technische universität
dortmund

fakultät für informatik
informatik 12

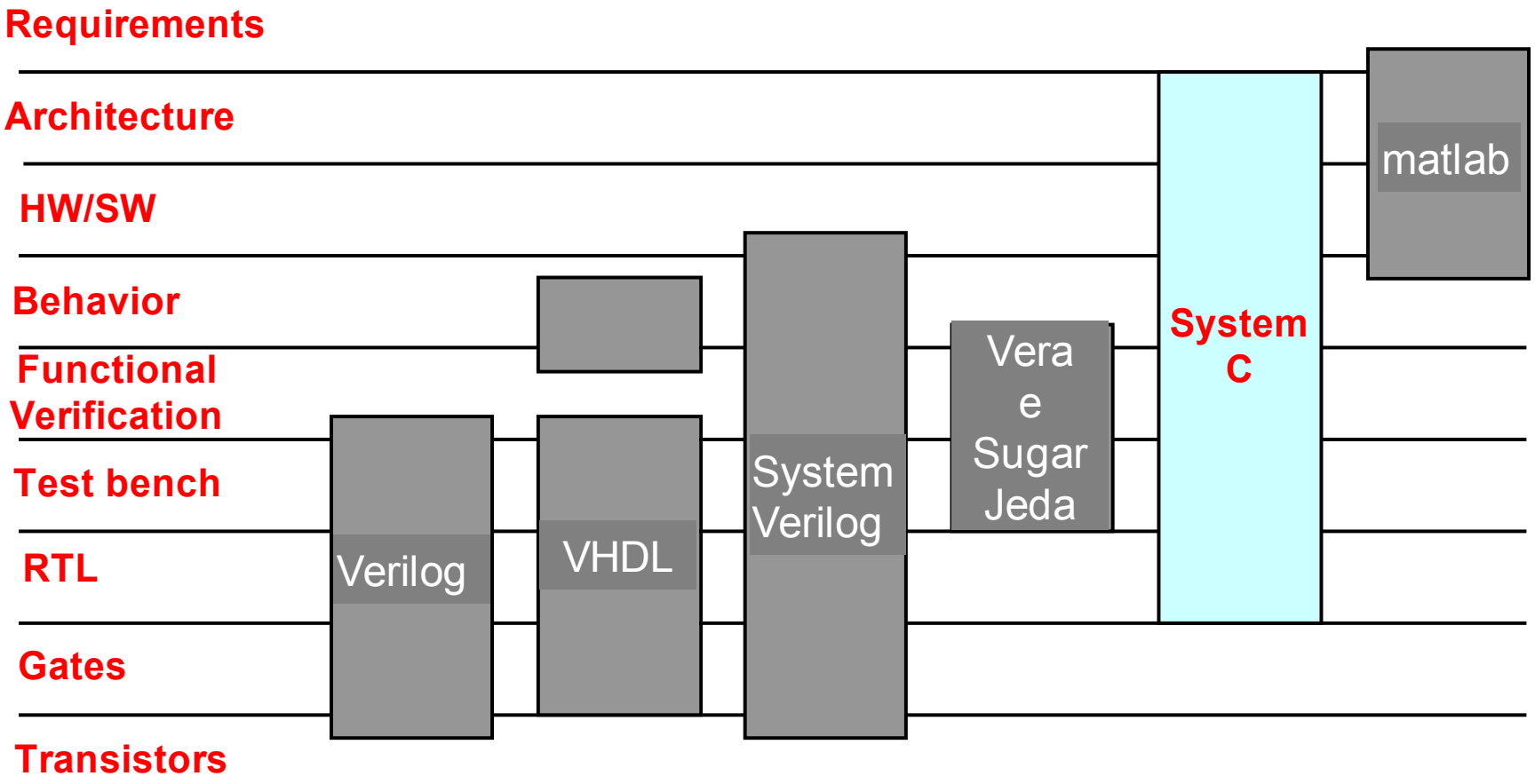
Languages and Models of Computation (Revisted)

Peter Marwedel
TU Dortmund
Informatik 12
Germany

Other languages

- **Estelle**: Designed to describe communication protocols; scope similar to SDL; unification of both failed.
- **LOTOS, Z**: Algebraic specification languages
- **Silage**: functional language for digital signal processing.
- **Rosetta**: Efforts on new system design language
- **Esterel:** reactive language; synchronous;
  all reactions are assumed to be in 0 time;
  communication based on ("instantaneous") broadcast;
  //www.esterel-technologies.com
- **IEC 60848, STEP 7**:
  Process control languages using graphical elements

Levels covered by the different languages



Language Comparison

Language	Behavioral Hierarchy	Structural Hierarchy	Programming Language Elements	Exceptions Supported	Dynamic Process Creation
StateCharts	+	-	-	+	-
VHDL	+	+	+	-	-
SpecCharts	+	-	+	+	-
SDL	+-	+-	+-	-	+
Petri nets	-	-	-	-	+
Java	+	-	+	+	+
SpecC	+	+	+	+	+
SystemC	+	+	+	- (2.0)	- (2.0)
ADA	+	-	+	+	+

Comparison of languages demonstrated

<table>
<tr><td>Communication/
local computations</td><td>Shared
memory</td><td colspan="2">Message passing
Synchronous | Asynchronous</td></tr>
<tr><td>Communicating finite state machines</td><td>StateCharts</td><td></td><td>SDL</td></tr>
<tr><td>Data flow model</td><td>Not useful</td><td></td><td>Kahn process networks</td></tr>
<tr><td>Von Neumann model</td><td>C, C++,
Java</td><td colspan="2">C, C++, Java with libraries
CSP, ADA |</td></tr>
<tr><td>Discrete event (DE) model</td><td>VHDL,
Verilog,
SystemC</td><td colspan="2">Only experimental systems, e.g. distributed DE in Ptolemy</td></tr>
</table>

Observations

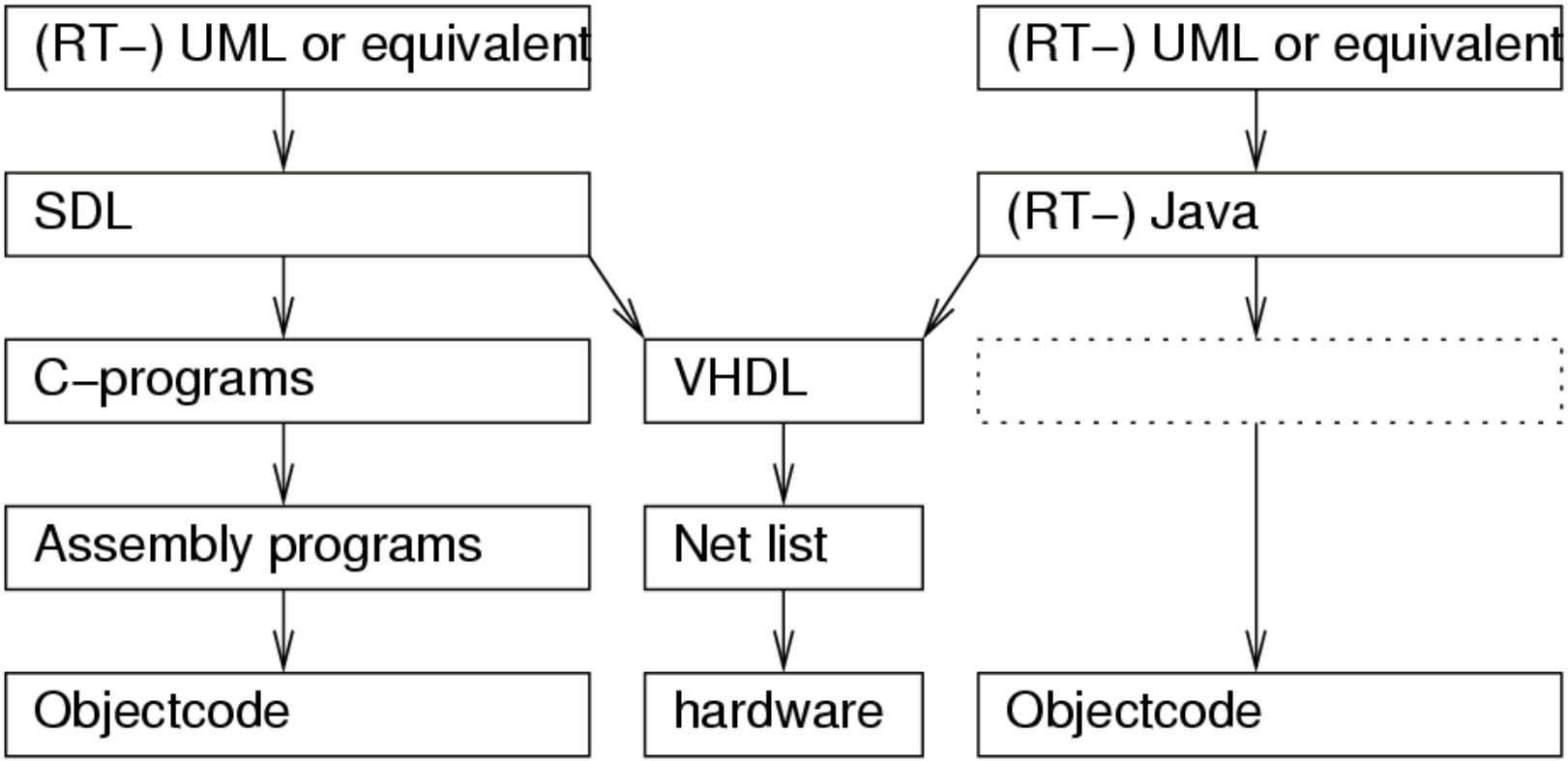
- Useful to consider model of computation matching with the class of applications considered
- Nevertheless, feasible to simulate one model in another model (e.g. most models are simulated with imperative computing ☞ Ptolemy)
- Commercial products available:
  - DSpace (Paderborn): Synthesis of control software for the power train in cars
  - Esterel Technologies (Toulouse): Synthesis from synchronous language SCADE (used by Airbus)

http://www.dspace.de/ww/de/gmb/home/company/dspace_pressroom/press/pr08040 03.cfm

http://www.esterel-technologies.com/technology/success-stories/airbus.html

How to cope with language problems in practice?

Mixed approaches:



Models of computation in Ptolemy

1. Finite state machines
2. Communicating sequential processes
3. Discrete event model
4. Distributed discrete event model
5. Process networks, including Kahn process networks
6. Synchronous dataflow (SDF)
7. Continous time
8. Synchronous/reactive models

UML and
Model Driven Design

Peter Marwedel
TU Dortmund
Informatik 12
Germany

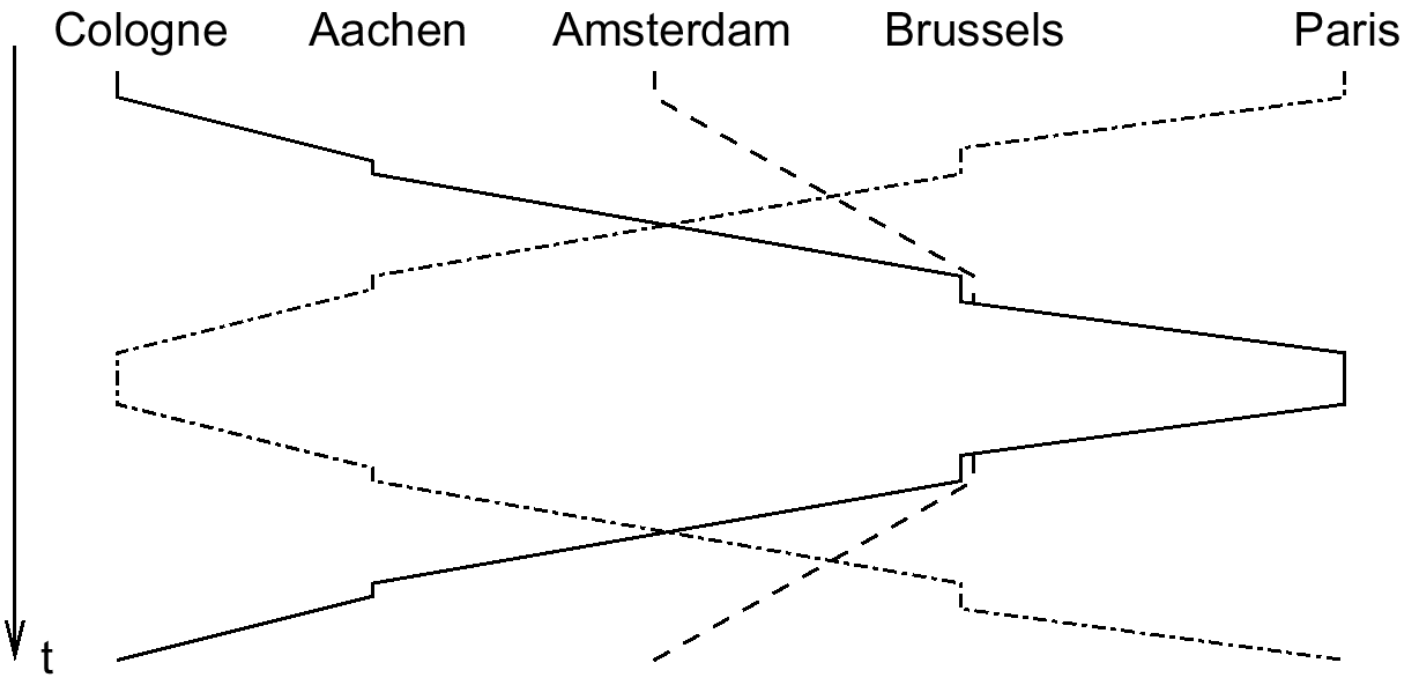
Higher levels of abstraction

- For many applications, it is desirable to start modeling at a level higher than the (Java-, ADA-, SDL-code level)
- Special modeling mechanisms have been designed for these cases
- In particular, the unified modeling language (UML)
  – building on concepts already presented - is designed to cover this requirement ☞ Elements

http://portal.uni-freiburg.de/hochschulsport_ss2007/nuf/Hochgebirge_Eis.JPG

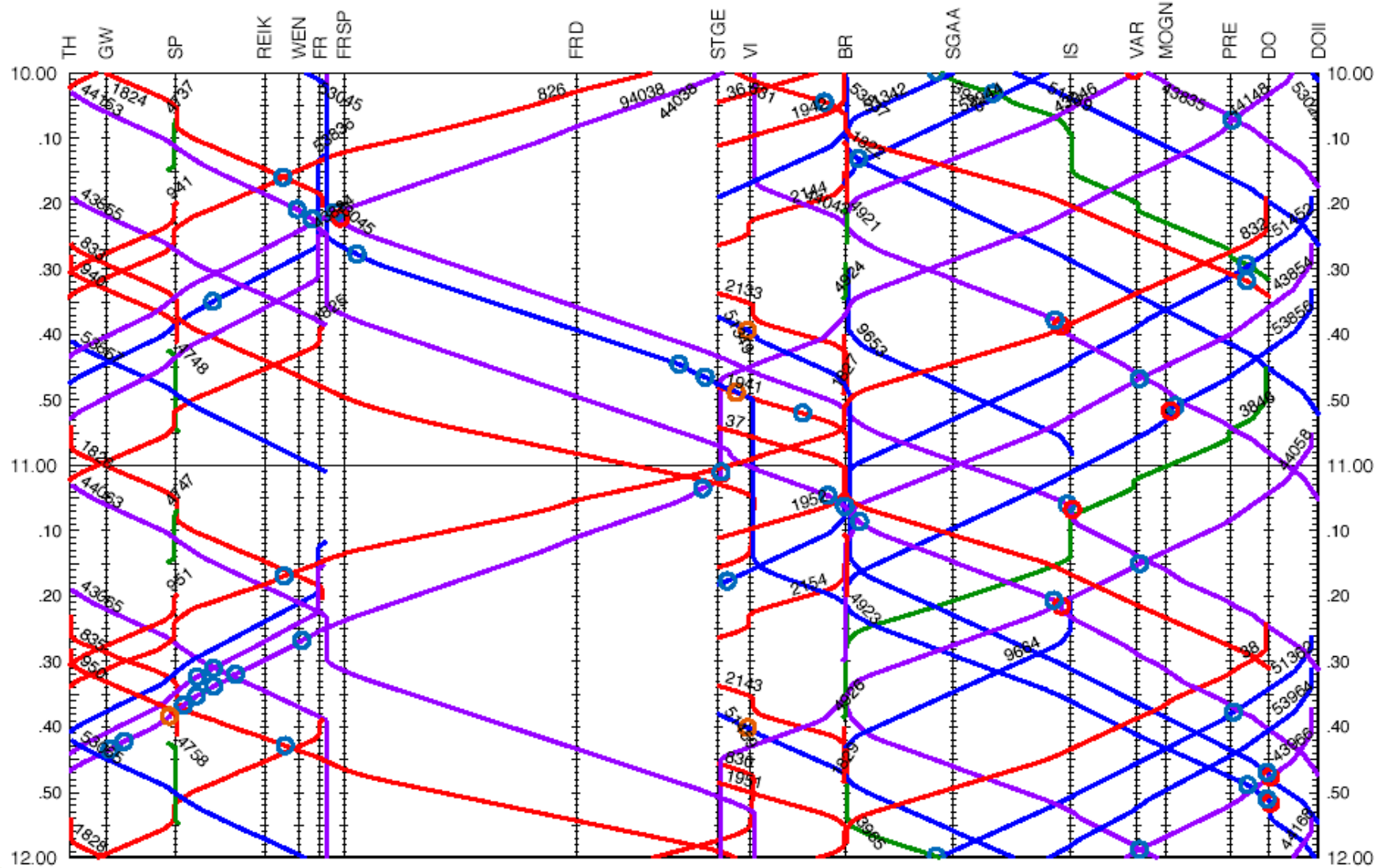
(Message) Sequence Charts (MSC)

Graphical means for representing schedules; time used vertically, geographical distribution horizontally.

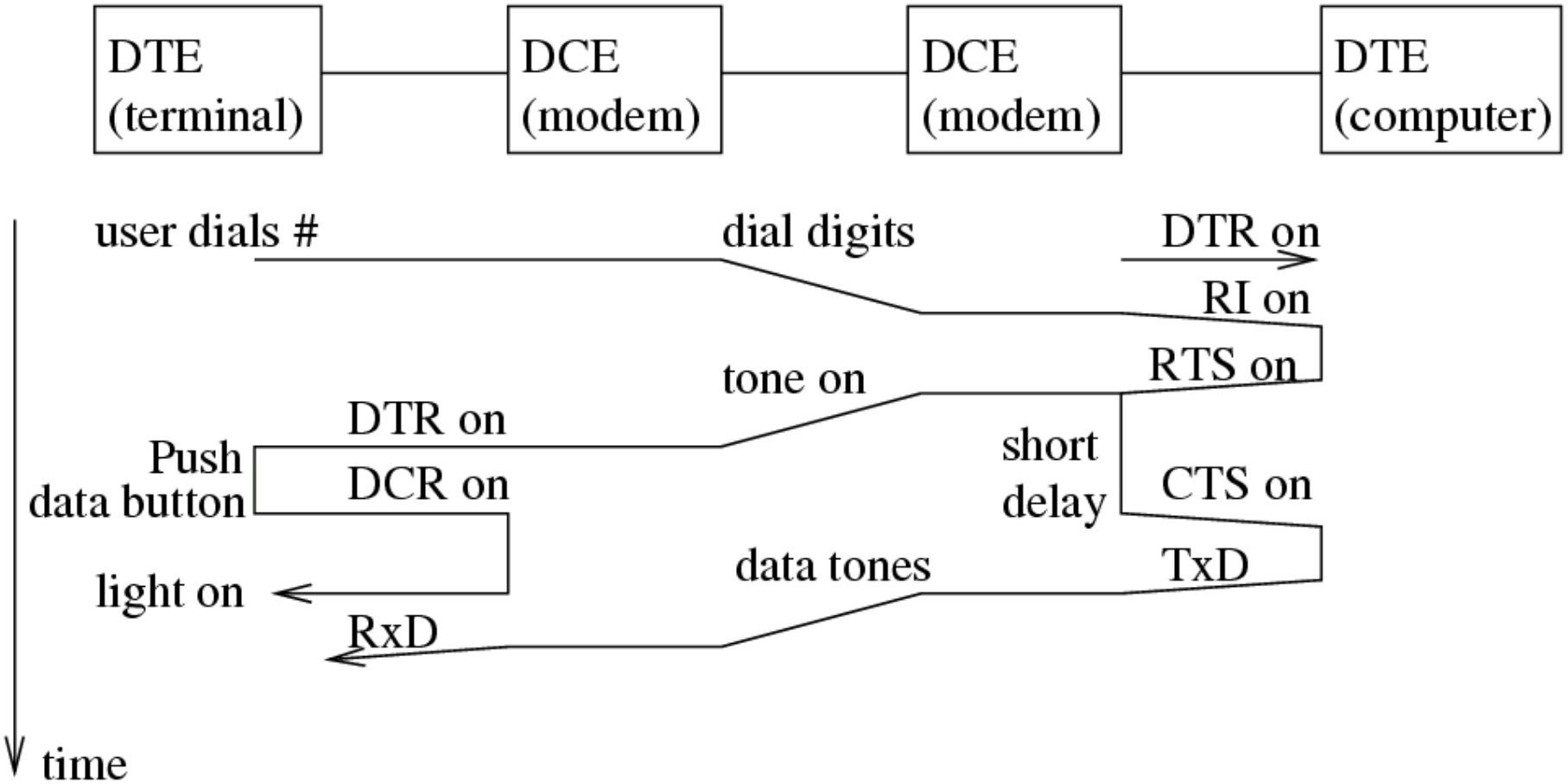


No distinction between accidental overlap and synchronization

Time/distance diagrams as a special case

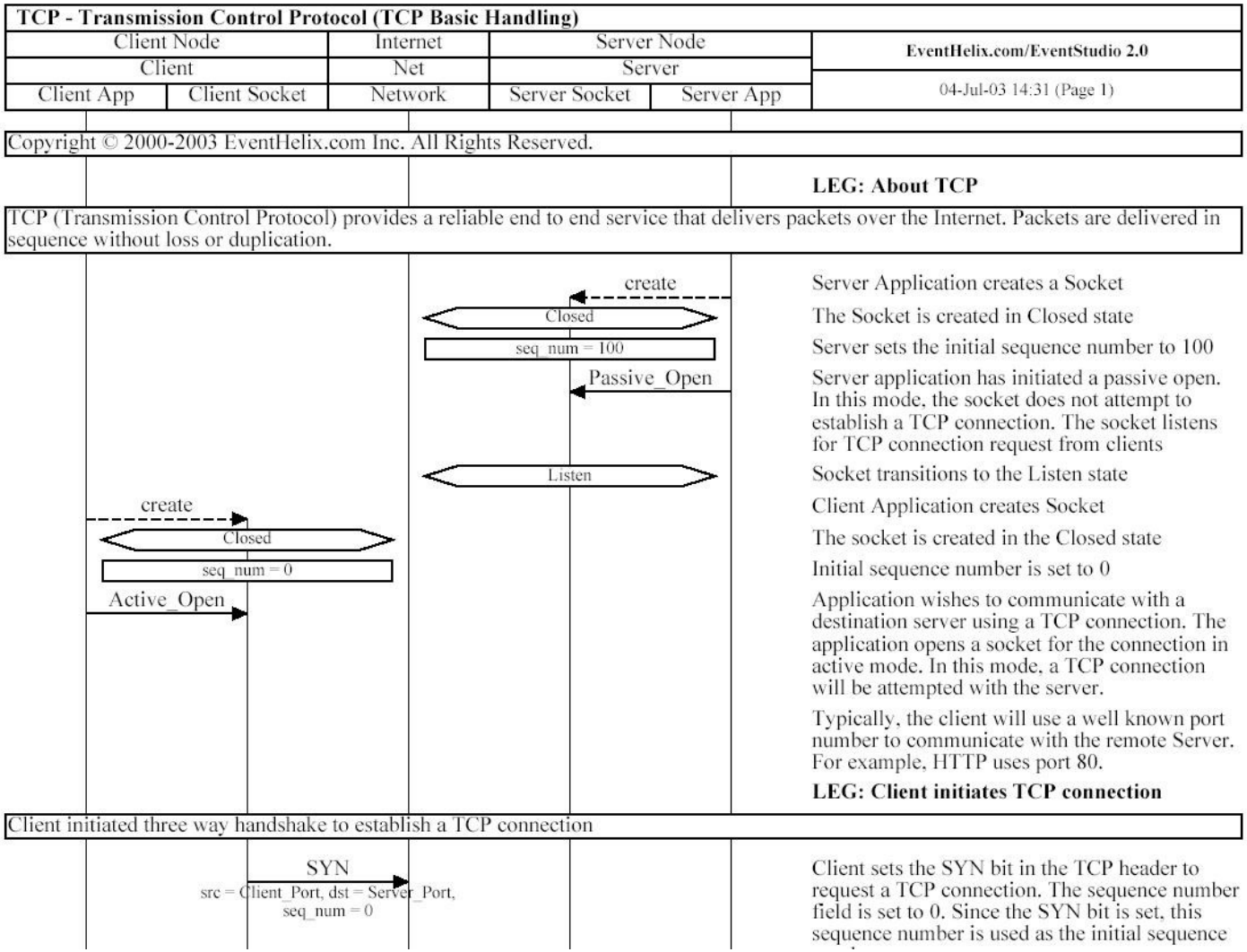


Example 1: establishing an (old-fashioned) modem connection



According to Stallings

Example 2: setting up a TCP connection



(Message) Sequence Charts

PROs:

- Appropriate for visualizing schedules,
- Proven method for representing schedules in transportation.
- Standard defined: *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC*), ITU-TS, Geneva, 1996.
- Semantics also defined: *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)—Annex* B: *Algebraic Semantics of Message Sequence Charts*, ITU-TS, Geneva.

CONS:

- describes just one case, no timing tolerances: "What *does an MSC specification mean: does it describe all behaviors of a system, or does it describe a set of sample behaviors of a system?"* *

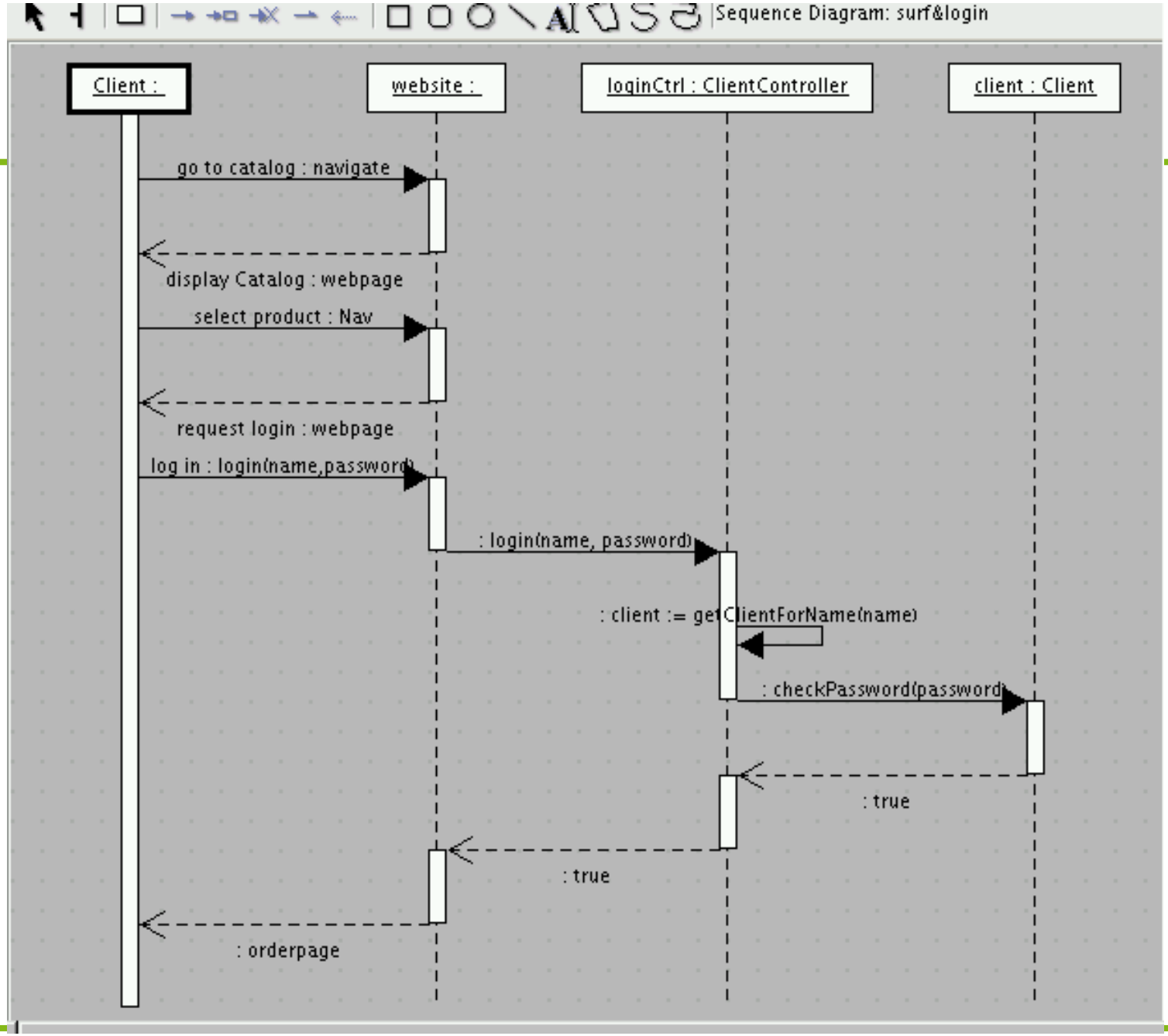
* H. Ben-Abdallah and S. Leue, "Timing constraints in message sequence chart specifications," in *Proc. 10th International Conference on Formal Description Techniques FORTE/PSTV'97*, Chapman and Hall, 1997.

Use in UML

Heavy usage of MSCs in UML (known as sequence diagram);

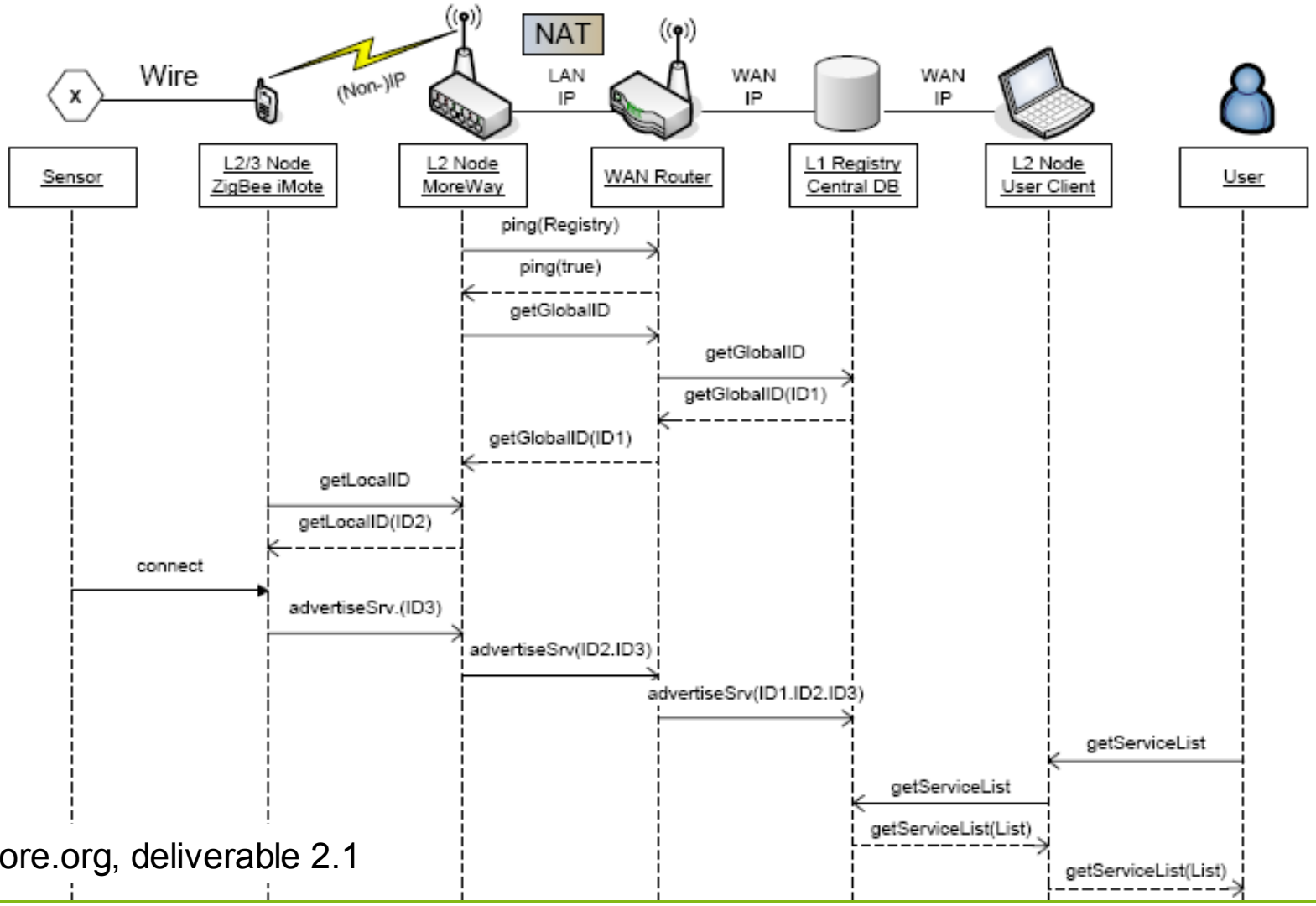
No precise timing.

Many kinds of additional elements



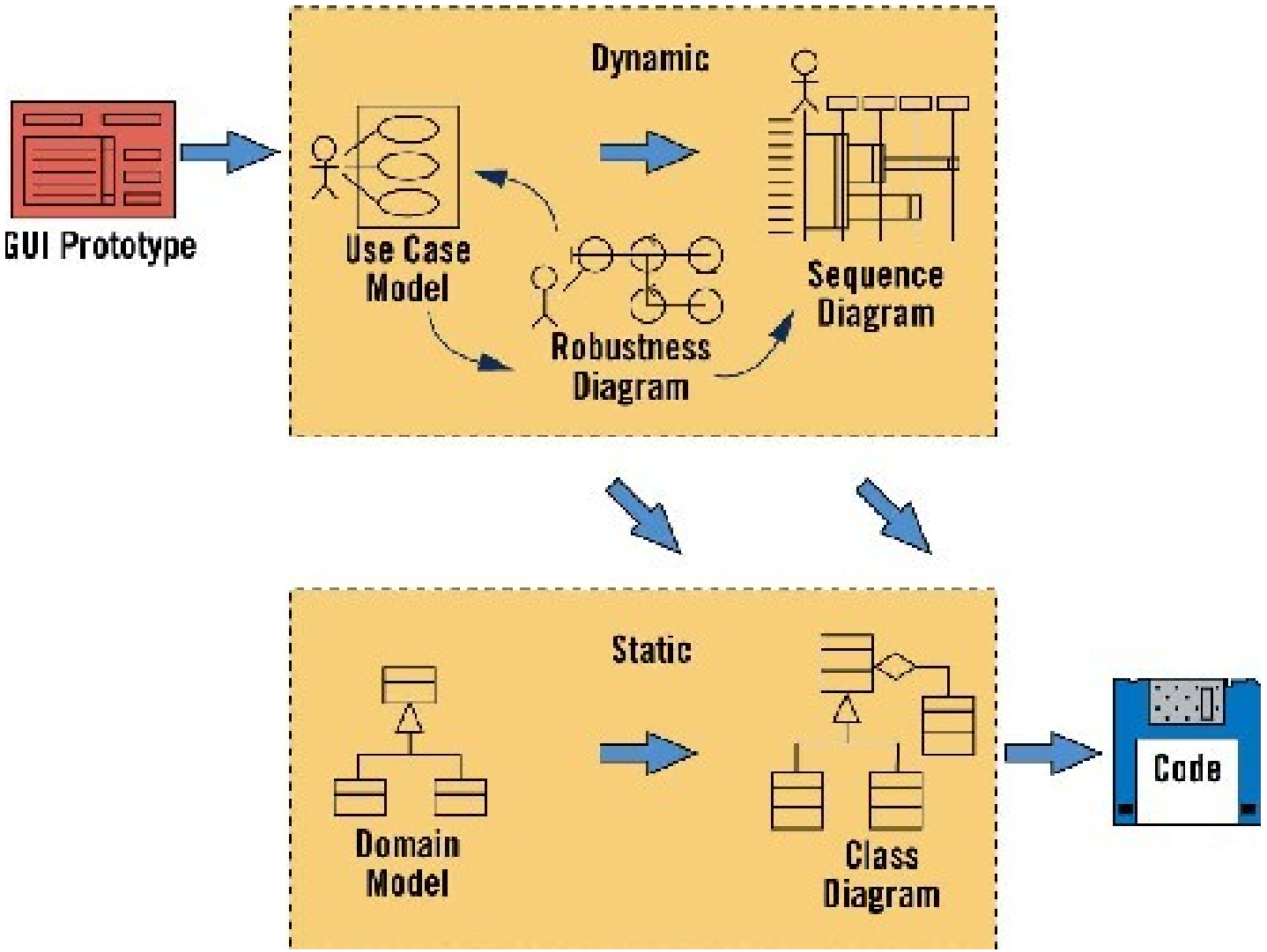
from: http://www.gentleware.com/products/documentation/PoseidonUsersGuide/x1462.html

Sequence Diagram (2)



www.ist-more.org, deliverable 2.1

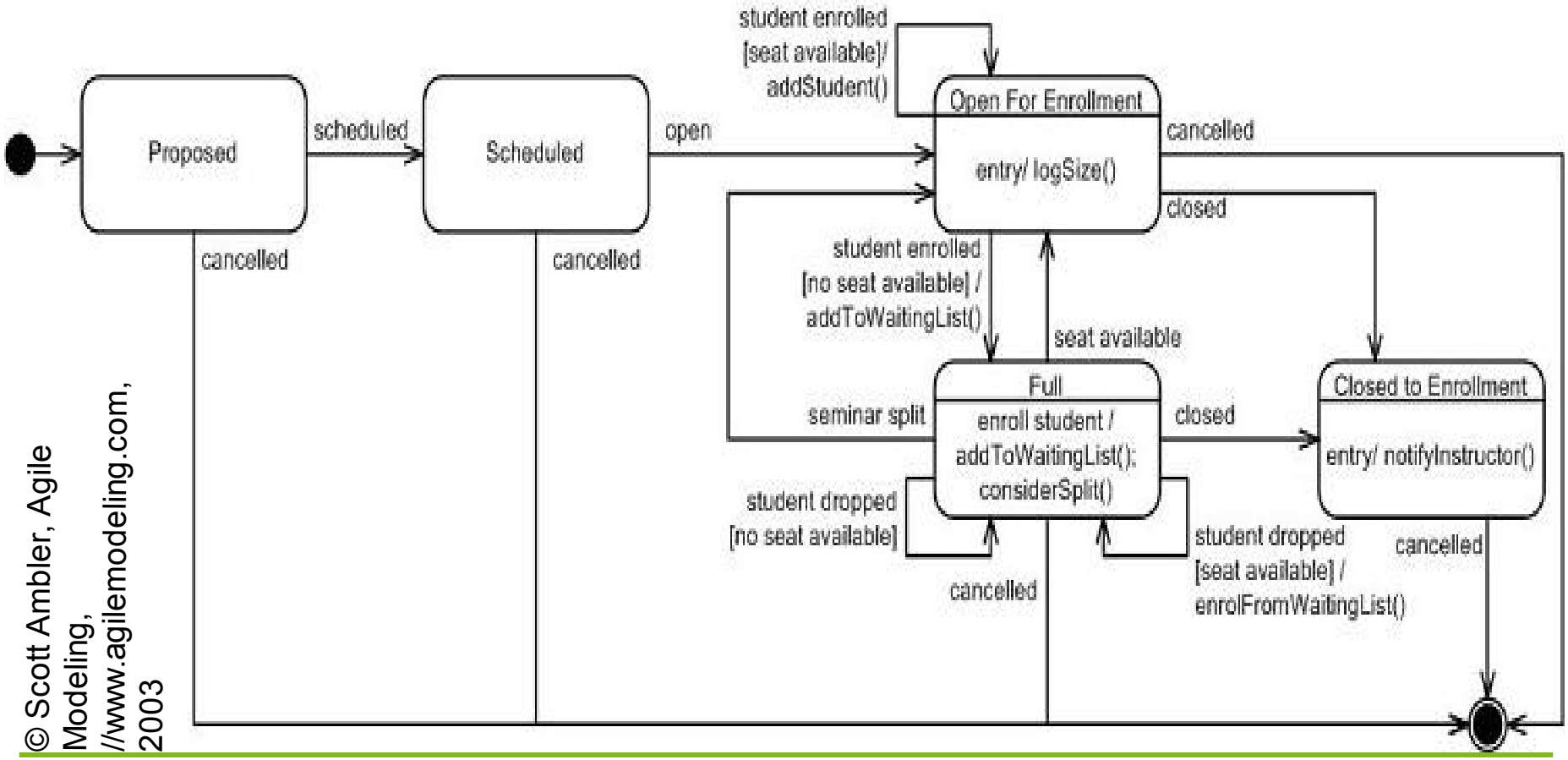
UML diagrams



From: www.sdmagazine.com/documents/s=815/sdm0012c/

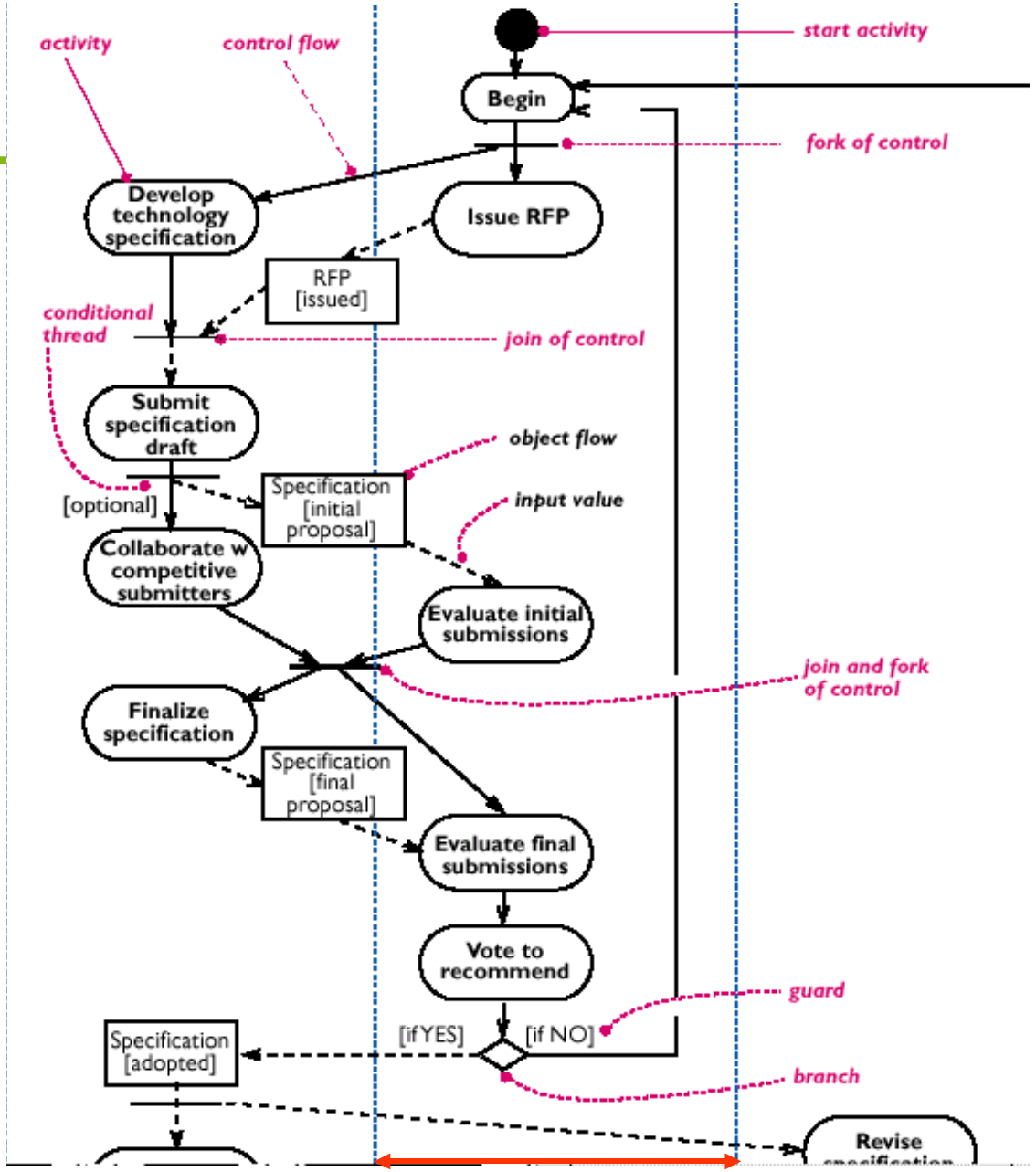
State machine diagrams (UML 2.x)
State diagrams (UML 1.x)

State machine diagrams/State diagrams:
UML includes variant of StateCharts



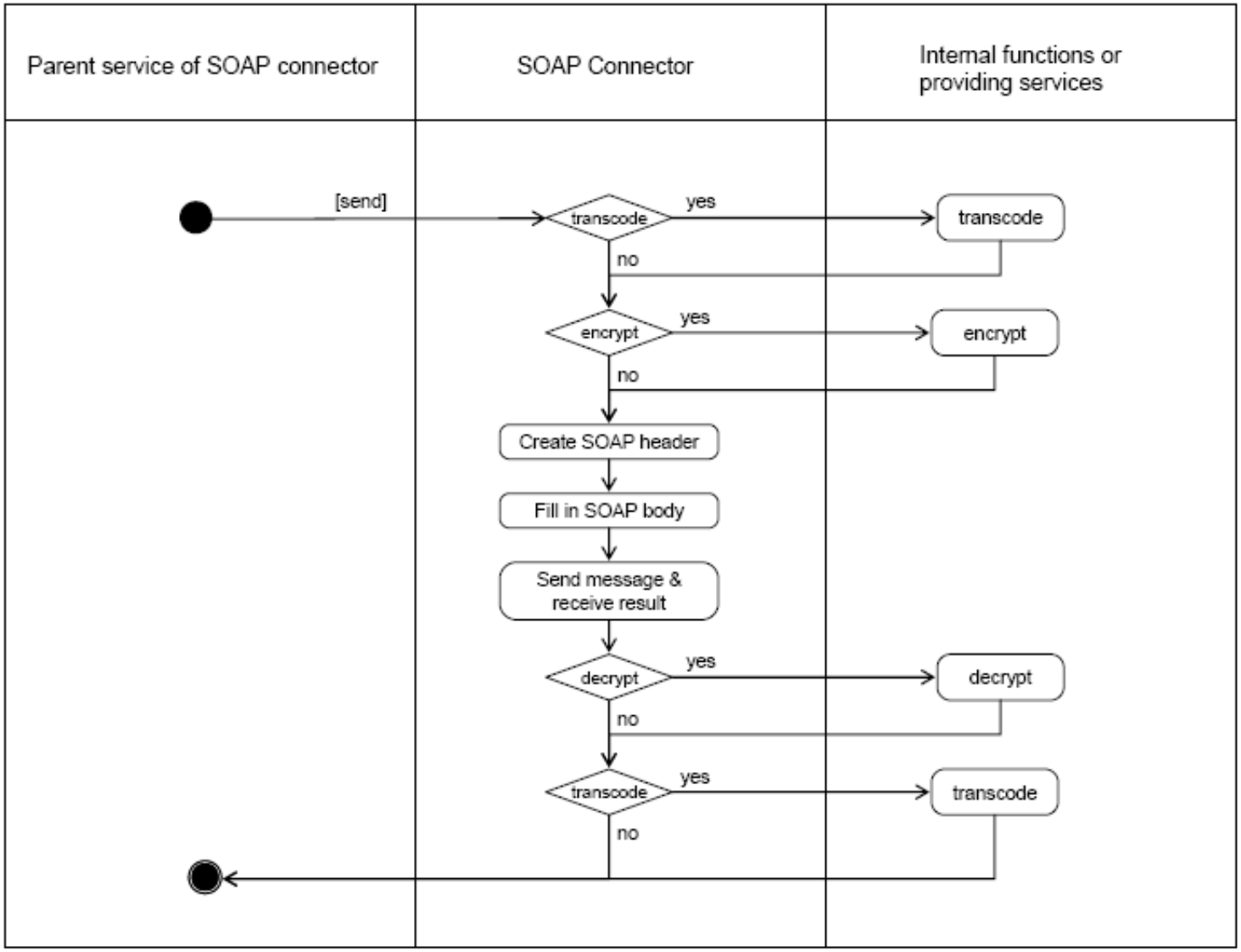
Activity diagram

Extended Petri nets.
Include decisions (like in flow charts).
Graphical notation similar to SDL.

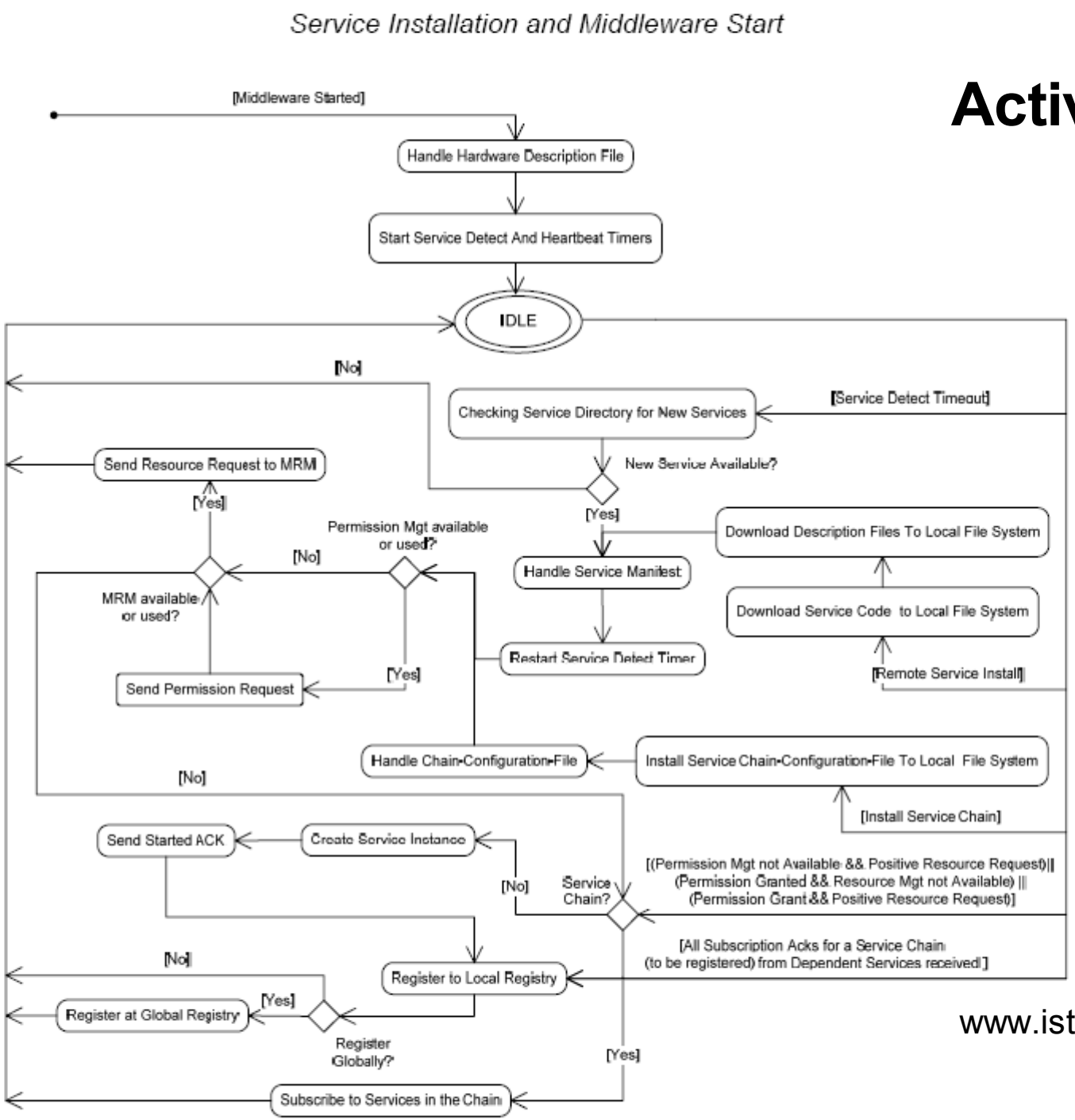


© Cris Kobryn: UML 2001: A Standardization Odyssey, CACM, October, 1999

Activity diagram (2)



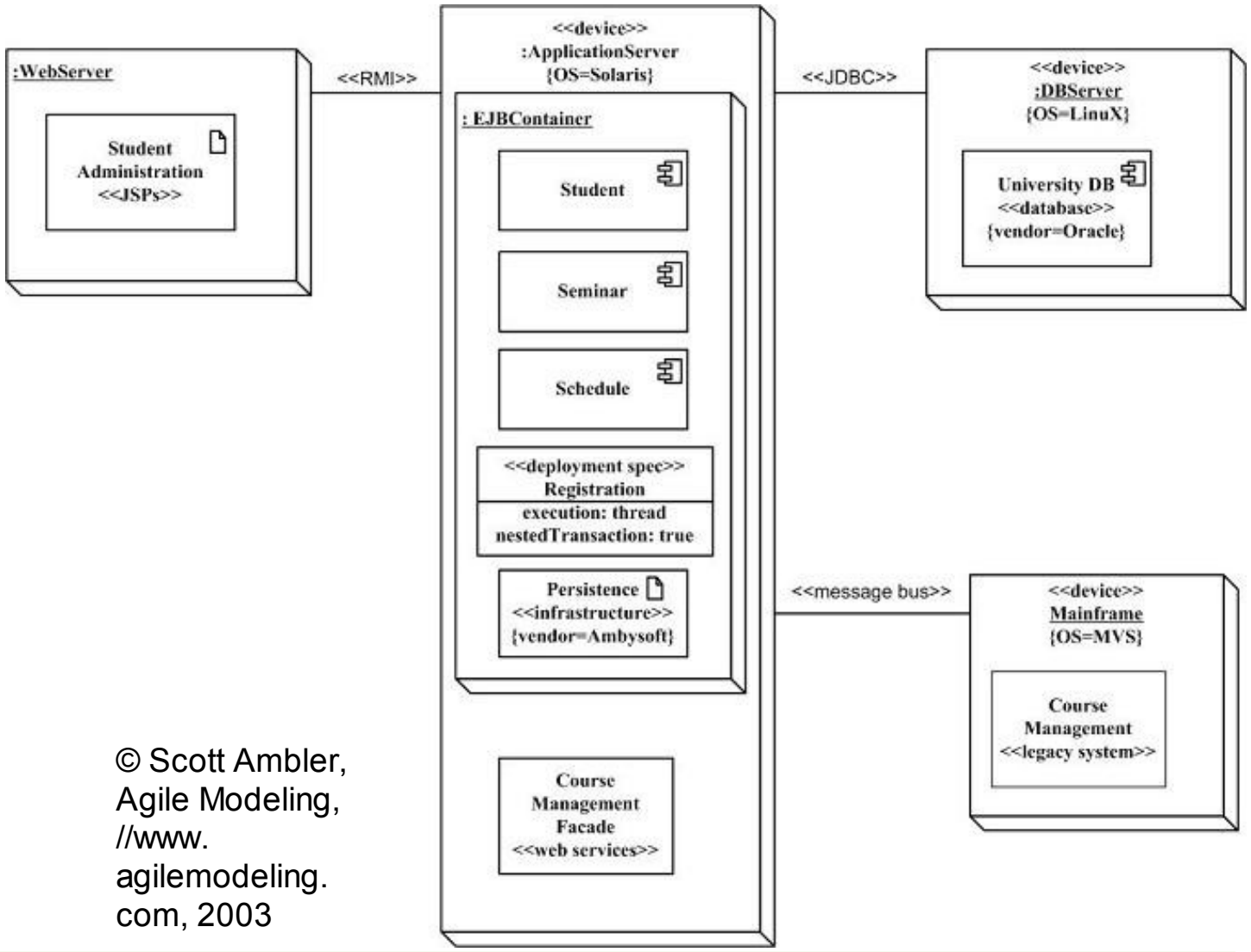
Activity diagram (3)



www.ist-more.org, deliverable 2.1

Deployment diagram

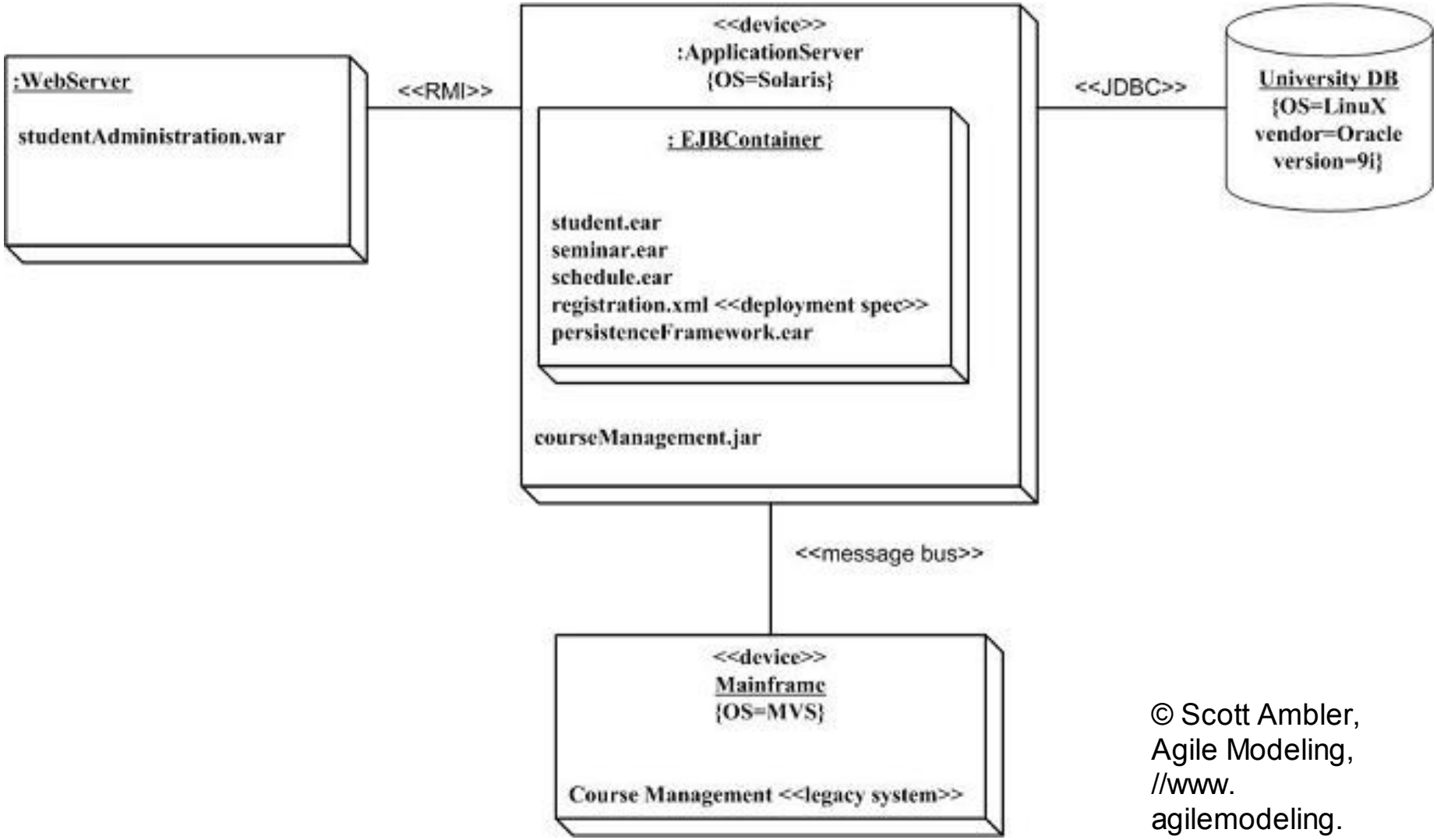
Example including some details:



Describe execution architecture of systems (HW or SW). Important for embedded systems.

© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

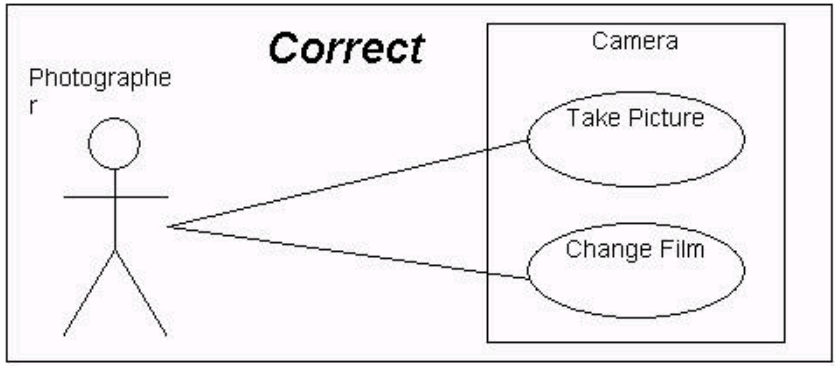
Deployment diagram
- More concise example -

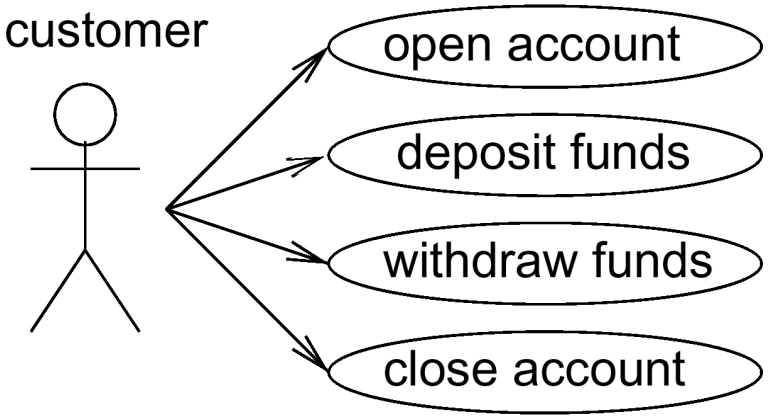


© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

Use case diagram

Captures typical application scenarios



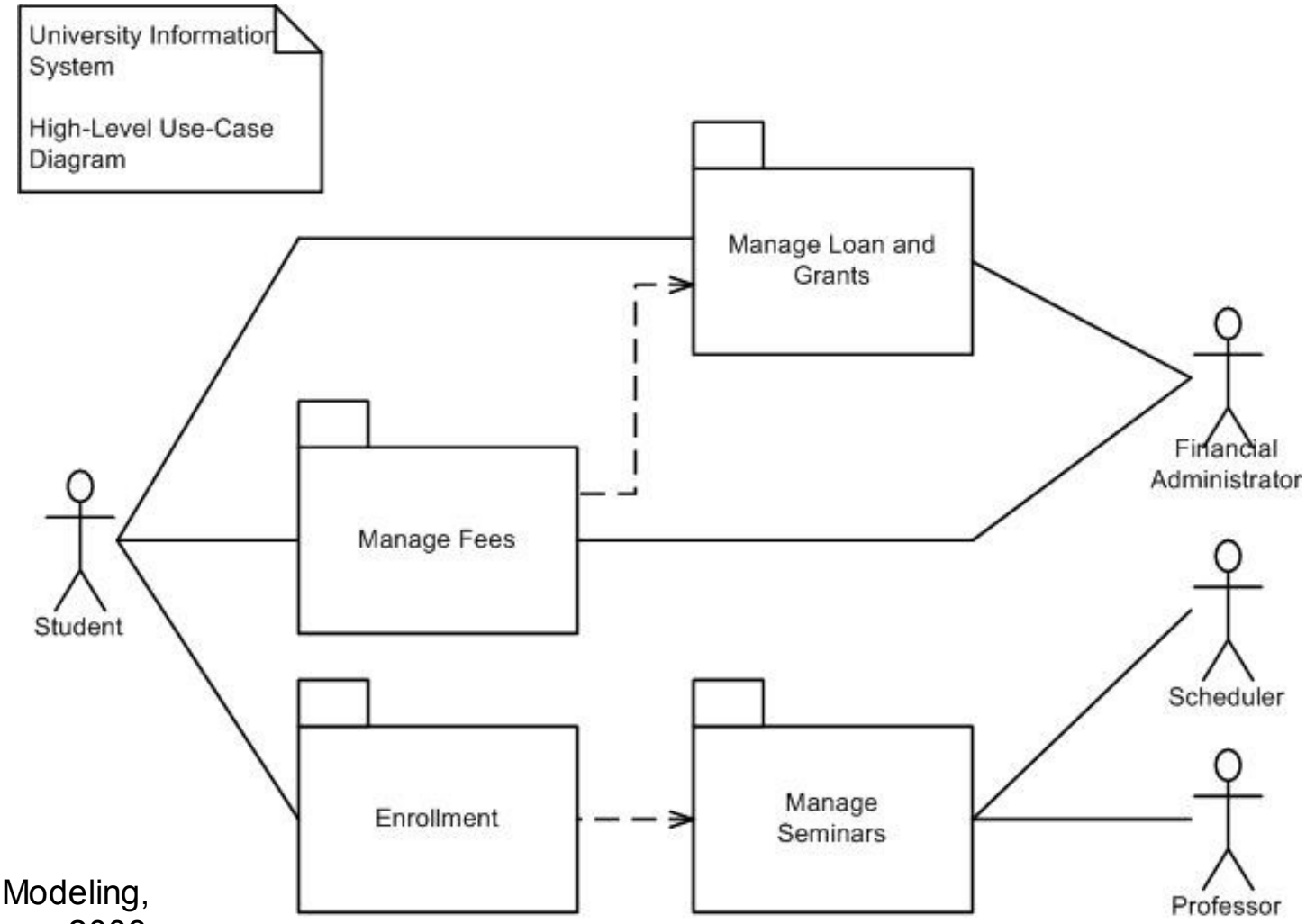


//sds.hss.cmu.edu/courses/Syllabi/ids/271/umlfaq.asp#ucdefinition

Package diagram

Represents the partitioning into packages. Introduces hierarchy.

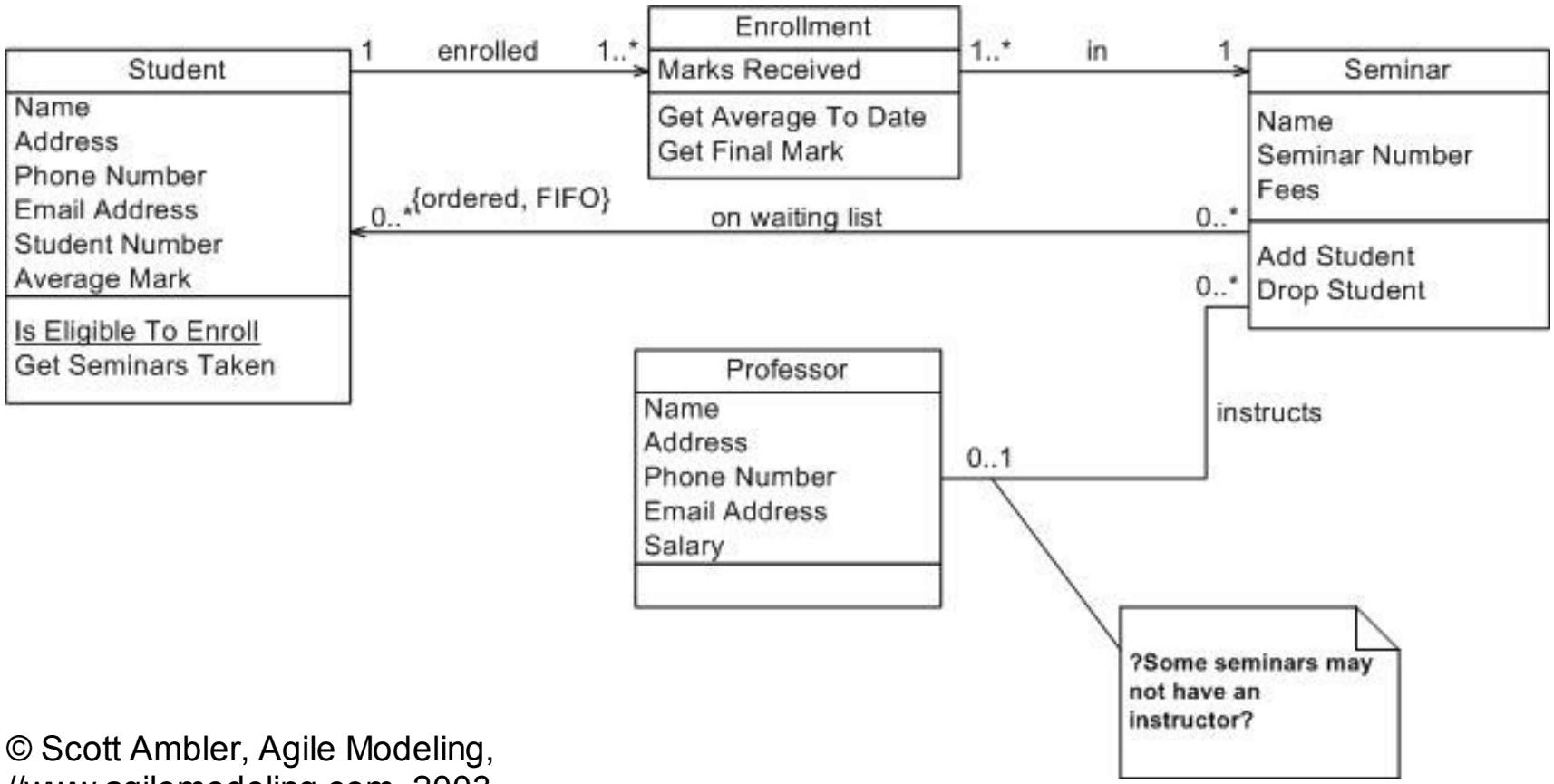
Example: Use case package diagram.



© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

Class diagrams

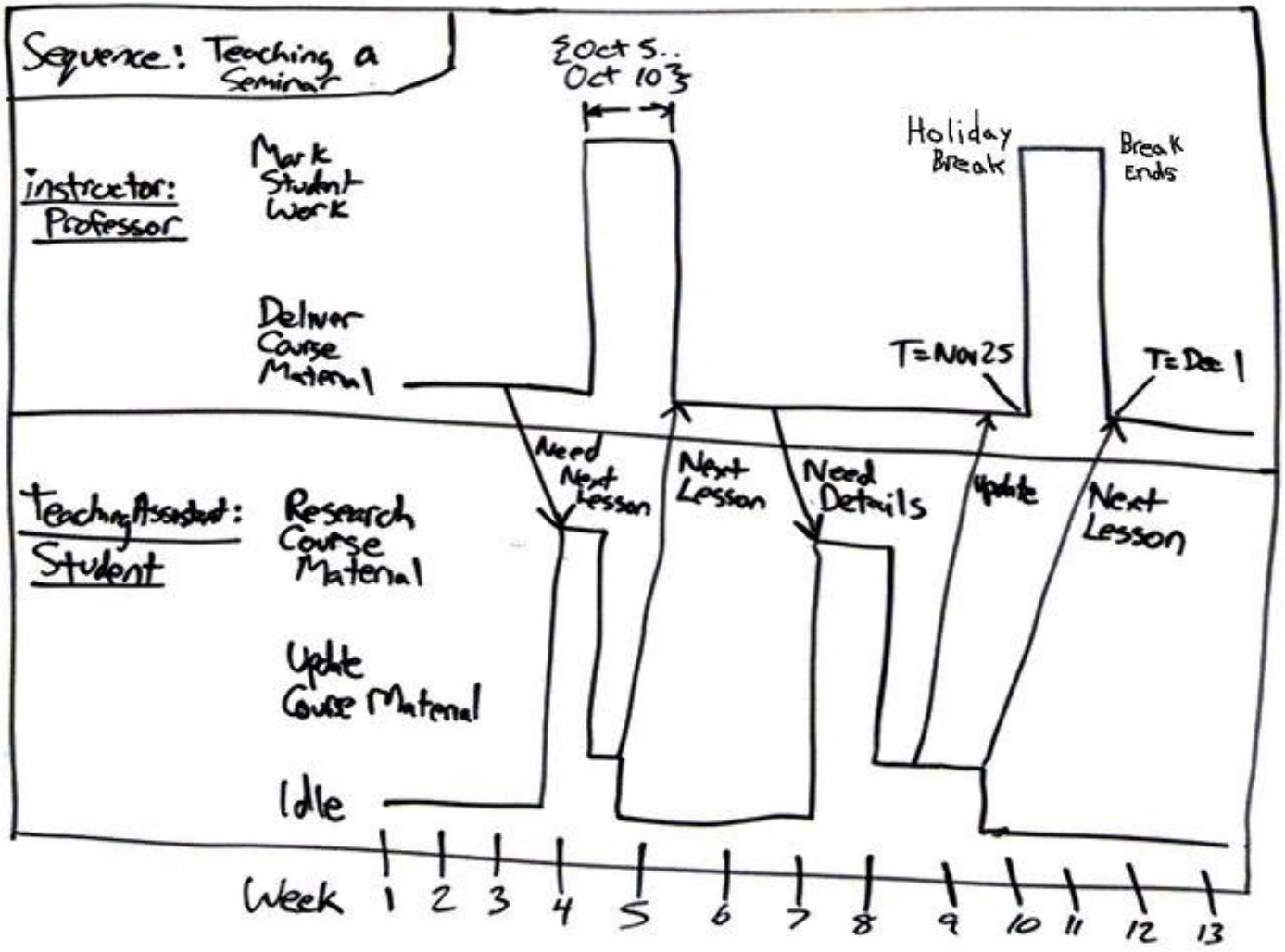
Describe object classes. Example:



© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

Timing diagrams

Can be used to show the change of the state of an object over time.

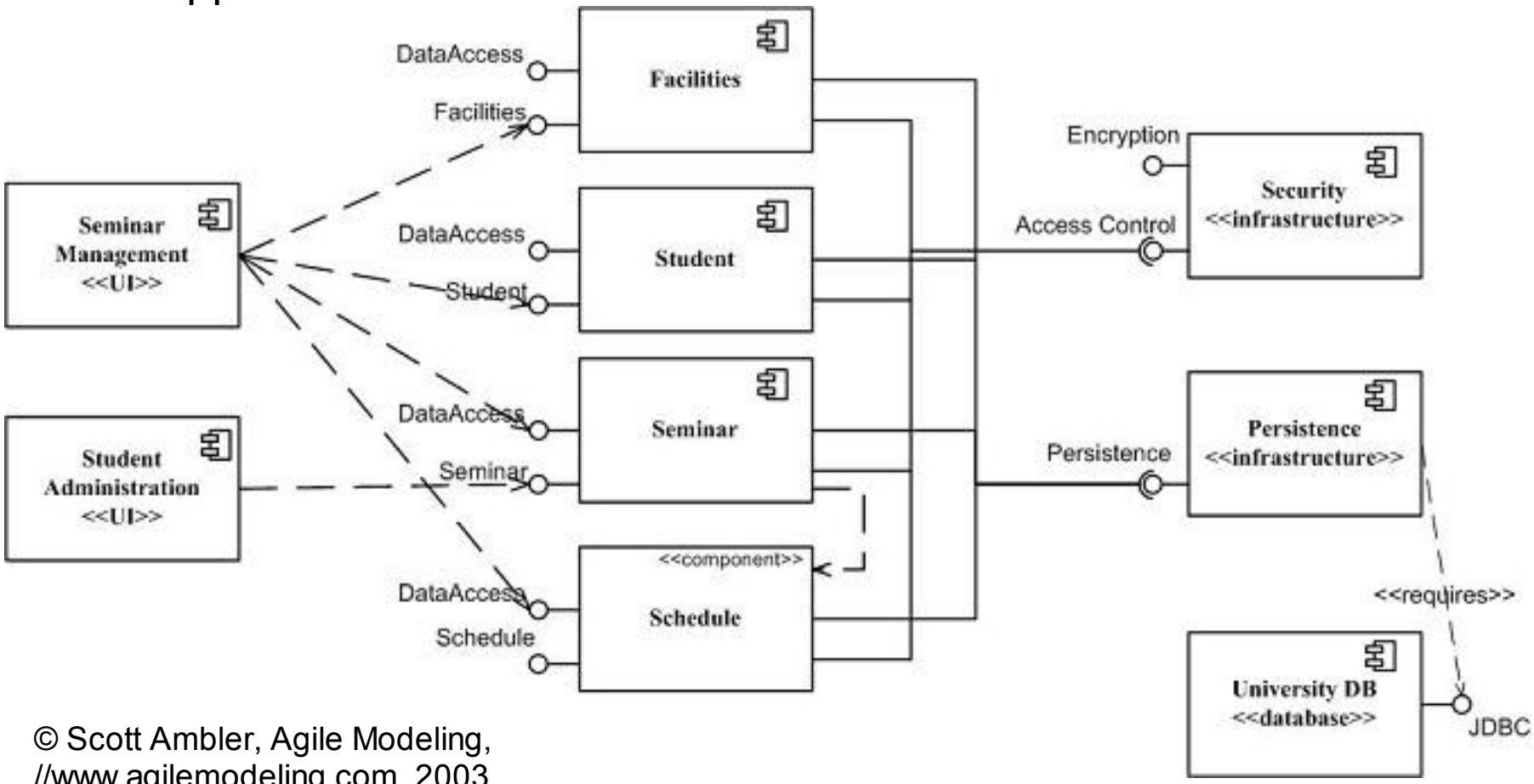


© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

Component diagram

Represent components used in applications:

.. model the business software architecture, the technical software architecture, … . Physical architecture issues, in particular hardware issues, are better addressed via UML deployment diagrams ..



© Scott Ambler, Agile Modeling, //www.agilemodeling.com, 2003

Additional diagrams

- **Communication diagram**
  (called collaboration diagram in UML 1.x)
- **Object diagrams**
- **Interaction overview diagrams**
- **Composite structure diagrams**

} Less frequently used

Evaluation

Precise specification of semantics?
Typically combined with some other “precise” language

UML for real-time?

Initially not designed for real-time.
Lacking features (1998):

- Partitioning of software into tasks and processes
- specifying timing
- specification of hardware components
- Projects on defining real-time UML based on previous work
- ROOM [Selic] is an object-oriented methodology for real-time systems developed originally at Bell-Northern Research.
- „UML profile for schedulability, performance and time“ http://www.omg.org/cgi-bin/doc?ptc/2002-03-02

UML Profiles Relevant for SoC

Existing (OMG)	▪ SPT (Schedulability, Performance, and Timing Analysis) ▪ Testing Profile ▪ QoS and Fault Tolerance
	▪ SysML (System Modelling Language) ▪ UML Profile for SoC
Upcoming (OMG)	▪ MARTE (Modeling and Analysis of Real-Time Embedded Systems)
non-OMG	▪ UML/SystemC (STMicroelectronics) ▪ SPRINT Profile (ST, NXP, Infineon, …)

DATE 2007 UML Workshop – W. Mueller 7/17

Example: Activity diagram with annotations

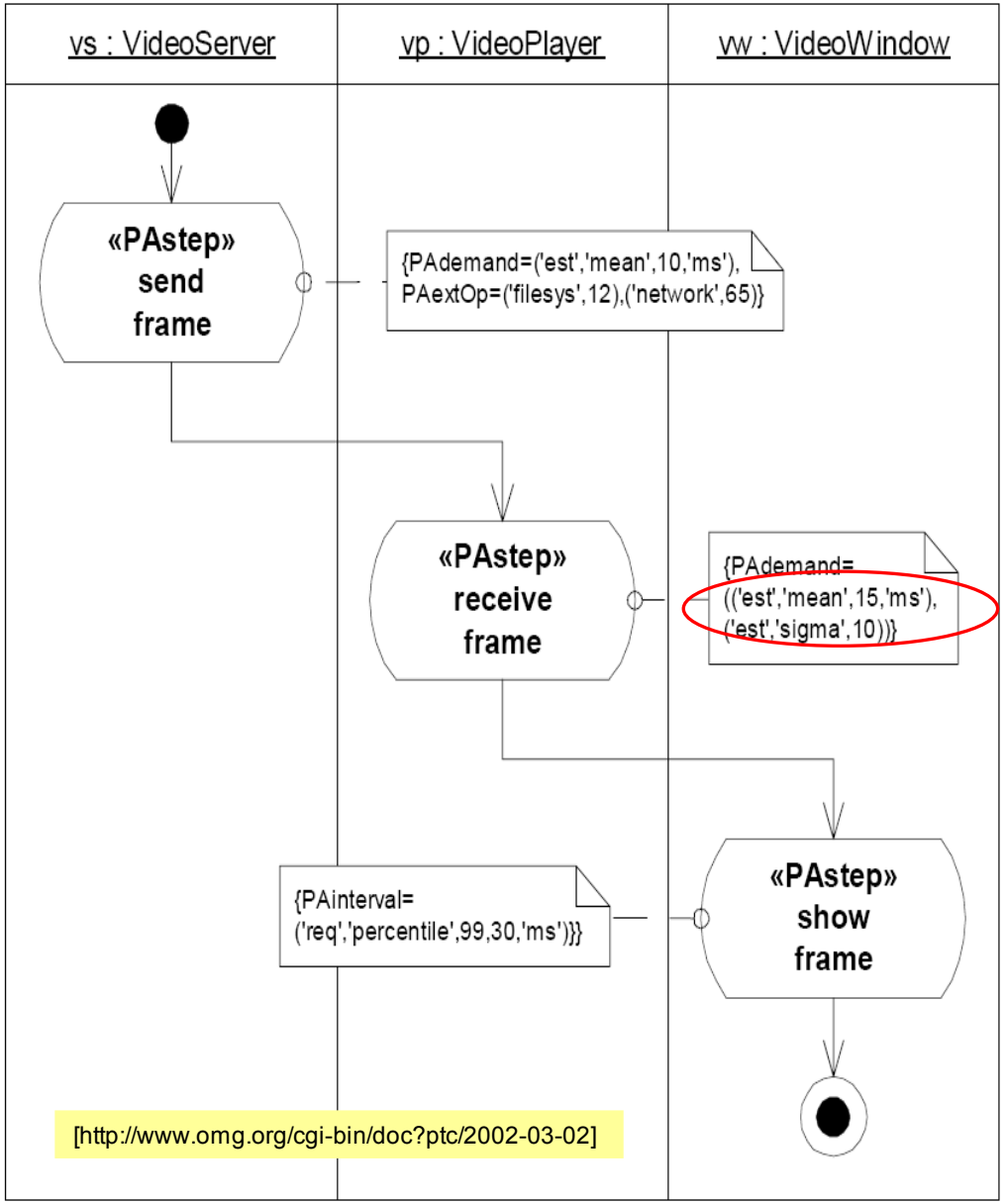


Figure 8-10 Details of the "send video" subactivity with performance annotations

See also W. Müller et al.: UML for SoC, http://jerry.c-lab.de/uml-soc/

UML Profile Summary

- UML Profile comes as class diagrams with constraints, textual outlines (semantics), icons, diagram symbols, …
  - Constraints and behavioral semantics typically leave several issues open (variation points)
- Different OMG profiles of related domains may not be compatible!
- Current OMG UML Profiles are mainly for modelling
- UML Profiles do not come with a formal semantics
- … but Hardware Design is not just modelling
- HW verification and synthesis requires a well-defined and precise behavioral semantics
- Several UML tools already support UML profile definition

Summary

- Message sequence charts
  - Original (ITU) version, Time-Distance diagrams
  - Sequence diagram in UML
- UML
  - State machine diagram (StateChart-like)
  - Activity diagram (extended Petri nets)
  - Deployment diagram (exec. arch.),
  - Use case diagram
  - Package diagram (hierarchy), Class diagrams,
  - Timing diagrams (UML 2.0), UML for real-time?