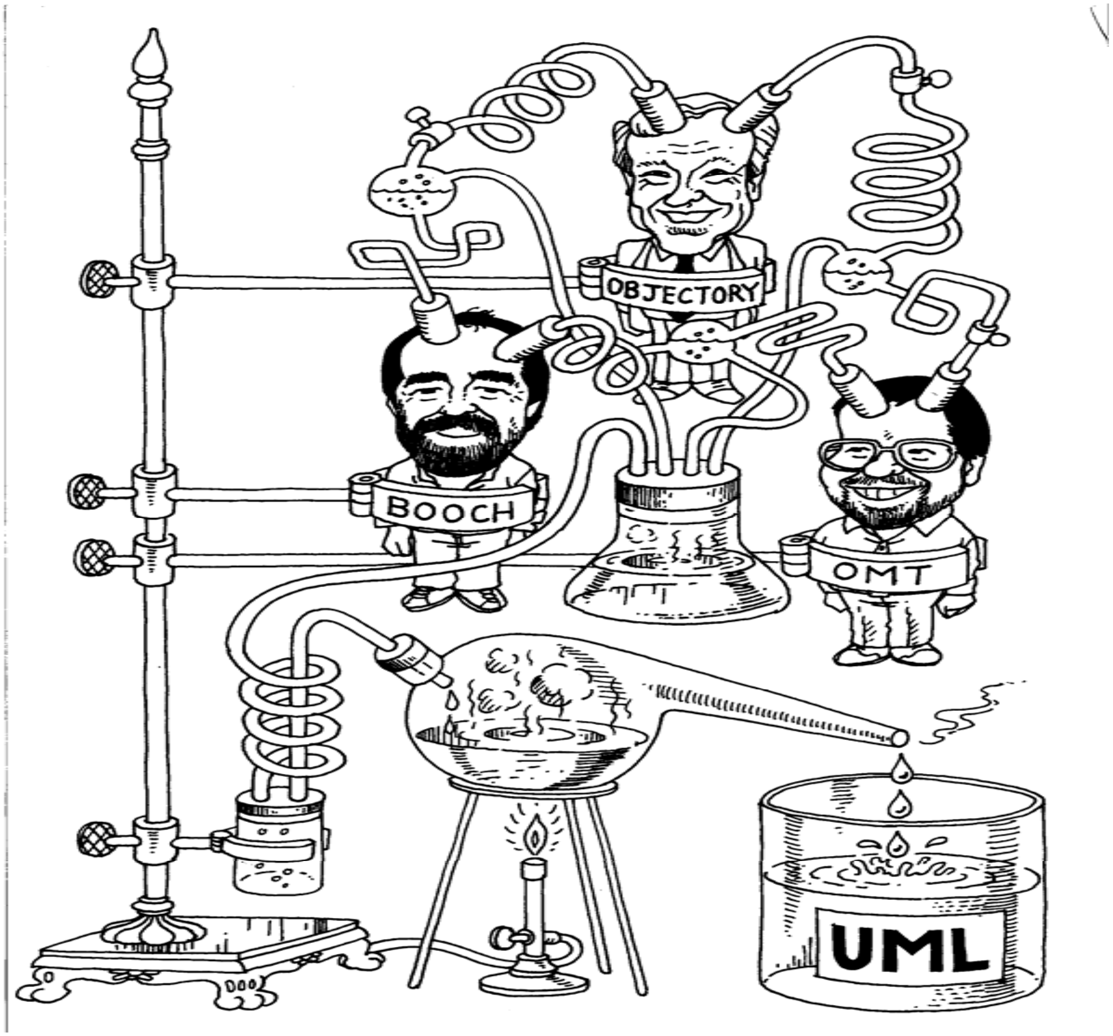
IBM Software Group

UML 2.0 profiles for modeling real-time and quality of service

Sky Matthews

IBM Rational

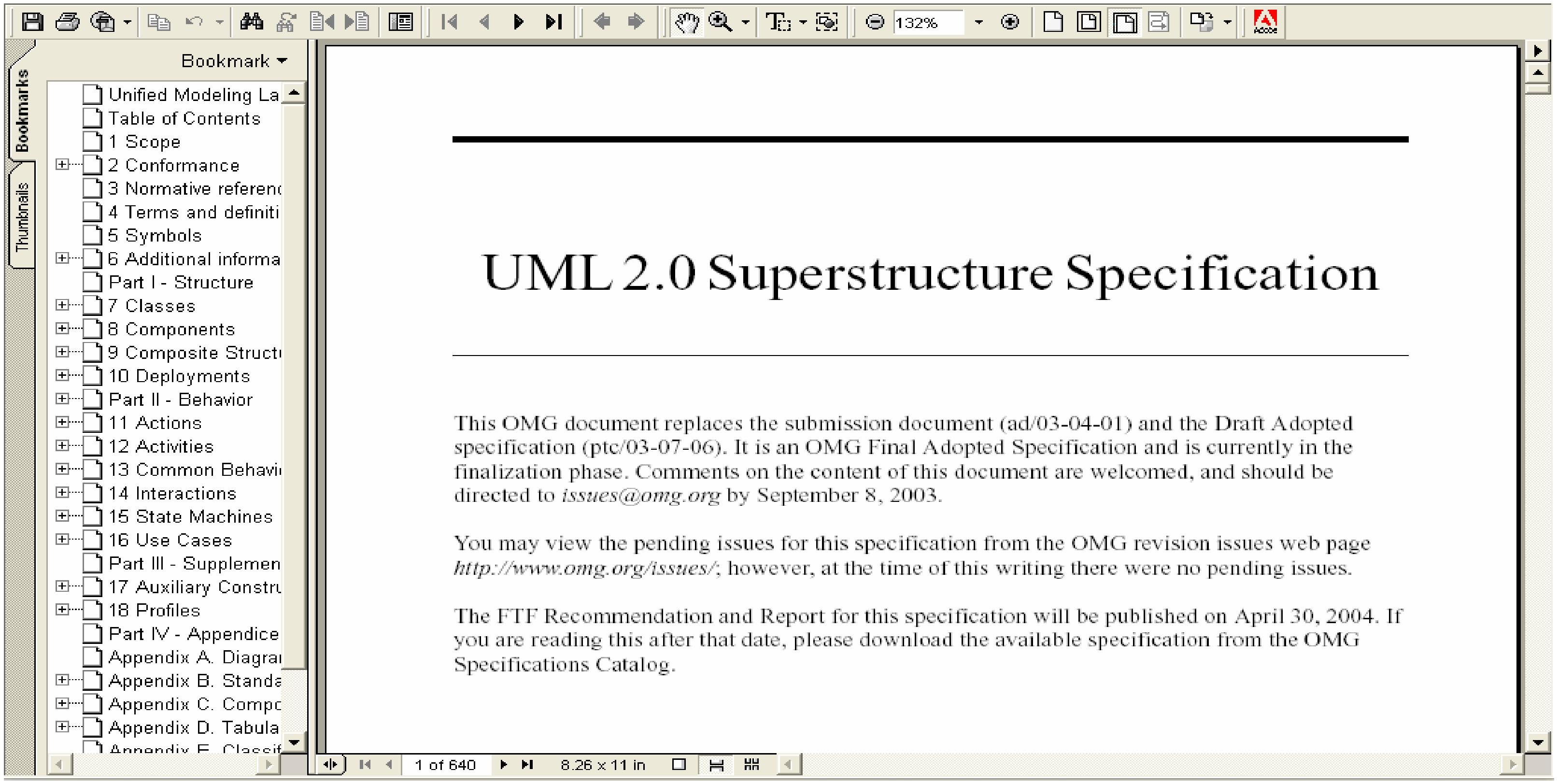
The Unified Modeling Language



- The UML is the standard language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system
- Standardized by OMG in 1997 following “OO method wars”
- UML 2.0 specification now in finalization stage

The UML 2.0 Specification

- Can be downloaded from http://www.omg.org/cgi-bin/doc?ptc/2003-08-02



UML 2.0 Superstructure Specification

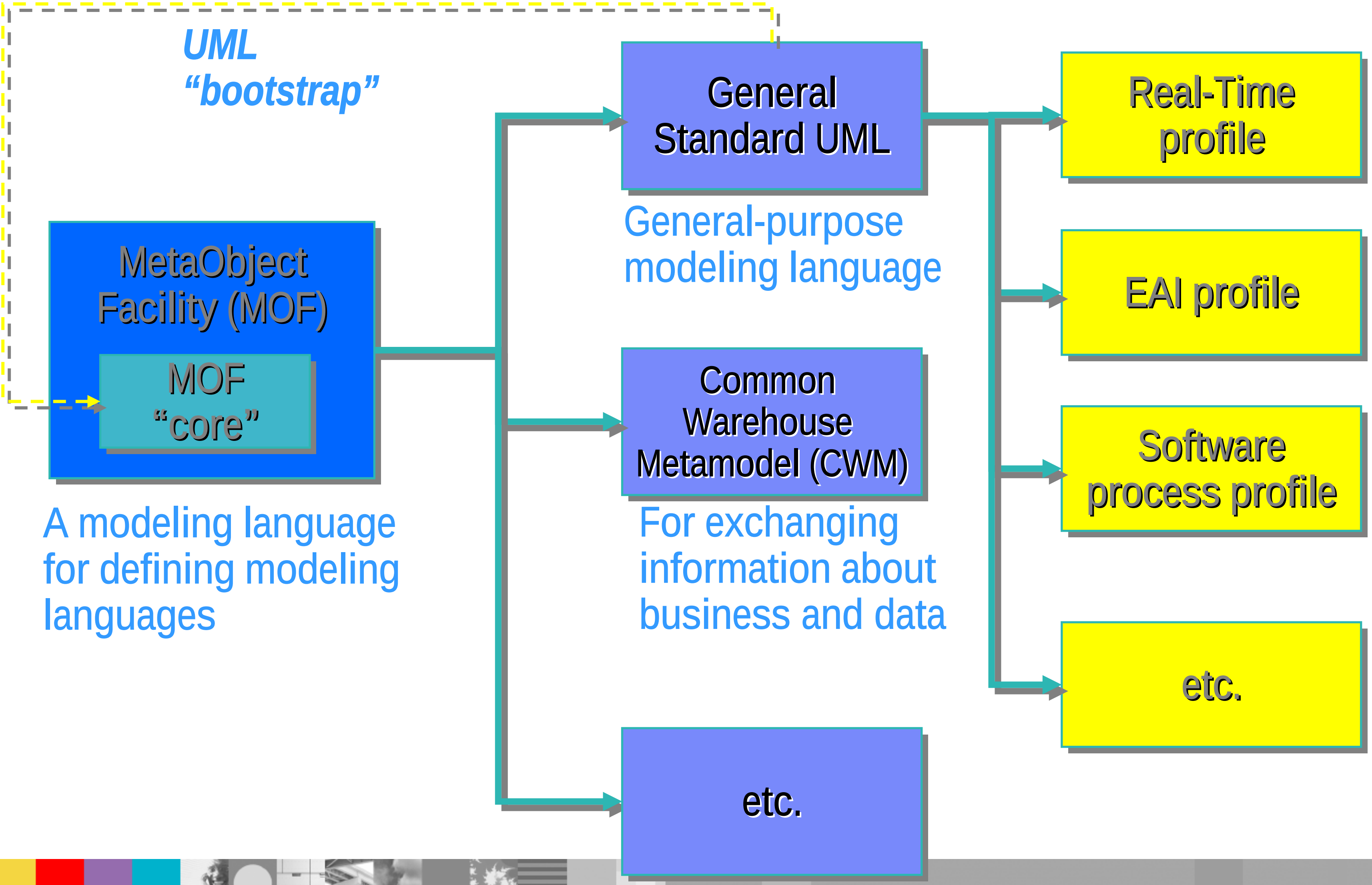
This OMG document replaces the submission document (ad/03-04-01) and the Draft Adopted specification (ptc/03-07-06). It is an OMG Final Adopted Specification and is currently in the finalization phase. Comments on the content of this document are welcomed, and should be directed to *issues@omg.org* by September 8, 2003.

You may view the pending issues for this specification from the OMG revision issues web page *http://www.omg.org/issues/*; however, at the time of this writing there were no pending issues.

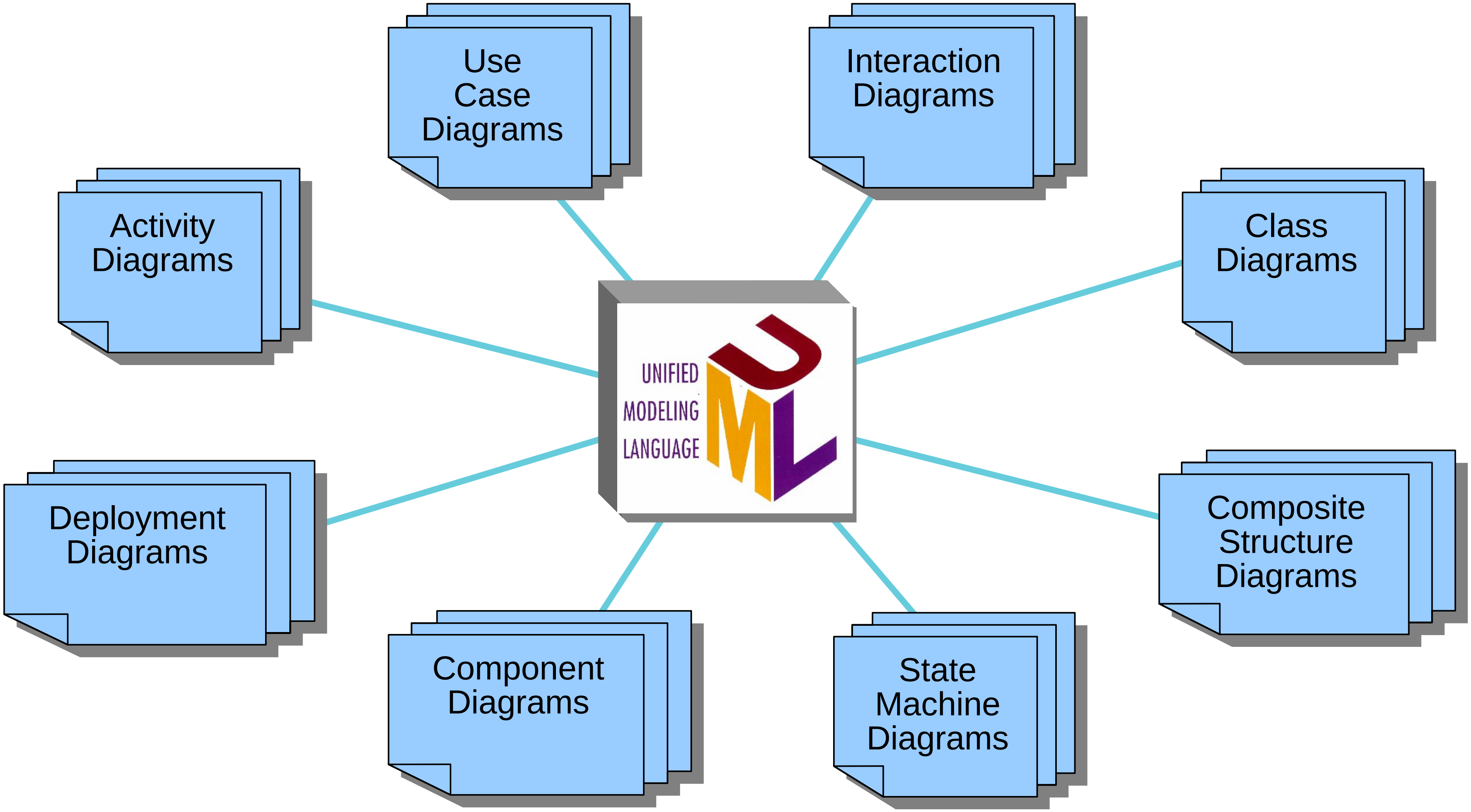
The FTF Recommendation and Report for this specification will be published on April 30, 2004. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

- Set of modeling languages for specific purposes

The Languages of MDA

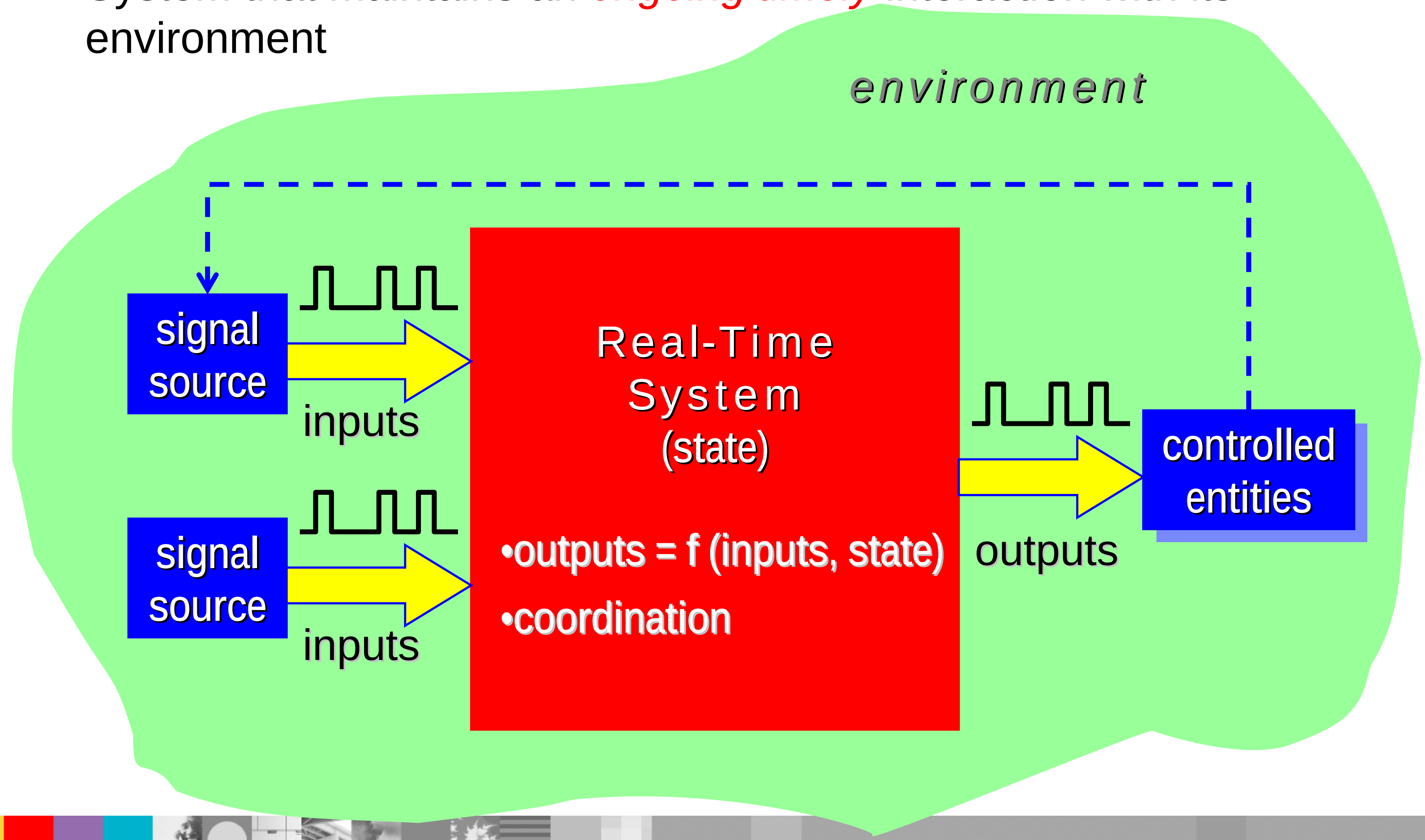

UML
"bootstrap"
MetaObject Facility (MOF)
MOF "core"
A modeling language for defining modeling languages
General Standard UML
General-purpose modeling language
Common Warehouse Metamodel (CWM)
For exchanging information about business and data
etc.
Real-Time profile
EAI profile
Software process profile
etc.

UML 2.0 Diagrams

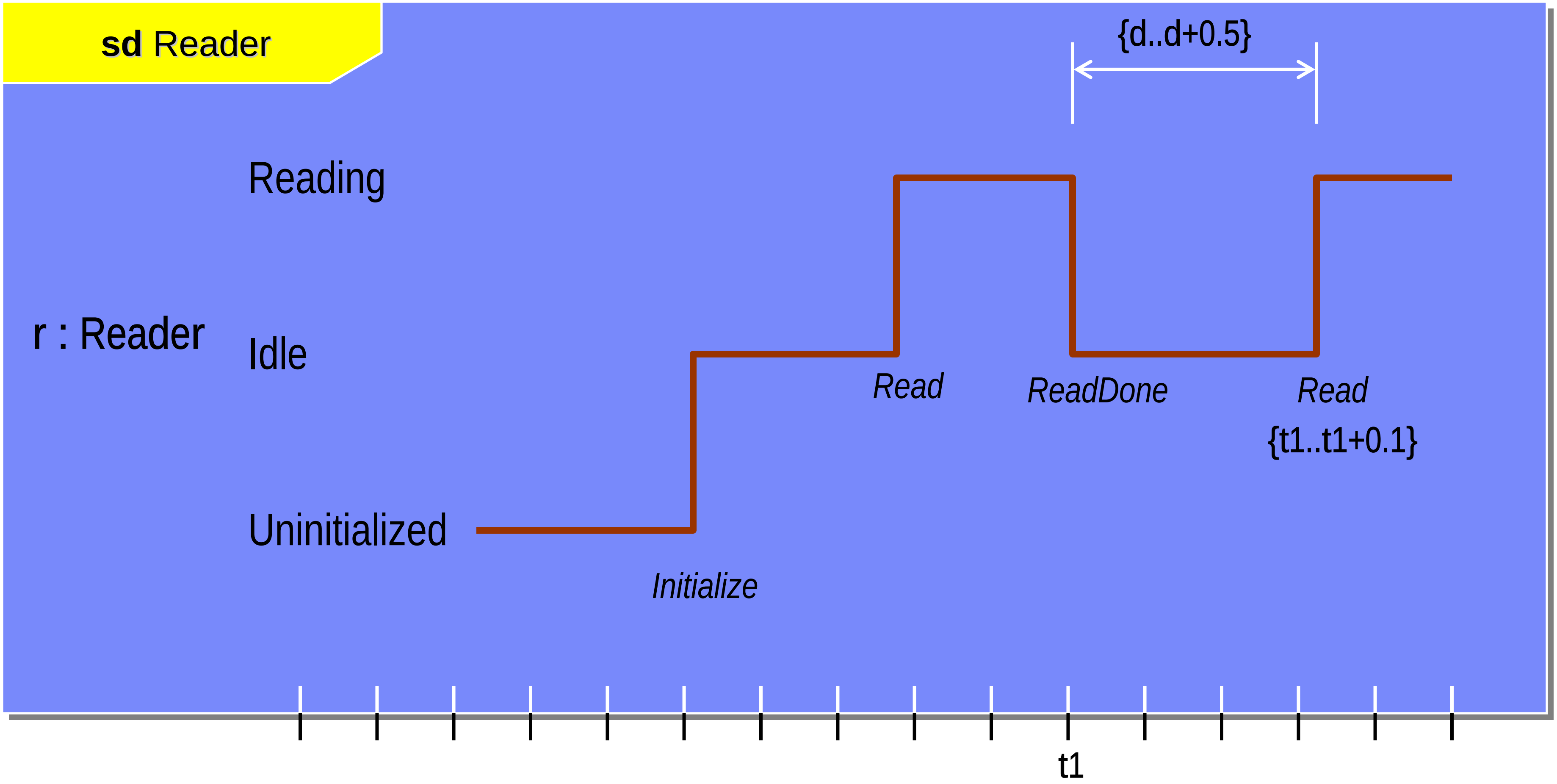


Real-Time System

- System that maintains an *ongoing timely* interaction with its environment



Timing Diagram Example



Yes, But What About...

- Modeling real-time specific phenomena?
  - time and timing mechanisms
  - resources (processors, networks, semaphores, etc.)
- Exploiting current real-time system theory?
  - schedulability analysis (e.g., rate-monototnic theory)
  - performance analysis (queueing theory)

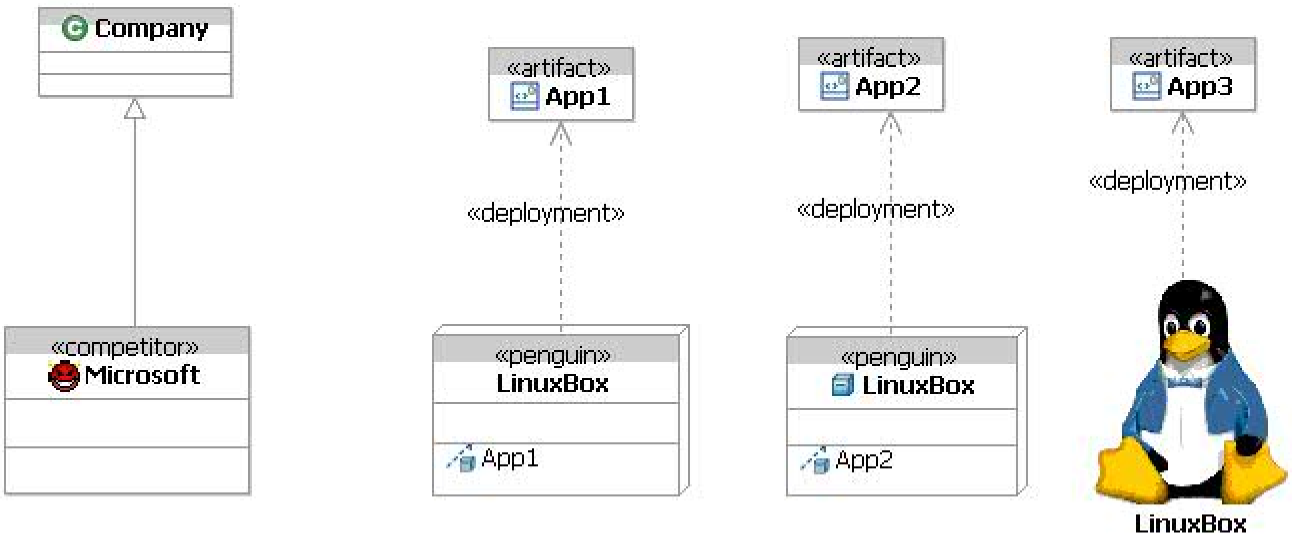
Extending the UML

- In order to model something effectively, the language that you use to model it must be rich and expressive enough to do so
- “Out of the box” UML is sufficient for modeling object-oriented software systems
- BUT… there are many more models that are useful in the software development process
- UML can easily be extended to add more semantics to cover other modeling situations
  - Database models
  - Business process
  - Web pages
  - On and on….

Extension Mechanisms

- Stereotypes
- Tag definitions and tagged values
- Constraints

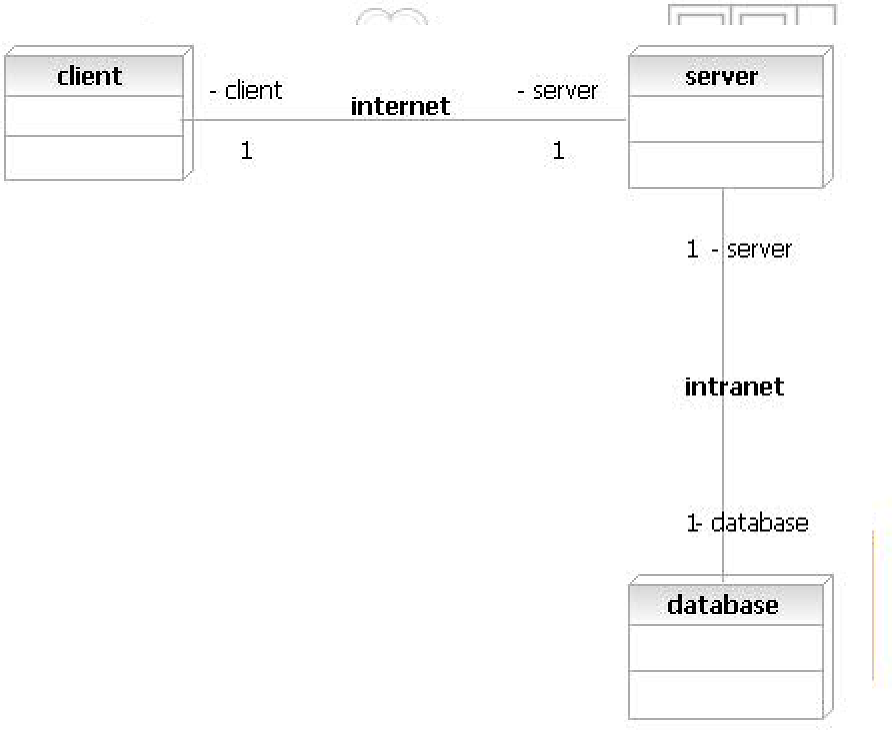
Stereotypes



Stereotype: A more refined semantic interpretation for a model element

Same semantic meaning – can be presented in several ways to make the visual easier to understand

Deployment Diagram



Tag Definitions and Tagged Values

- A **tag definition** the ability to associate extra information with a modeling element
  - Defines the name and the type of information
  - A special attribute that all members of the set of <<tag>> have
  - E.g., <<CPU>> might have <performance> tag
- A **tagged value** is the actual instance of a tag definition with a value that is associated to the modeling element
  - E.g., MyLaptop <<CPU>> has <performance=low>

Constraints

- A **constraint** is a rule that is applied to a modeling element
  - Represented by curly braces in UML
- Used to evaluate if a modeling element is “well-formed”
- Example
  - The name of a column cannot exceed the maximum length for column names for the associated database
- The language of constraints is Object Constraint Language (OCL)

Profiles

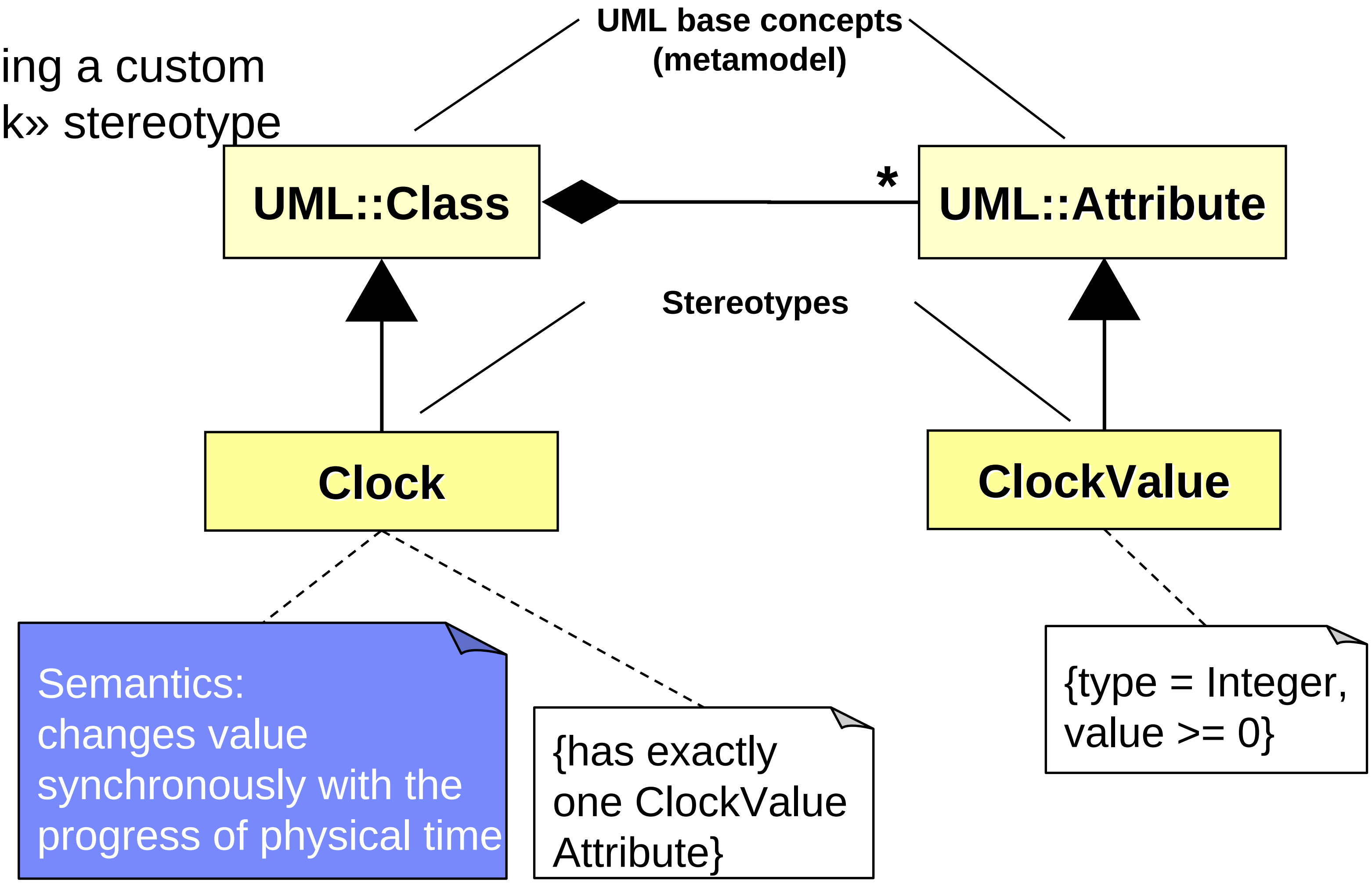
- A **profile** is a collection of steretoypes, tag definitions and constraints that work together to define new semantics for a model
- Example
  - Data modeling profile
  - Business modeling profile
  - SOA profile
  - Real-time profile

The Profile-Based Approach to DSLs

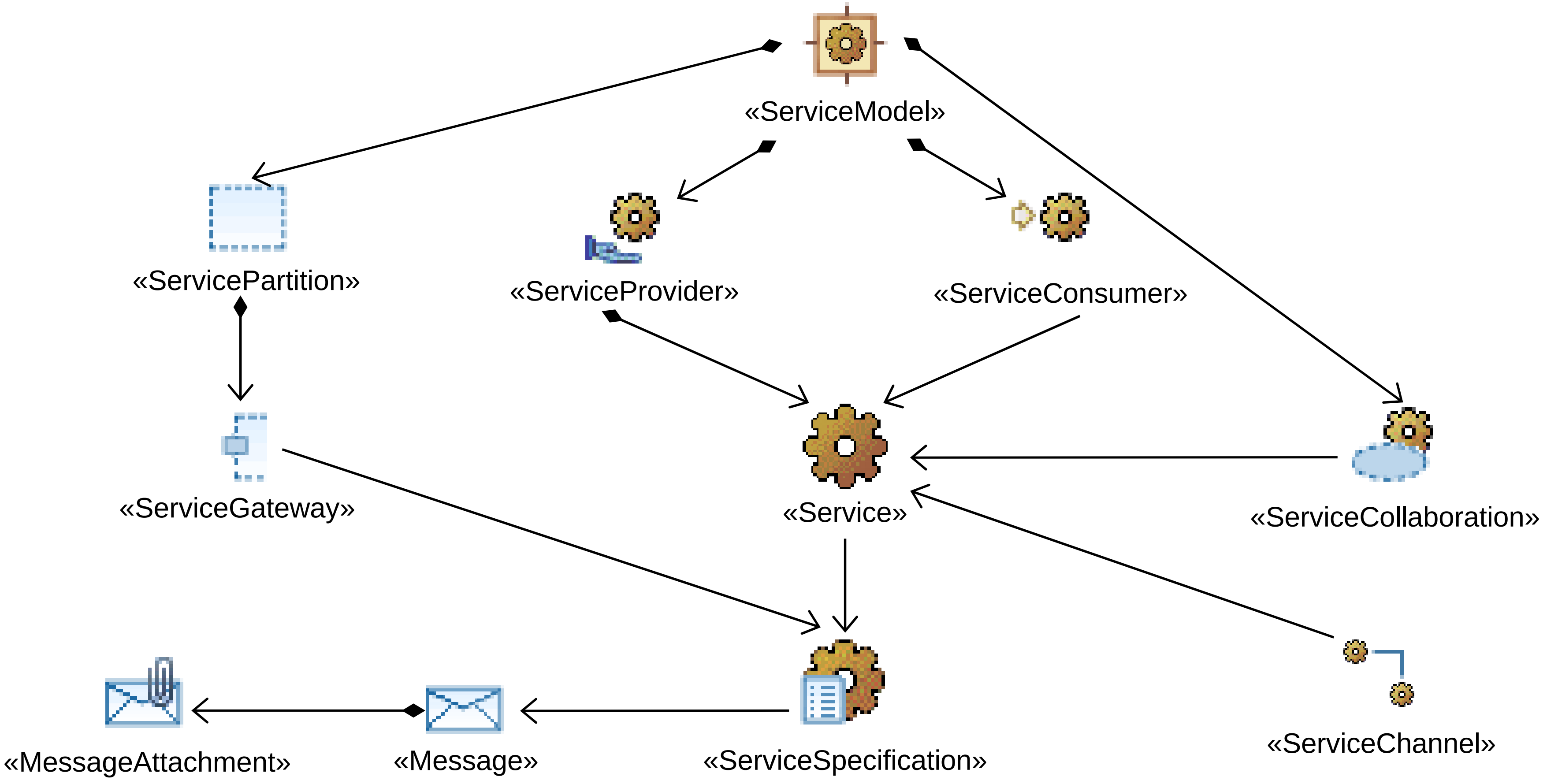
- Profile = a compatible specialization of an existing modeling language by
  - Adding constraints, characteristics, new semantics to existing language constructs
  - Hiding unused language constructs
- Advantages:
  - Supported by the same tools that support the base language
  - Reuse of base language knowledge, experience, artifacts
- Example: ITU-T standard language SDL (Z.100)
  - Modeling language used in telecom applications
  - Now defined as a UML profile (Z.109)

UML Profile Creation

- Defining a custom «clock» stereotype


UML base concepts (metamodel)
UML::Class
*
UML::Attribute
Stereotypes
Clock
ClockValue
Semantics: changes value synchronously with the progress of physical time
{has exactly one ClockValue Attribute}
{type = Integer, value >= 0}

Example: UML Profile for SOA



The Real-Time A&D Group in OMG

- An OMG working group
  - mission: to investigate and issue requests (RFPs) for standard ways and means to apply UML to real-time problems
- Three principal areas of investigation:
  - Time-related modeling issues
  - General quality of service/fault tolerance modeling issues
  - Architectural modeling issues
- Status:
  - UML Profile for Schedulability, Performance, and Time Specification
    - V1.1, January 2005
    - Related profile UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms

Quantitative Methods for RT Systems

- Once we have included QoS information in our models, we can use *quantitative methods* to:
  - predict system characteristics (detect problems early)
  - analyze existing system
  - synthesize elements of the model
- Current quantitative analysis methods:
  - Schedulability analysis

    will the system meet all of its deadlines?
  - Performance analysis based on queueing theory

    what kind of response will the system have under load?

Quality of Service Concepts

- An abstract, technology-independent representation of the engineering model can be specified using the general concept of *Quality of Service (QoS):*

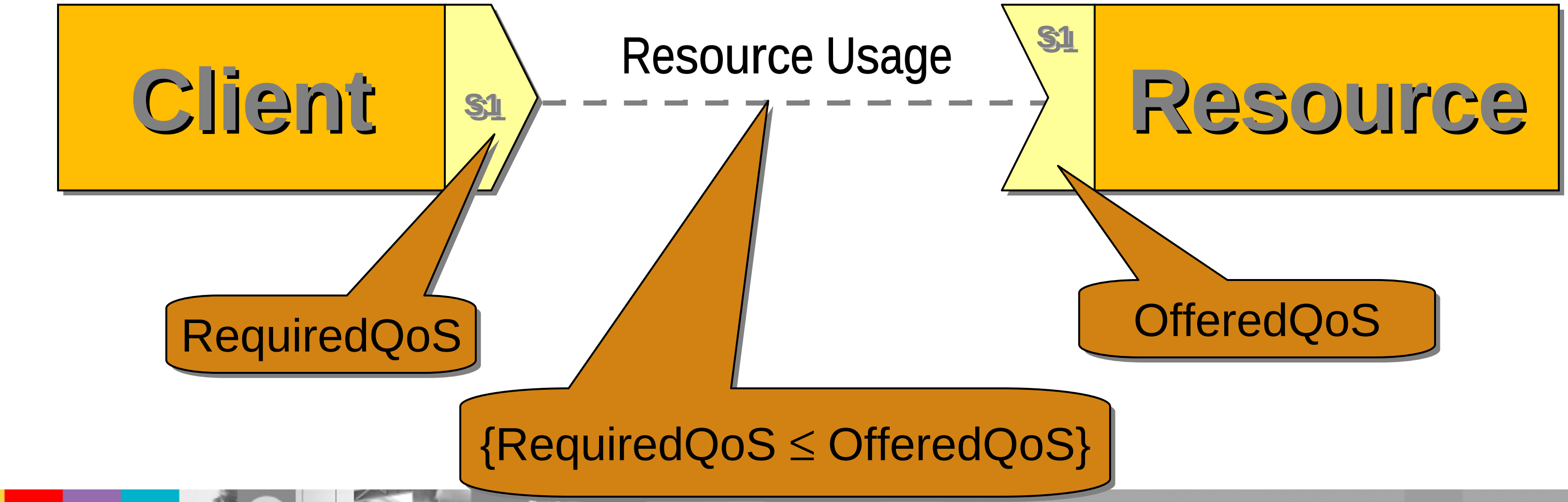
  a specification (usually quantitative) of how a particular service is (to be) performed

  - e.g. throughput, capacity, response time

- The specification of a model element can include:
  - *offered QoS:* the QoS that it provides to its clients
  - *required QoS:* the QoS it requires from other components to support its QoS obligations

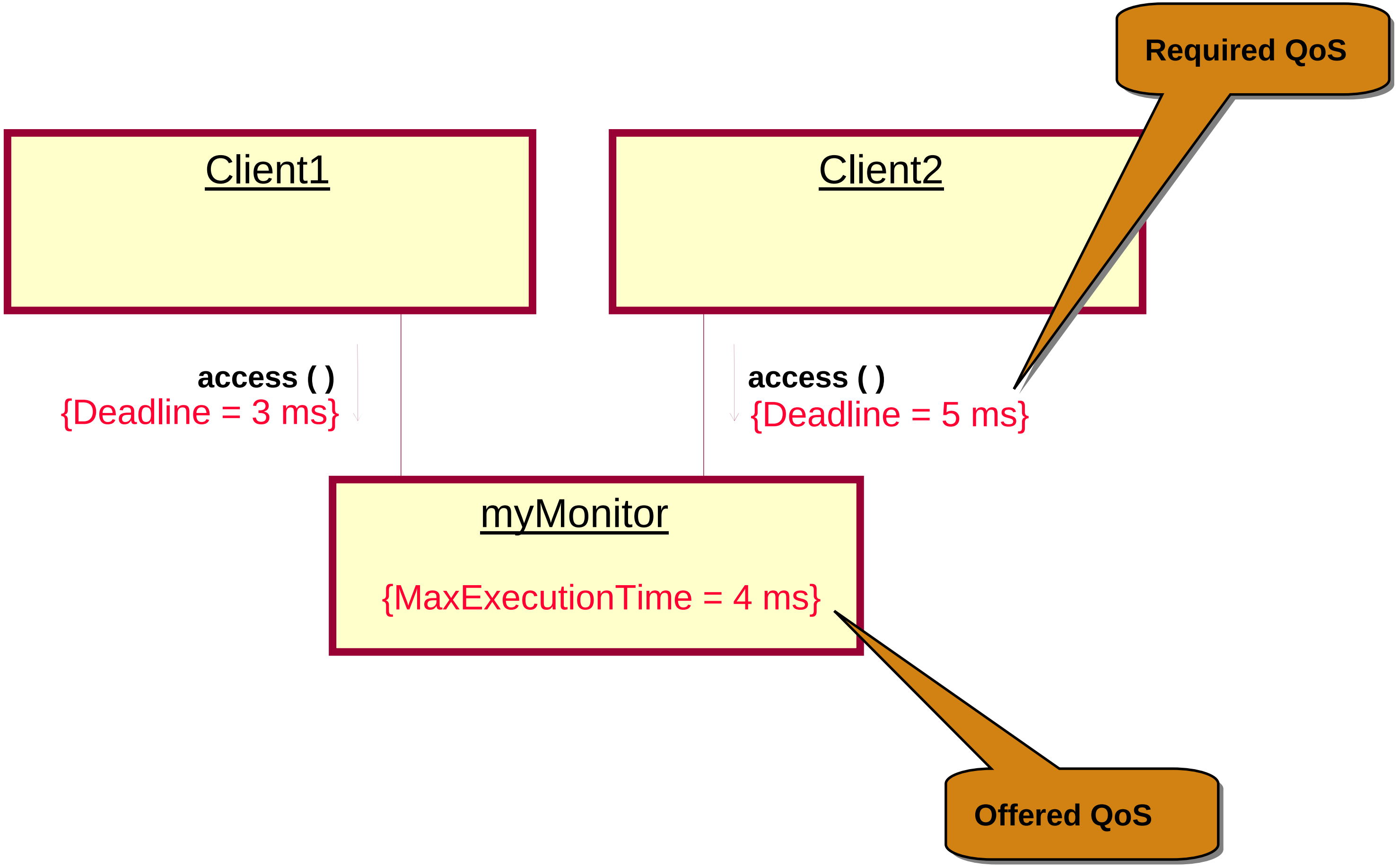
Resources and Quality of Service

- *Resource:* an element whose service capacity is limited, directly or indirectly, by the finite capacities of the underlying physical computing environment
- The services of a resource are characterized by one or more *quality of service (QoS)* attributes
  - capacity, reliability, availability, response time, etc.



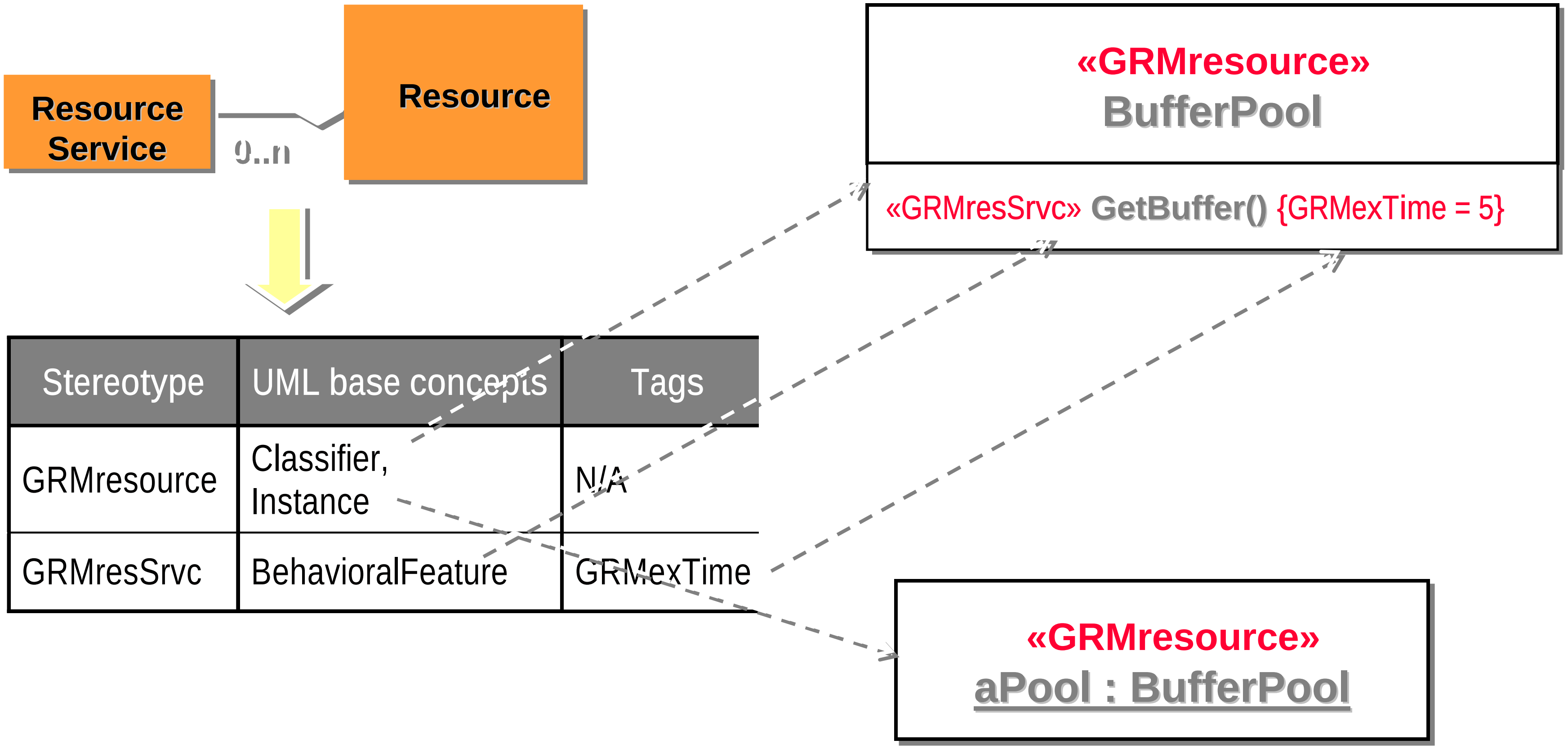
Simple Example

- Concurrent tasks accessing a monitor with known response time characteristics



Mapping to UML Extensions

- Elements of the general resource model are represented as stereotypes (with tags) of base UML concepts:



Resource Service

0..n

Resource

Stereotype	UML base concepts	Tags
GRMresource	Classifier, Instance	N/A
GRMresSrvc	BehavioralFeature	GRMexTime

«GRMresource»
BufferPool

«GRMresSrvc» GetBuffer() {GRMexTime = 5}

«GRMresource»
aPool : BufferPool

Standard Stereotypes

- To allow an analysis tool to extract the necessary QoS information, we define a set of standard stereotypes and

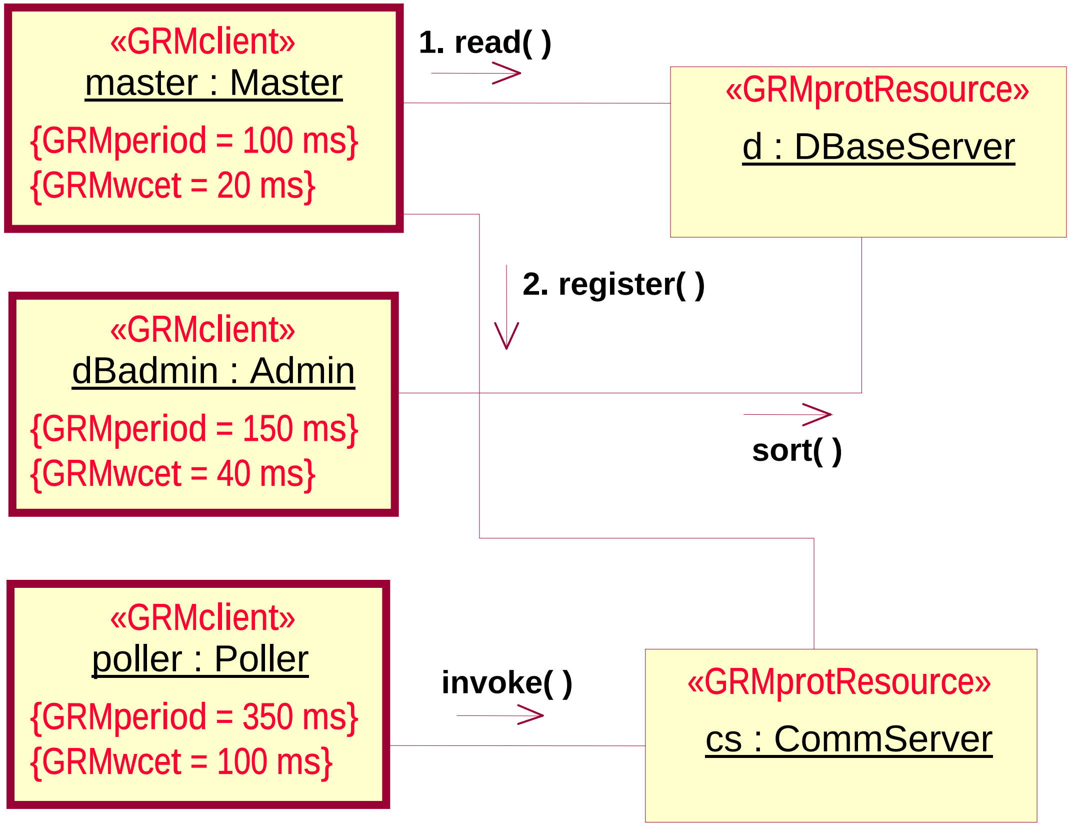
Stereotype	UML base concepts	Tags
GRMclient	Classifier, Instance	GRMperiod, GRMwcet
GRMprotResource	Classifier, Instance	N/A
GRMresService	BehavioralFeature	GRMwcet

Tag	Tag Type
GRMperiod	RTtimeString
GRMwcet	RTtimeString

* The stereotypes and tags have been simplified for this presentation

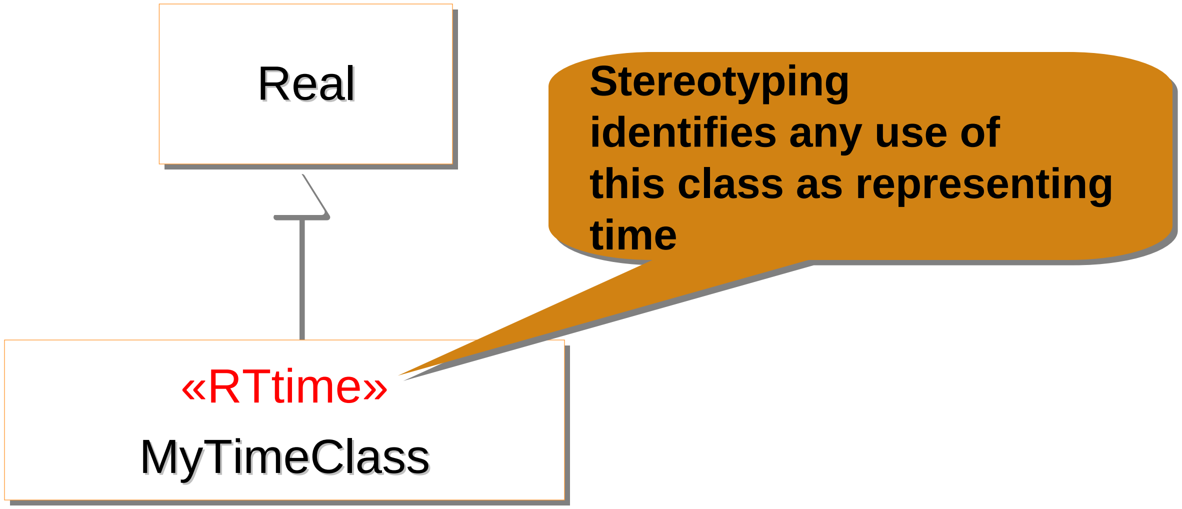
Example: QoS Annotations

- Using the standard stereotypes...



RT Profile: Modeling Time

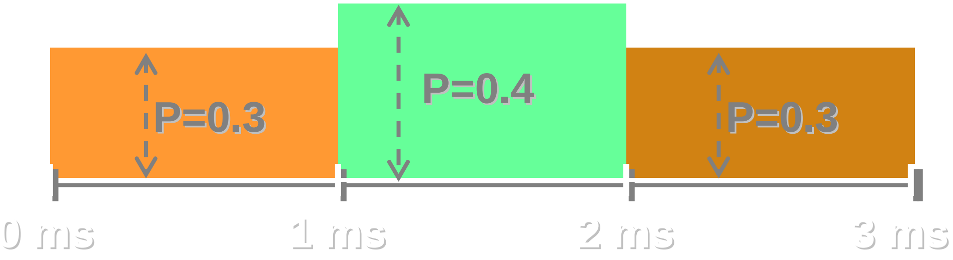
- «RTtime»: a stereotype of Classifier (and Instance)
  - supports both continuous and discrete time representations
  - e.g. as a kind of real value



Stereotyping identifies any use of this class as representing time

Specifying Time Values

- Time values can be represented by a special stereotype of Value («RTtimeString») in different formats; e.g.
  - “12:04” (time of day)
  - “5.3 ms” (time interval)
  - “2000/10/27” (date)
  - “Wed” (day of week)
  - .$param ms” (parameterized value)
  - “poisson 5.4 sec” (time value with a Poisson distribution)
  - “histogram 0:0.3 1:0.4 2:0.3 3 ms”



Notation: Timing Marks and Constraints

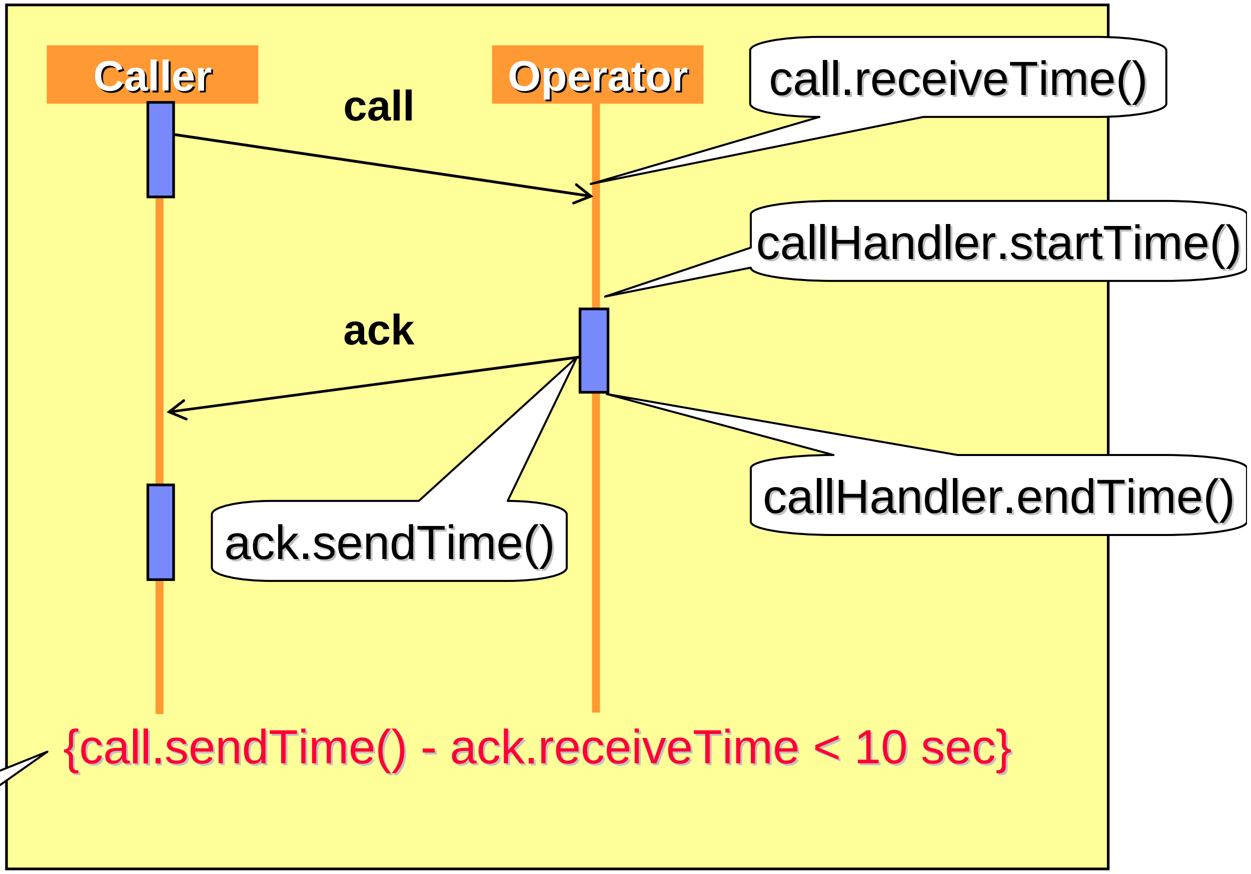
- A *timing mark* identifies the time of an event occurrence

- On messages:

  sendTime()
  receiveTime()

- On action blocks (new):

  startTime()
  endTime()



Caller

Operator

call

ack

call.receiveTime()

callHandler.startTime()

callHandler.endTime()

ack.sendTime()

{call.sendTime() - ack.receiveTime < 10 sec}

Timing constraint

Summary: The Real Time UML Profile

- The RT UML Profile defines a set of extensions for directly expressing real-time domain concepts in UML:
  - resources
  - concurrency mechanisms
  - time and timing mechanisms
- Furthermore, it allows the specification of quantitative aspects in the same models such that the models can be analyzed
  - predictive models that can be used to validate (risky) design approaches before major investments are made

Thank
You