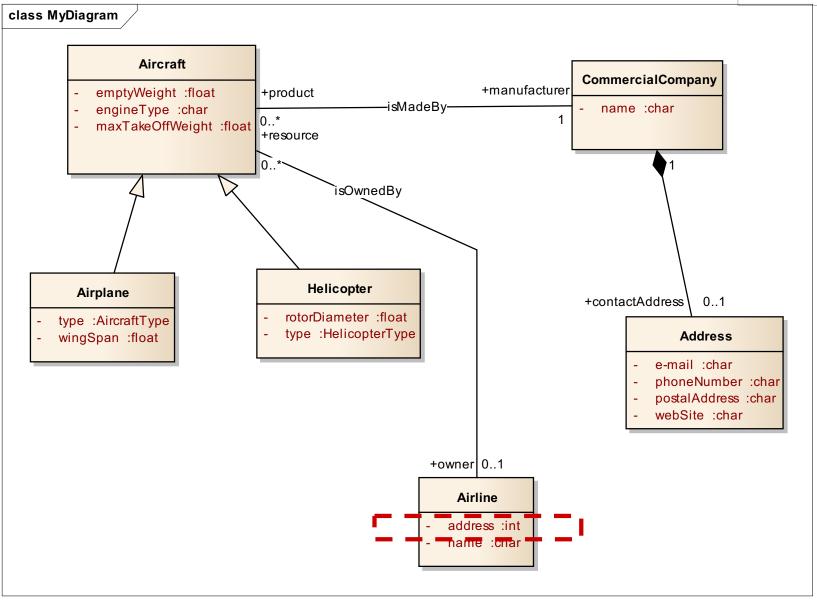 - Rooms of a House

class UML elements

Classes:: CommercialCompany		+contactAddress	Classes::Address
- name: char	1	0..1	- postalAddress: char - phoneNumber: char - e-mail: char - webSite: char

- Aggregation implies a relationship where the child can exist independently of the parent.
  - Engines of an Aircraft
    - could be re-used

class UML elements

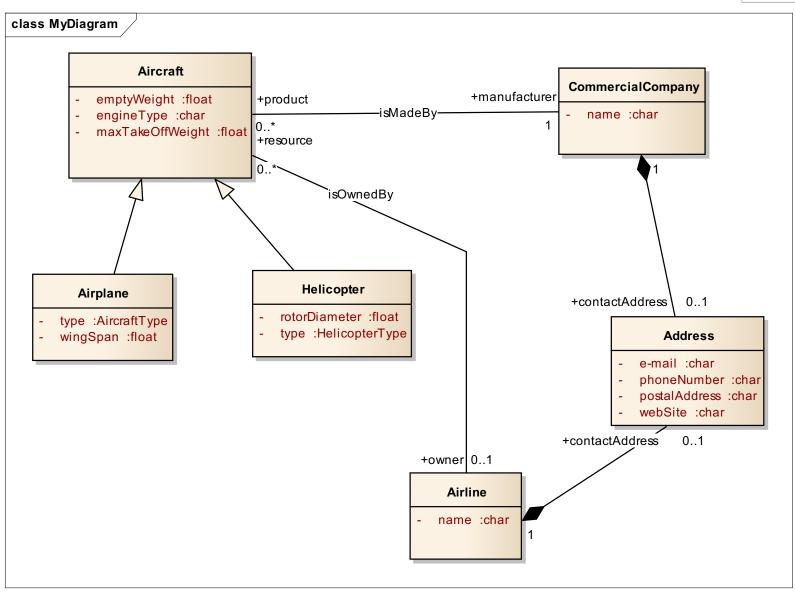| Classes::Aircraft		+aircraftEngine	Engine
- emptyWeight: float - maxTakeOffWeight: float - engineType: char			+ Type: char

UML – the Interest of Associations

UML – the interest of associations

UML – Lists of Values

class Data types

MyPackage::Aircraft

-    type :char
-    emptyWeight :float
-    maxTakeOffWeight :float
-    engineType :char
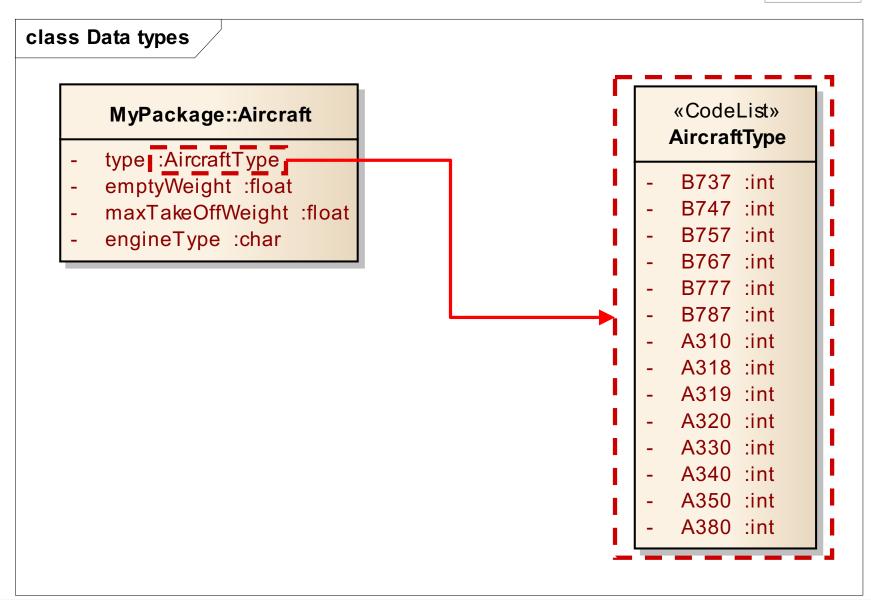
Aircraft Types

B737

B747

B757

B767

A310

A319

A380

Etc.

UML – lists of values

class Data types

```
        MyPackage::Aircraft
  -   type  :AircraftType
  -   emptyWeight  :float
  -   maxTakeOffWeight  :float
  -   engineType  :char
```

```
        «CodeList»
        AircraftType
  -   B737  :int
  -   B747  :int
  -   B757  :int
  -   B767  :int
  -   B777  :int
  -   B787  :int
  -   A310  :int
  -   A318  :int
  -   A319  :int
  -   A320  :int
  -   A330  :int
  -   A340  :int
  -   A350  :int
  -   A380  :int
```

UML– Definitions

- Correct definitions are critical for the correctness of a model
    - Classes
    - Attributes
    - Association names/roles
    - Lists of values

Network Manager
nominated by
the European Commission

UML – AIXM example

- www.aixm.aero/wiki

UML class model - usage

- Data collection requirements

- Database definition

- Data input forms

- Data verification rules
  - Data types, Further constraints

- Data exchange
  - Syntax, Semantics, etc.

Data exchange formats

- Text based versus binary
    - The mark-up languages produce a text-based format
    - Binary formats: the real life objects are "serialised", resulting is a sequence of bits (0, 1)
- Text based
    - Position based
    - Separator based
    - Markup languages
        - Languages : HTML
        - Meta-languages : SGML, XML

HTML

```html
<html>
  <head>
    <title>Enter the title of your HTML document here</title>
  </head>
  <body>
    <h1>Title</h1>
    <table border="1">
      <tr>
        <td>Name</td>
        <td>Age</td>
      </tr>
      <tr>
        <td>Dino</td>
        <td>65 Million Years</td>
      </tr>
    </table>
  </body>
</html>
```

Title

Name	Age
Dino	65 Million Years

XML

- eXtensible Markup Language
- It's a "meta language"
  - Used to build markup based exchange formats - XHTML, for example
  - designed to transport and store data.
- Plenty of good trainings on the Web
  - www.w3schools.com

XML

- http://www.w3schools.com/xml/default.asp
  - Example
  - Elements versus content
- http://www.w3schools.com/xml/xml_whatis.asp
  - The difference between HTML and XML
  - With XML you invent your own tags
  - XML is a W3C Recommendation

XML

- http://www.w3schools.com/xml/xml_tree.asp
  - Root, child elements, tree structure

- http://www.w3schools.com/xml/xml_syntax.asp
  - Elements – closing tag
  - Case sensitive
  - Properly nested
  - Attributes
  - Entity references

XML

- http://www.w3schools.com/xml/xml_attributes.asp
  - Attributes versus elements

- http://www.w3schools.com/xml/xml_schema.asp
  - Well formed XML
  - Valid XML
  - XML Schema

- http://www.w3schools.com/xml/xml_validator.asp
  - Try the syntax-check

Example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MyData>
	<Airplanes>
		<Airplane>
			<type>Airbus 318</type>
			<wingSpan uom="m">36</wingSpan>
			<emptyWeight uom="kg">30000</emptyWeight>
			<engineType>jet</engineType>
			<maxTakeOffWeight uom="kg">45000</maxTakeOffWeight>
			<manufacturer>Airbus</manufacturer>
		</Airplane>
		<Airplane>
			<type>Boeing 737</type>
			<wingSpan uom="m">36</wingSpan>
			<emptyWeight uom="kg">30000</emptyWeight>
			<engineType>jet</engineType>
			<maxTakeOffWeight uom="kg">45000</maxTakeOffWeight>
			<manufacturer>Boeing</manufacturer>
		</Airplane>
	</Airplanes>
	<Manufacturers>
		<CommercialCompany>
			<name>Airbus</name>
		</CommercialCompany>
		<CommercialCompany>
			<name>Boeing</name>
		</CommercialCompany>
	</Manufacturers>
</MyData>
```

Questions?