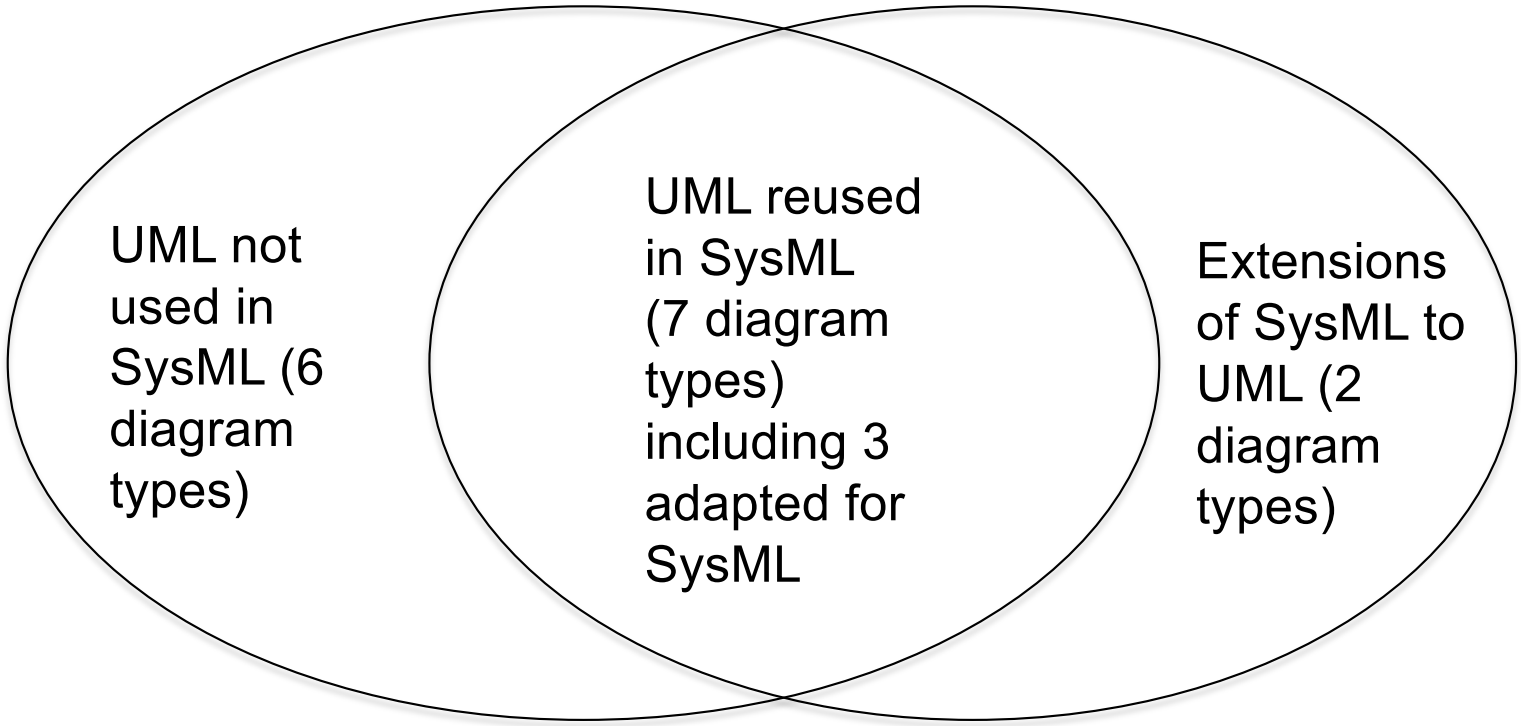
UML/SysML Generalities

SysML

- 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.

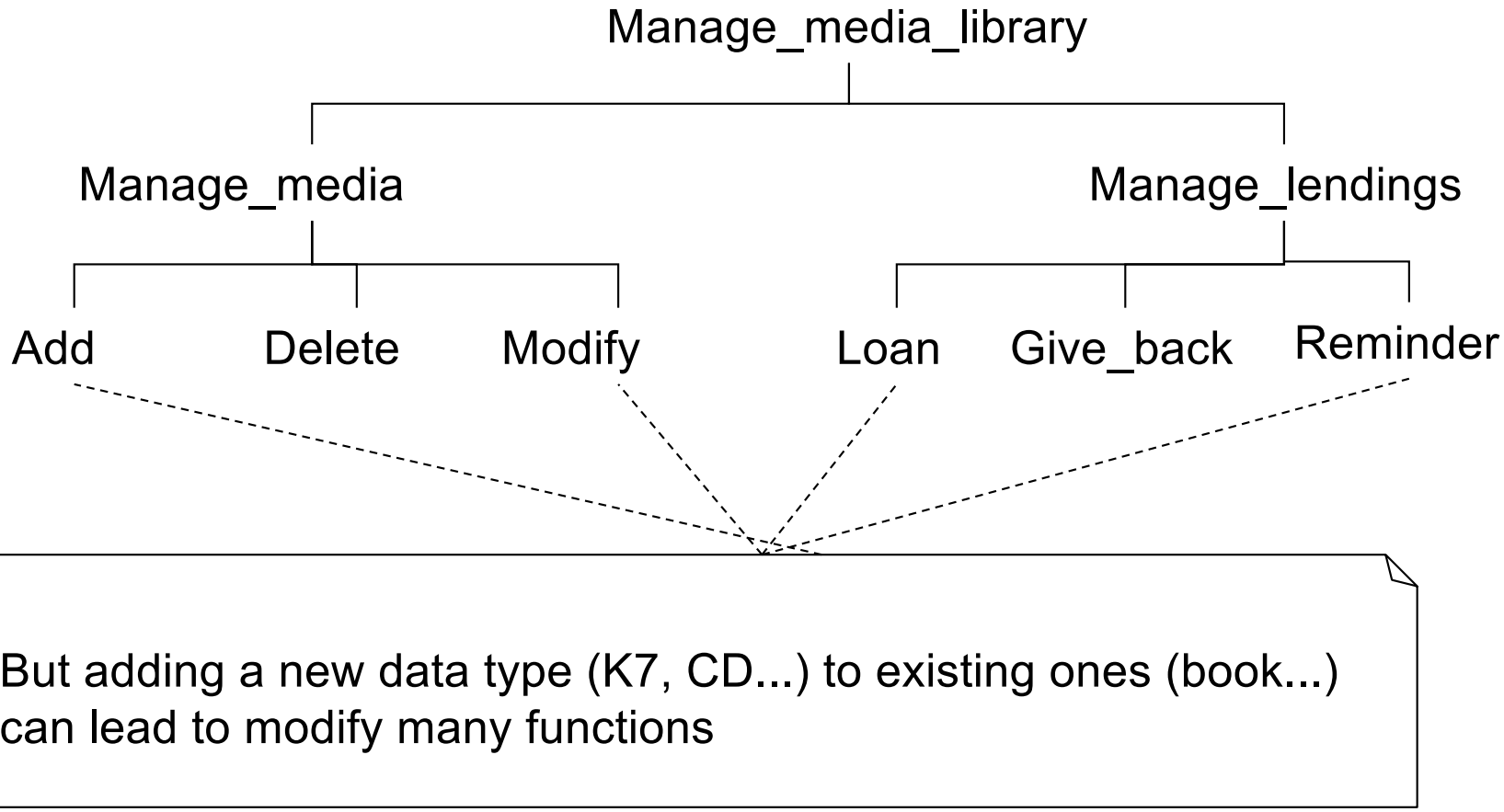
SysML

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



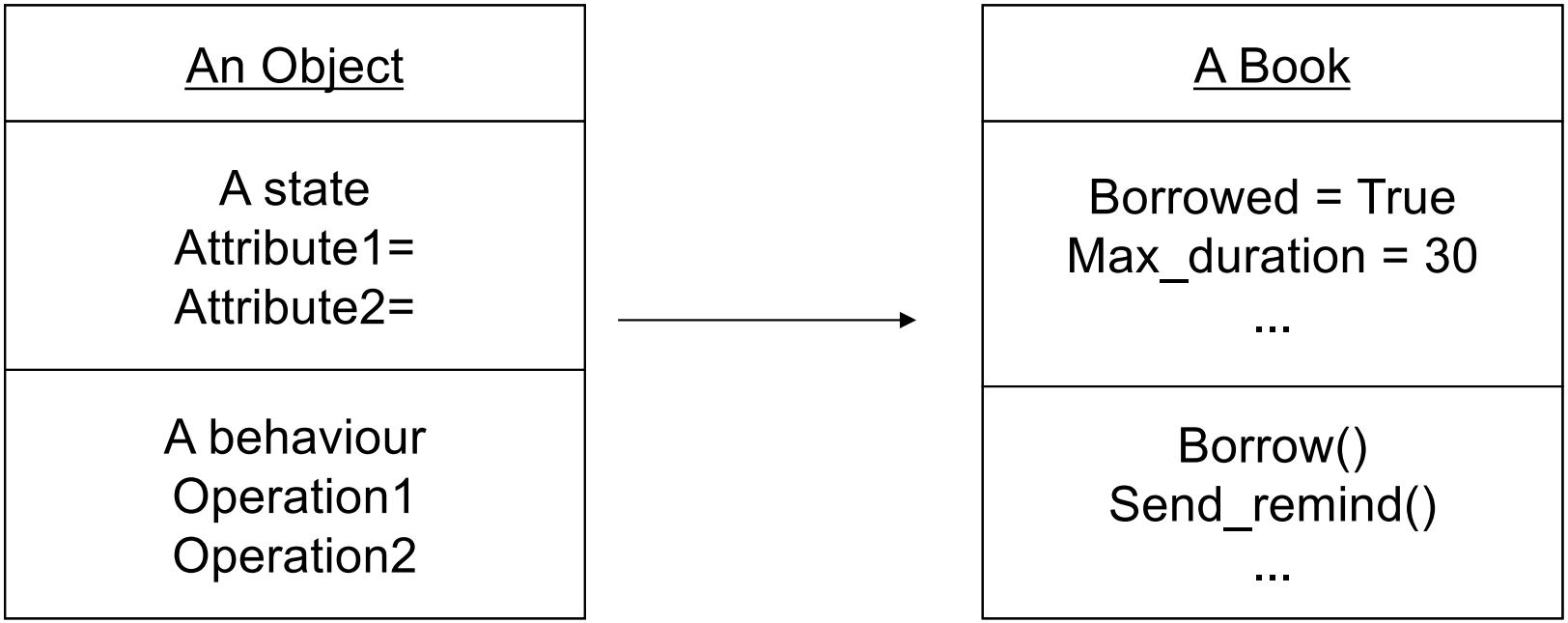
Origins of UML: from functional methods to object methods

- Functional breakdown : the most intuitive



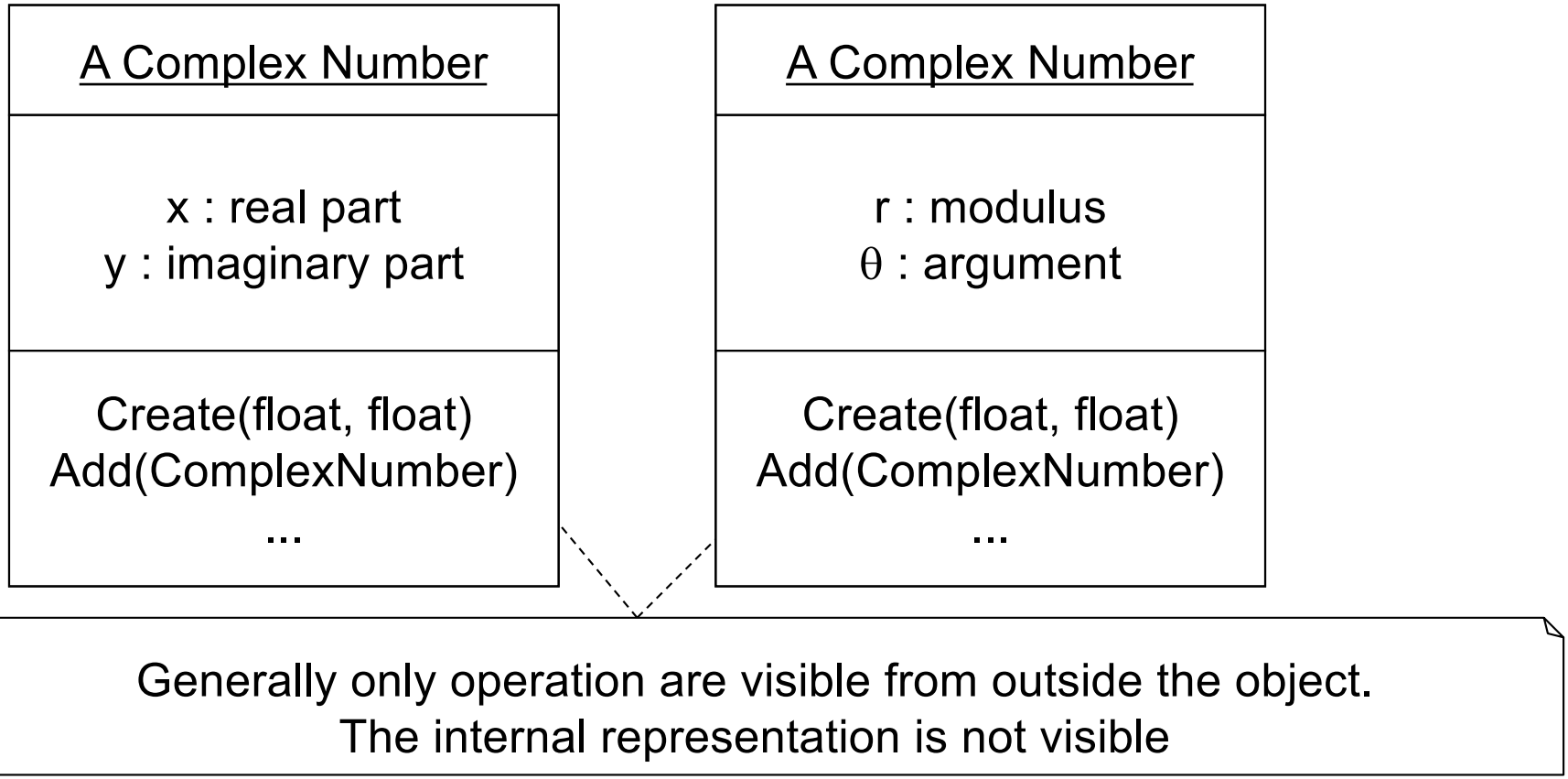
History of UML: from functional to object methods

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



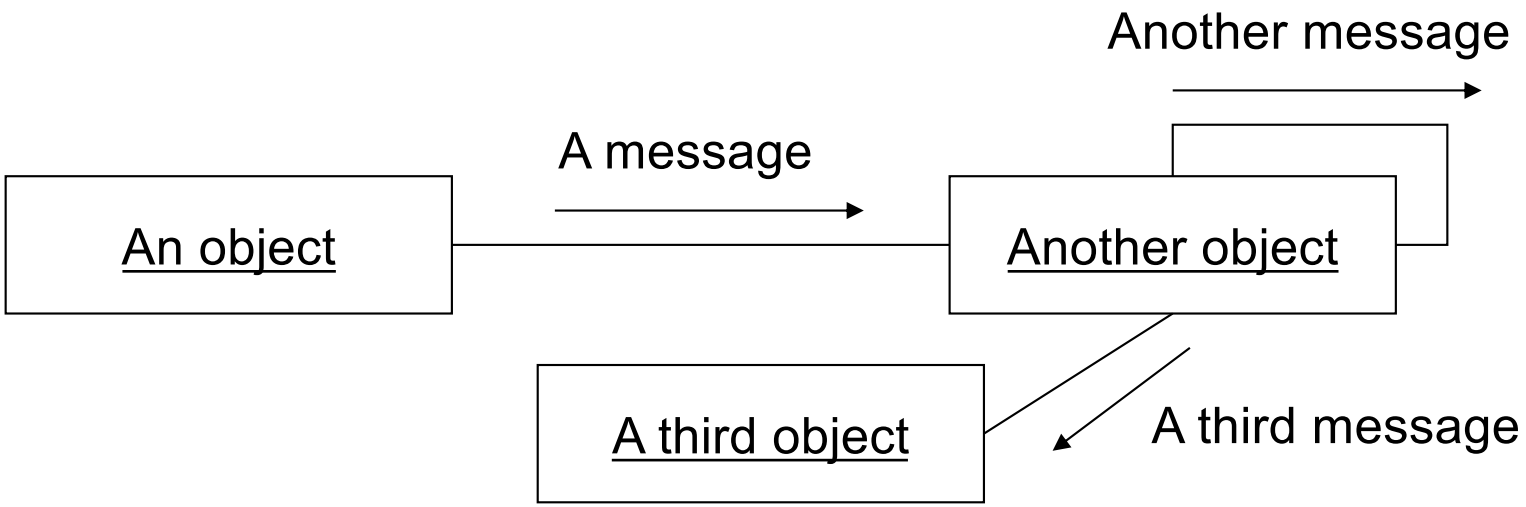
Interest of object approach

- Encapsulation => reusability, evolutivity



Structure of an object application

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



History of UML: the family of object methods

- 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 the1990's : multiplication of object methods (more than 50 !)

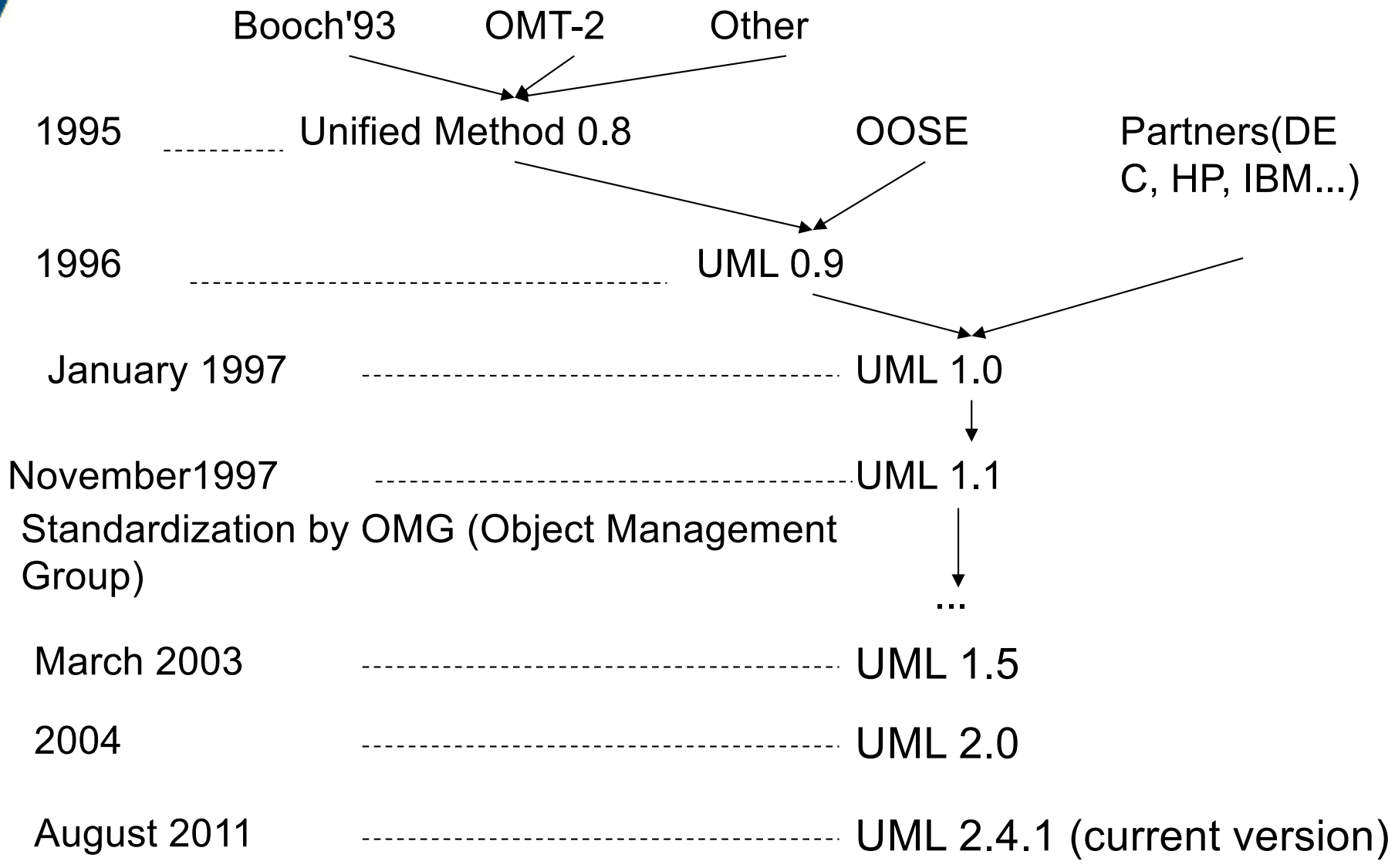
History of UML: the family of object methods

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

- The family tree:



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

- 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

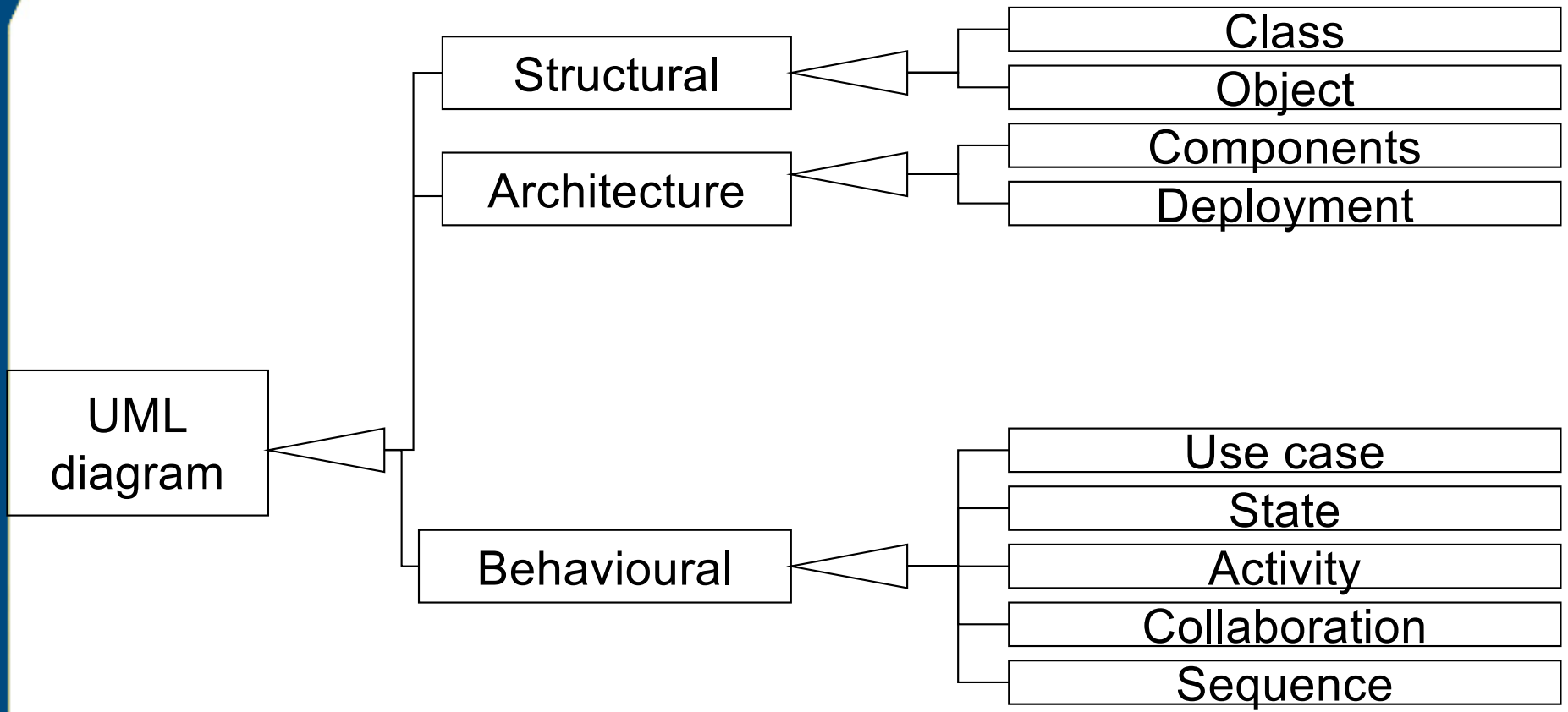
What is UML?

- 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

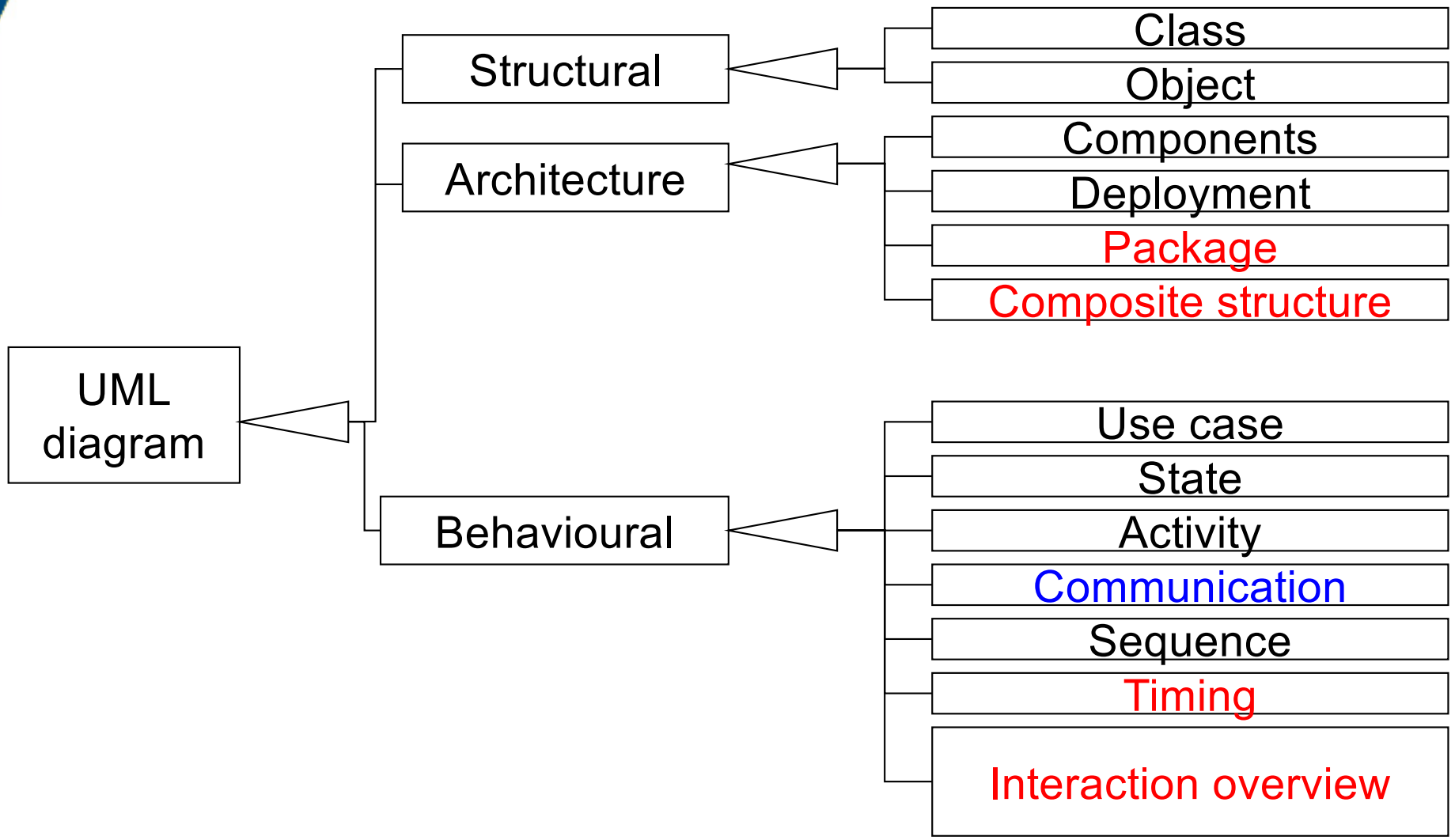
What is UML

- 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

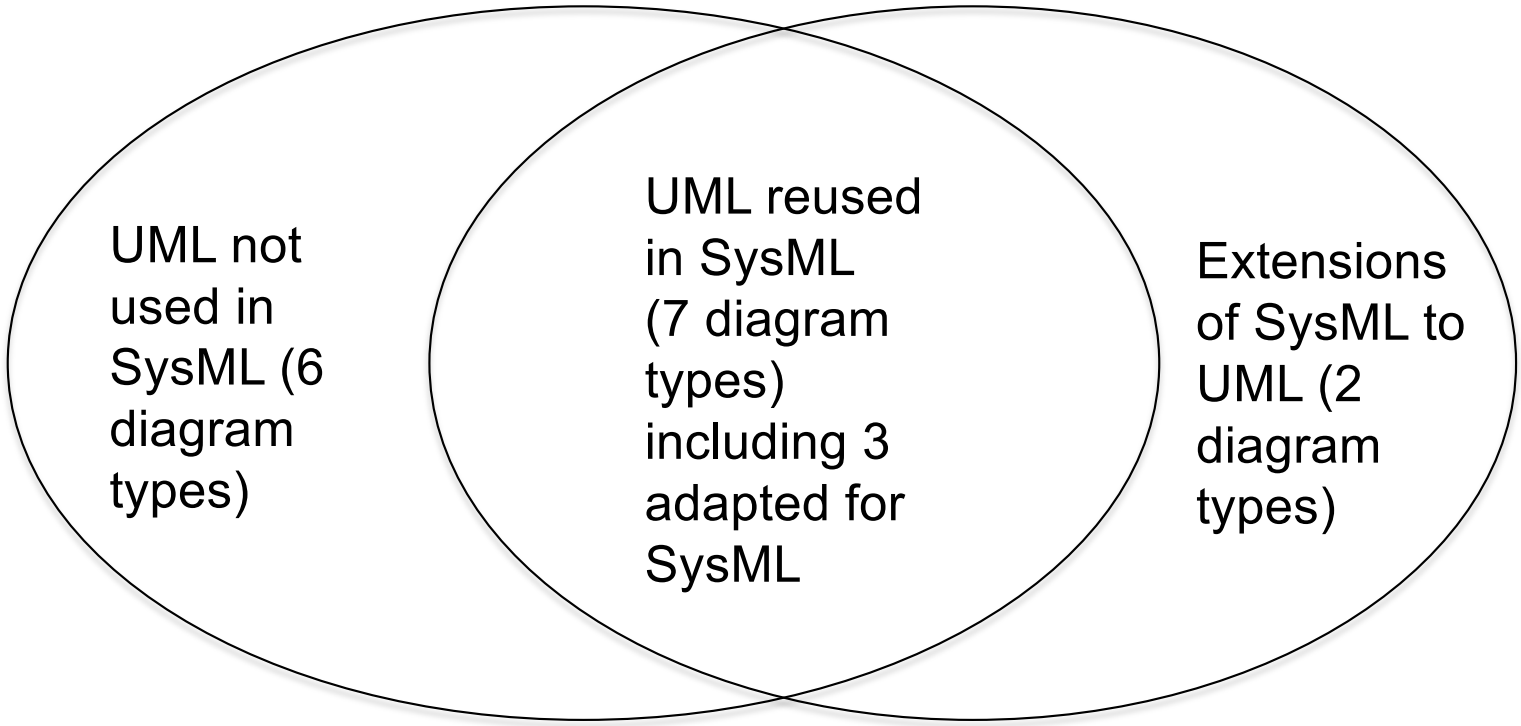


UML diagrams

- 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

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



SysML

- 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

- 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