UML/SysML Generalities

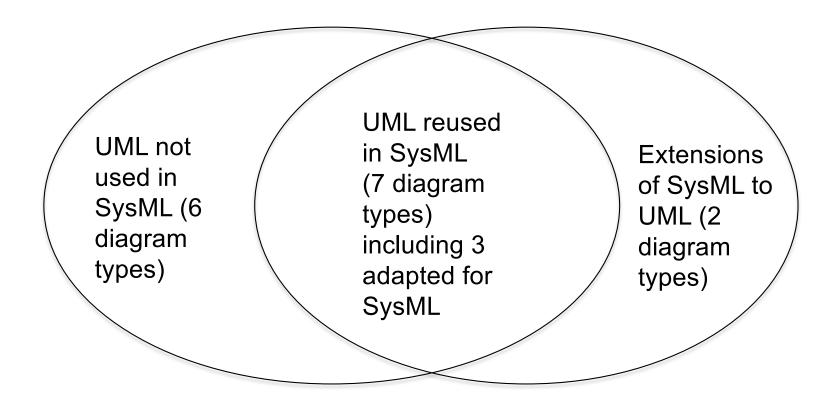
SysML

2

- At the end of the 1990's UML was already widely used for software development. But at the system level such a consensus didn't exist and there was a need of a standardized system modelling language.
- In 2003 INCOSE (International Council for System Engineering) decided to use UML (with some adaptation) for this purpose.
- The most "software" terms in UML (like "object", "class" etc.) are therefore replaced by more neutral ones (like "blocks") to appear applicable for system engineering.

walter.schon@utc.fr

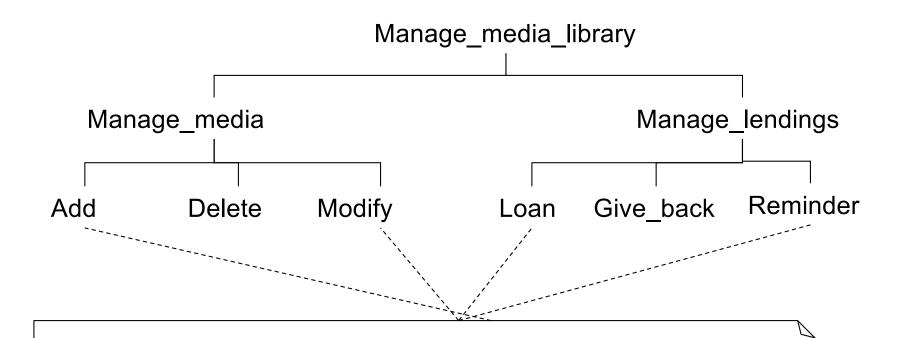
 SysML reuses an important part (but not all) UML (with some terminological changes) and adds some extensions:



Origins of UML: from functional methods to object methods

4

Functional breakdown: the most intuitive



But adding a new data type (K7, CD...) to existing ones (book...) can lead to modify many functions

History of UML: from functional to object methods

5

 An object: grouping together of state and behaviour inside a unique entity.

An	Ob	iect
<i>/</i> \111		

A state
Attribute1=
Attribute2=

A behaviour Operation1 Operation2

A Book

Borrowed = True Max_duration = 30

Borrow()
Send_remind()

...

Interest of object approach

6

Encapsulation => reusability, evolutivity

A Complex Number

x : real part y : imaginary part

Create(float, float)
Add(ComplexNumber)

. . .

walter.schon@utc.fr

A Complex Number

r: modulus θ : argument

Create(float, float)
Add(ComplexNumber)

. .

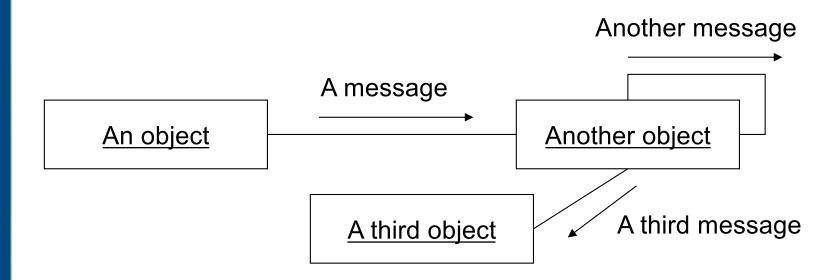
Generally only operation are visible from outside the object.

The internal representation is not visible

Structure of an object application

7

- Object application: set of objects collaborating to perform system's functions.
- Objects therefore communicate by exchanging messages



© Walter SCHÖN tous droits réservés

History of UML: the family of object methods

8

- Beginning of object languages (Smalltalk, C++): 1980's years
- Less intuitive than classical languages ("function oriented") => need of methods covering the whole development process ("object orientation" begins with requirements expression!)
- Beginning of the 1990's: multiplication of object methods (more than 50!)

History of UML: the family of object methods

9

Major ancestors:

- OMT : Object Modeling Technique (James Rumbaugh) : for General Electric
- OOD : Object Oriented Design (Grady Booch) : for the US DoD
- OOSE : Object Oriented Software Engineering (Ivar Jacobson) : for Ericsson

History of UML: the family of object methods

10 The family tree: Booch'93 OMT-2 Other 1995 Unified Method 0.8 OOSE Partners(DE C, HP, IBM...) UML 0.9 1996 January 1997 .----- UML 1.0 November 1997 -----UMI 1.1 Standardization by OMG (Object Management Group) ----- UML 1.5 March 2003 2004 ----- UML 2.0 August 2011 ----- UML 2.4.1 (current version)

© Walter SCHÖN tous droits réservés

UML today: a widespread standard

- Standardized by OMG (regroups many actors of the computer science world)
- Unifies the multiple methods of the 1990's
- Object of a wide consensus
- Covers the whole lifecycle
- Many tools (Rational Rose, Rhapsody...), even freewares (ArgoUML...)

References: books by the main authors of UML

12

- The Unified Software Delopment Process, I. Jacobson, G. Booch, J. Rumbaugh, The Object Technologies Series, Addison-Wesley 1999
- The Unified Modeling Language Reference Manual, J. Rumbaugh, I. Jacobson, G. Booch, The Object Technologies Series, Addison-Wesley 1999
- The Unified Modeling Language User Guide, G. Booch, J. Rumbaugh, I. Jacobson, The Object Technologies Series, Addison-Wesley 1999

References: other books and web sites

- Modélisation objet avec UML, P.A. Muller, N. Gaertner, Eyrolles 2000 (the reference in French)
- UML 2.0, M. Fowler, Campus Press 2004 (French translation of « UML distilled »)
- The Rational Unified process, P. Krutchen, The Object Technologies Series, Addison-Wesley 1999
- UML2 en action, P. Roques, F. Vallée Eyrolles 2004
- UML2 par la pratique, P. Roques Eyrolles 2004
- http://www.rational.com (UML Tool: Rose)
- http://www.ilogix.com (UML Tool: Rhapsody)
- http://uml.free.fr

What is UML?

14

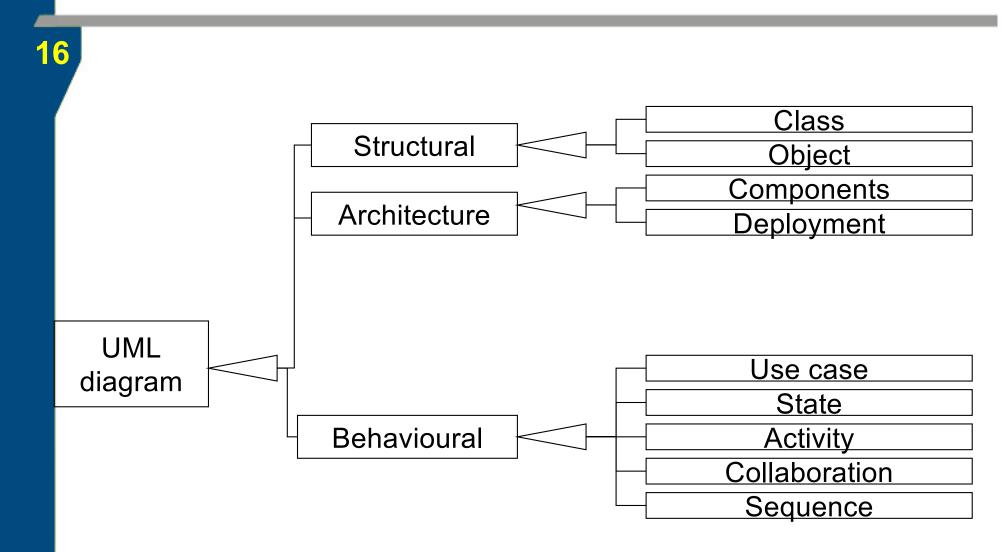
- A normalized notation allowing to express in a more or less formal manner different views of a system:
 - Structural or static views
 - Behavioural or dynamical views
 - Architectural views

What is UML

15

- A communication support allowing to limit ambiguities and misunderstandings:
- UML is formal enough to:
 - Describe itself (metamodel)
 - Allow a partial automation of software code production
- UML is not a method (does not define a process)

UML diagrams (-> 1.5): 9 types



UML 2.0 diagrams: 13 types

Class Structural Object **Components** Architecture **Deployment** <u>Package</u> Composite structure **UML** Use case diagram State Behavioural **Activity** Communication <u>Sequence</u> **Timing** Interaction overview

© Walter SCHÖN tous droits réservés

UML diagrams

18

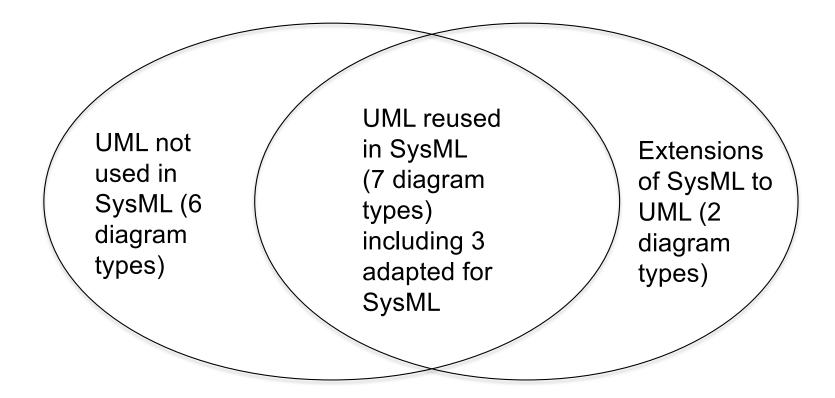
- UML is flexible and extensible: mixing different types of diagrams is possible, and extending the UML metamodel by new model elements (mainly by using stereotypes) is also possible.
- Nobody uses (or even understands) the whole UML: you have to find the subset of UML that is useful for you.
- Practically class diagrams, sequence diagrams and use case diagrams are widely used and quasi-mandatory in UML modelling.

SysML

19

walter.schon@utc.fr

SysML reuses an important part (but not all) UML (with some terminological changes) and adds some extensions:



© Walter SCHÖN tous droits réservés

SysML

20

- SysML uses 9 diagram types, each concerning a particular aspect of the system model:
- Four behavioural diagrams: sequence, state, use case (reused from UML) and activity (adapted from UML)
- Four structural diagrams: block definition (adaptation of UML class diagram), internal block (adaptation of UML composite structure diagram), parametric (specific of SysML: specialization of the Internal Block Diagramm) and package (reused from UML)
- A transverse and particularly important diagram: the requirement diagram (specific to SysML).

Differences SysML/UML2

21

- SysML doesn't use 6 UML2 diagrams (object, component, deployment, interaction overview, communication and timing).
- SysML adapts 3 UML2 diagrams (class becoming block, composite structure becoming internal block, activity being also adapted).
- SysML adds 2 specific diagram types (requirements and parametric)
- Summary 13 (UML2 diagrams) -6 (not used UML2 diagrams) +2 (specific diagrams) = 9 (SysML diagrams).

SysML diagrams: 9 types

22 Class -> Block Structural Object **Components** Architecture **Deployment** Package -> Package Requirement Composite structure -> Internal Block SysML **Parametric** Use case ->Use case Diagram State -> State Behavioural Activity -> Activity Communication <u>Sequence</u> **Timing** Interaction overview

© Walter SCHÖN tous droits réservés