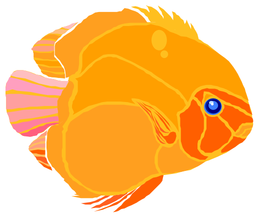
# V. Object-Oriented Modeling

**Object-Oriented Analysis Techniques**
**Coad's OOA Technique**
**Short History**
**Terminological Comparison**
**Remarks**



## Object-Oriented Analysis

- *OOA is a collection of like-minded requirements modeling and analysis techniques for software systems.*
- *Proposed in the late '80s, such techniques have been influenced primarily by object-oriented programming, but also semantic data models and semantic networks.*
- ***Basic idea:*** *streamline software development by making* ***objects***, ***classes***, ***methods*** *and the like the atomic units out of which one builds requirements, designs and implementations.*

## Basic Concepts

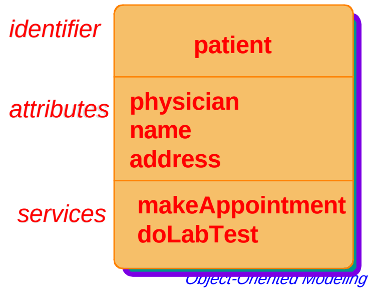
- Such techniques focus on the ***things*** that exist within the application domain, model them with ***objects*****.**
- These techniques use ***classification***, ***generalization***, ***aggregation*** to structure object assemblies.
- ***Actions*** (services/activities) are associated with objects.
- ***State changes*** are effected by actions performed on objects.

## Origins of Object-Oriented Analysis

- ***Object-Oriented Programming*** -- tries to adopt as many of the O-O programming features to O-O design and analysis [Booch86]
- ***Database Design*** -- adopts semantic data modeling ideas, including E-R diagrams and generalization, aggregation, classification [Chen76]
- ***Structured Analysis*** -- including SADT and other structured analysis techniques [Ross77]
- ***Knowledge Representation*** -- uses ideas from frame-based and semantic network representations [Borgida85]

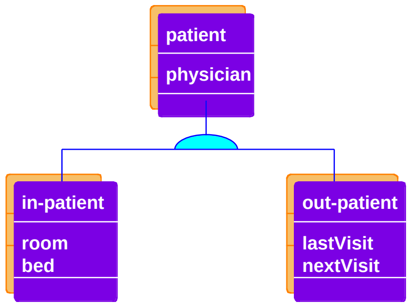
## Coad's Object-Oriented Analysis

- Proposed by Peter Coad [Coad91].
- An ***object*** is defined as a real world entity related to the problem domain, with "crisply defined boundaries".
- Objects are encapsulated with attributes and behaviour.
- OOA offers five kinds of concepts: ***objects***, ***attributes***, ***structures***, ***services*** and ***subjects***.



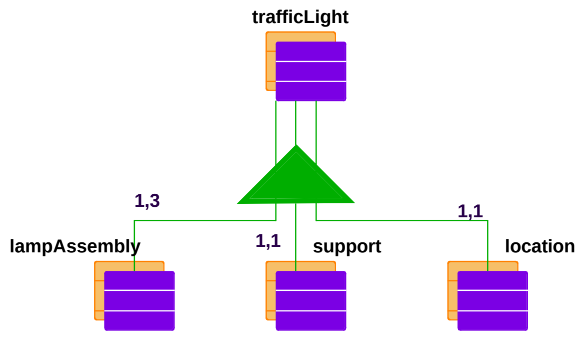
## Gen-Spec Structures

Gen-Spec (generalization/specialization) structures organize classes into taxonomies. ***Patients*** are either ***in-patients*** or ***out-patients***. The ***physician*** attribute of ***patients*** is inherited by both ***in-patients*** and ***out-patients***.



## Whole-Part Structures

Whole-Part structures describe an object as an assembly of other objects. A ***traffic light*** consists of 0 to 3 ***lampAssemblies***, a single ***support*** and a single ***location***.

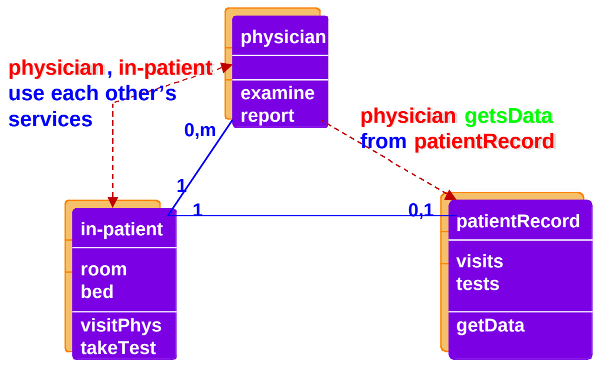


## Services

- *Objects provide* ***services*** *to other objects in their environments. For example, a physician object may provide services* ***examine, report.***
- *Coad distinguishes three types of services:*
  - ***Occurence services,*** *whereby objects are created, destroyed, changed,...;*
  - ***Calculate services****, where an object performs a calculation for some other object;*
  - ***Monitor services****, where an object is monitoring some process to see if some condition applies;*
- *A special notation is used (dashed-line arrow) to indicate that an object is using services from another object.*

***OOA views the world with Smalltalk glasses...***

## Services and Relationships



## Methodology

- Identify objects and ***classes*** (i.e., generic objects)
- Identify ***structures*** and build generalization, aggregation hierarchies.
- Define ***subjects***. These partition all the objects and classes of an object model into subject layers, which represent the application from a particular perspective. Often whole Gen-Spec or Part-Whole structures are grouped under one subject.
- Identify information that should be associated with each object. Place ***attributes*** at the right structural level.
- Define ***services*** for each class.

## Terminology

| OOA | OOSE (Jacobson) | OOD (Booch) | OMT (Rumbaugh) |
|---|---|---|---|
| | | Metaclass | |
| Object | Instance | Object | Object |
| Gen-Spec | Inheritance | inherits | Generalization |
| Whole-Part | Consists-of | | Aggregation |
| Instance conn. | Acquaintance | | Link |
| Message | Stimuli | Message | Event |
| Message conn. | Communication | | |
| Attribute | Attribute | | Attribute |
| Service | Operation | | Operation |
| Subject | ~View (subsystem) | | Sheet |
| (Execution thread) | Use case | | ~Scenario |
| (User) | Actor | | |

## What is Good About OOA?

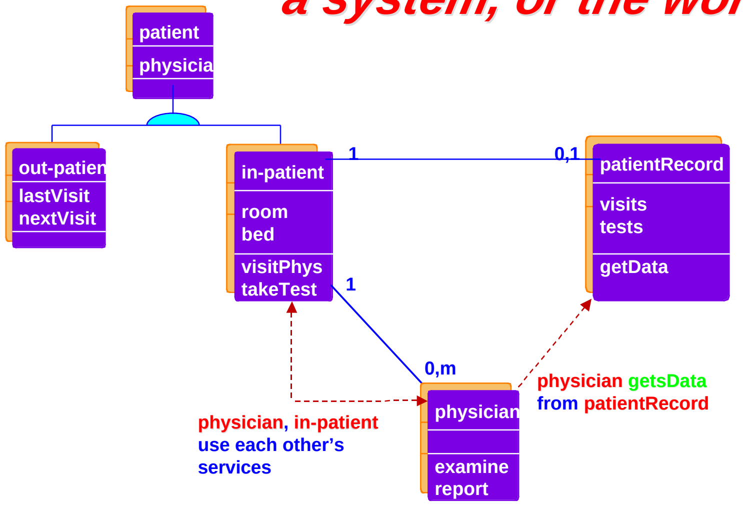
*Advances the state-of-practice! Earlier prevalent modeling techniques, such as structured analysis (SA), data flow diagrams (DFD), entity-relationship diagrams (ERD),... were:*

- **fragmented**, *e.g., use of data flow and entity-relationship diagrams*
- **weakly structured**: *see DFD and ERD*
- **informal** : *see DFD*
- *based on an* **outdated programming paradigm** *(structured programming)*

## What is Questionable About OOA?

- Is OOA intended for modeling software, or applications? If only the former, then it is not meant for requirements modeling!
- Its ontological assumptions. Who says that objects, relationships, services and the like constitute a good set of primitive concepts for modeling the real world? ...organizations? ...people? ...industrial processes?
- The promise of easier design and implementation won't work for large systems where requirements, design and implementation have drastically different architectures and are based on very different concepts.

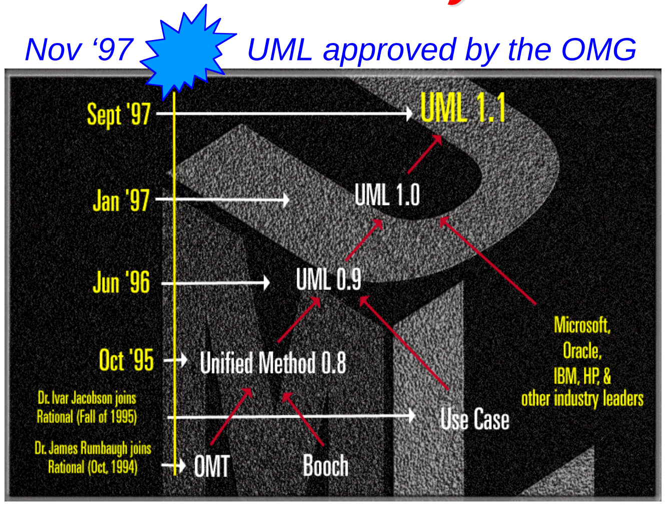
## What does this model, a system, or the world?



## The Unified Modeling Language (UML)

- Booch and Rumbaugh started working towards a unified modelling language (UML) in 1994 under the auspices of Rational Inc. They were later joined by Jacobson.
- UML only offers a notation, not a methodology for modeling (as various OOA techniques do).
- Combines Jacobson's use cases with Booch and Rumbaugh concepts for object modeling, along with statecharts.
- UML has been adopted by the Object Management Group (OMG) as an (object) modeling standard. OMG UML 1.0 is the first version of this new modeling standard.

## UML History


Nov '97
UML approved by the OMG
Sept '97
UML 1.1
Jan '97
UML 1.0
Jun '96
UML 0.9
Oct '95
Unified Method 0.8
Microsoft, Oracle, IBM, HP & other industry leaders
Dr. Ivar Jacobson joins Rational (Fall of 1995)
Use Case
Dr. James Rumbaugh joins Rational (Oct, 1994)
OMT
Booch

## References

- [Booch86] Booch, G., "Object-Oriented Development", *IEEE Transactions on Software Engineering 12(2)*, February 1986.
- [Booch94] Booch, G., *Object-Oriented Analysis and Design*, Benjamin-Cummings, 1994 (2nd edition).
- [Coad91] Coad, P. and Yourdon, E., *Object-Oriented Analysis*, Prentice Hall, 1991.
- [Jacobson92] Jacobson, I., Christerson, M., Jonsson, P., Overgaard, G., *Object-Oriented Software Engineering*, Addison-Wesley, 1992.
- [Martin93] Martin, J., *Object-Oriented Analysis and Design*, Prentice-Hall, 1993.
- [Rumbaugh91] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., *Object-Oriented Modeling and Design*, Prentice-Hall, 1991.
- [UML00] http://www.rational.com