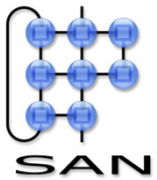
TU/e

Architecture of Distributed Systems 2015-2016

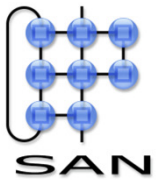
UML (very) basic vocabulary + example homework assignment



R.H. Mak

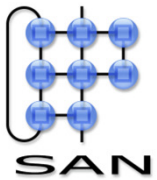
Unified Modeling Language

- A graphical language
  - for visualizing, specifying, constructing and documenting artifacts of a software-intensive system
  - originated from unification of methods by Booch, Rumbaugh (OMT) and Jacobson (OOSE)
  - standardized by the OMG
  - with a well-defined syntax and semantics
    - including the possibility for user-defined extensions
  - with support for Kruchten's 4+1 views
  - with tool support for Model-Driven Architecture (MDA)



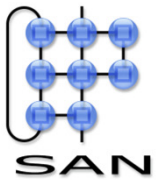
UML: terms and concepts

- System
  - A set of elements organized to accomplish a purpose and described by a set of models possible from different view points
  - Can be decomposed into a set of subsystems
- Model
  - A simplification of reality, an abstraction of a system, created in order to better understand the system
- View
  - A projection of a model, which is seen from one perspective or vantage point and omits entities that are not relevant to this perspective



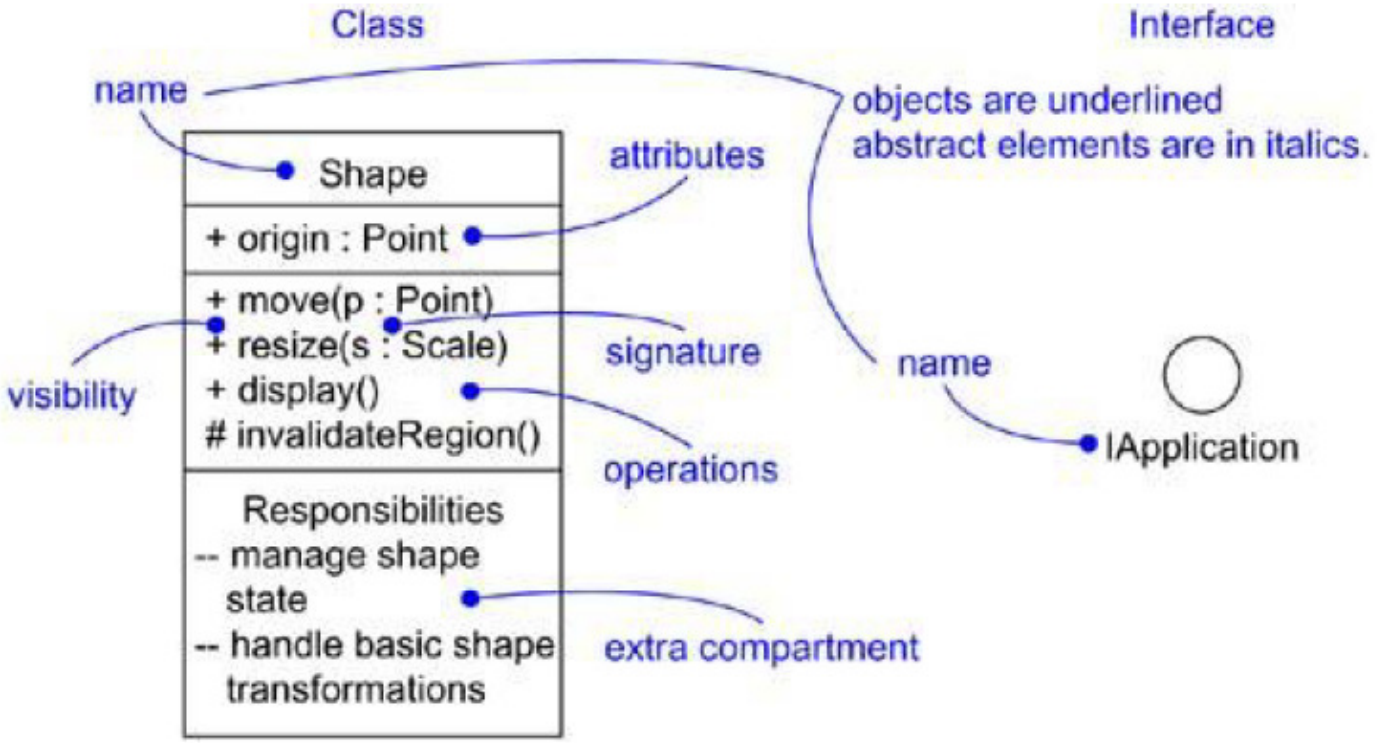
UML vocabulary

- Things
  - Structural, behavioral, grouping, annotational
- Relationships
  - Dependencies, associations, generalizations, realizations
- Structural diagrams
  - class diagram, object diagram, component diagram, deployment diagram
- Behavioral diagrams
  - use case diagram, interaction (sequence or collaboration) diagram, statechart diagram, activity diagram

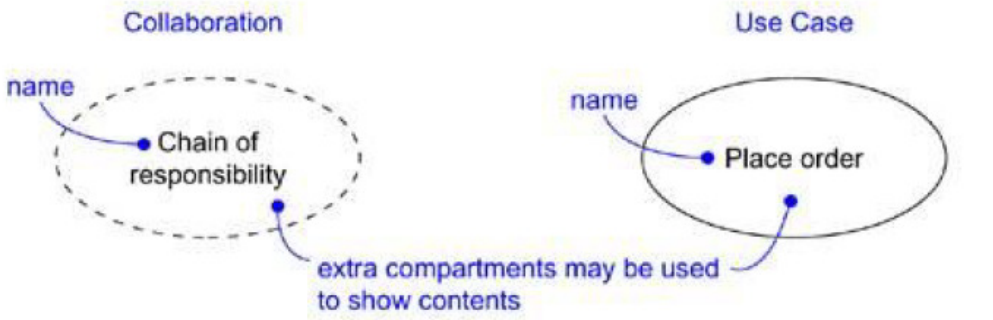


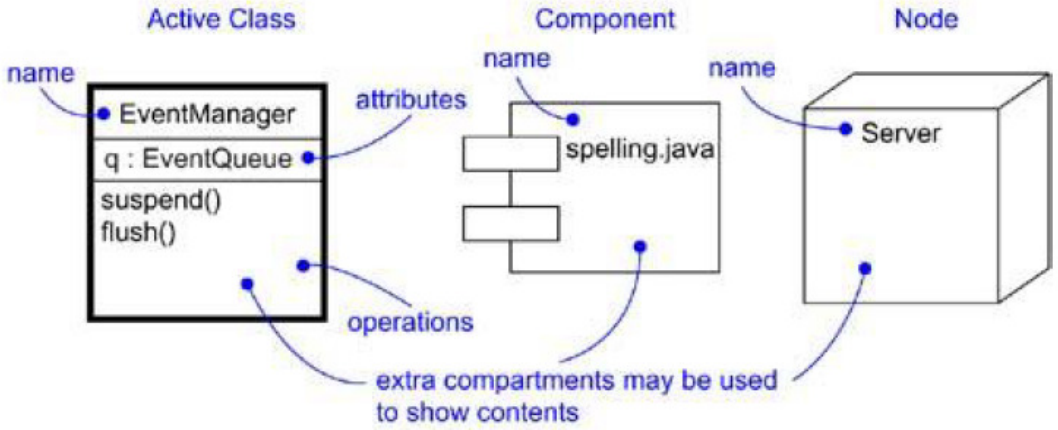
Structural things (1)

- Represent building blocks for modeling the static structure of the architecture
- Dynamic models contain separate building blocks

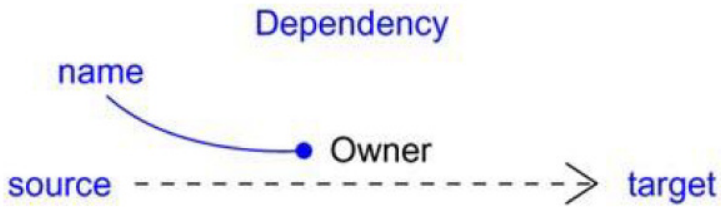


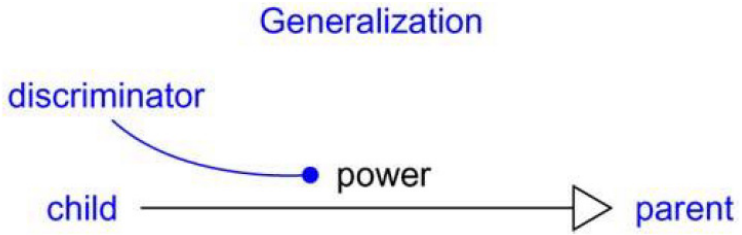
Structural things (2)

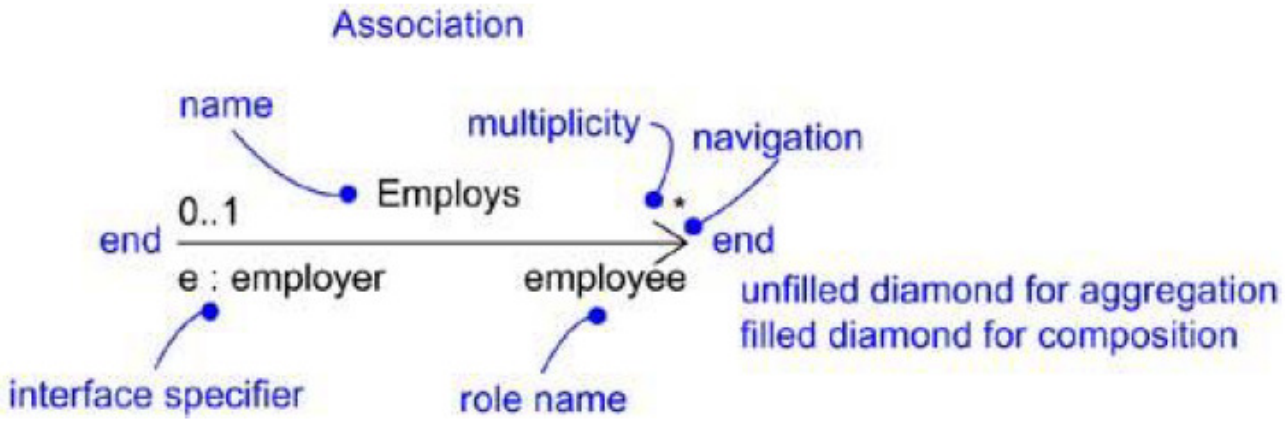




Relationships

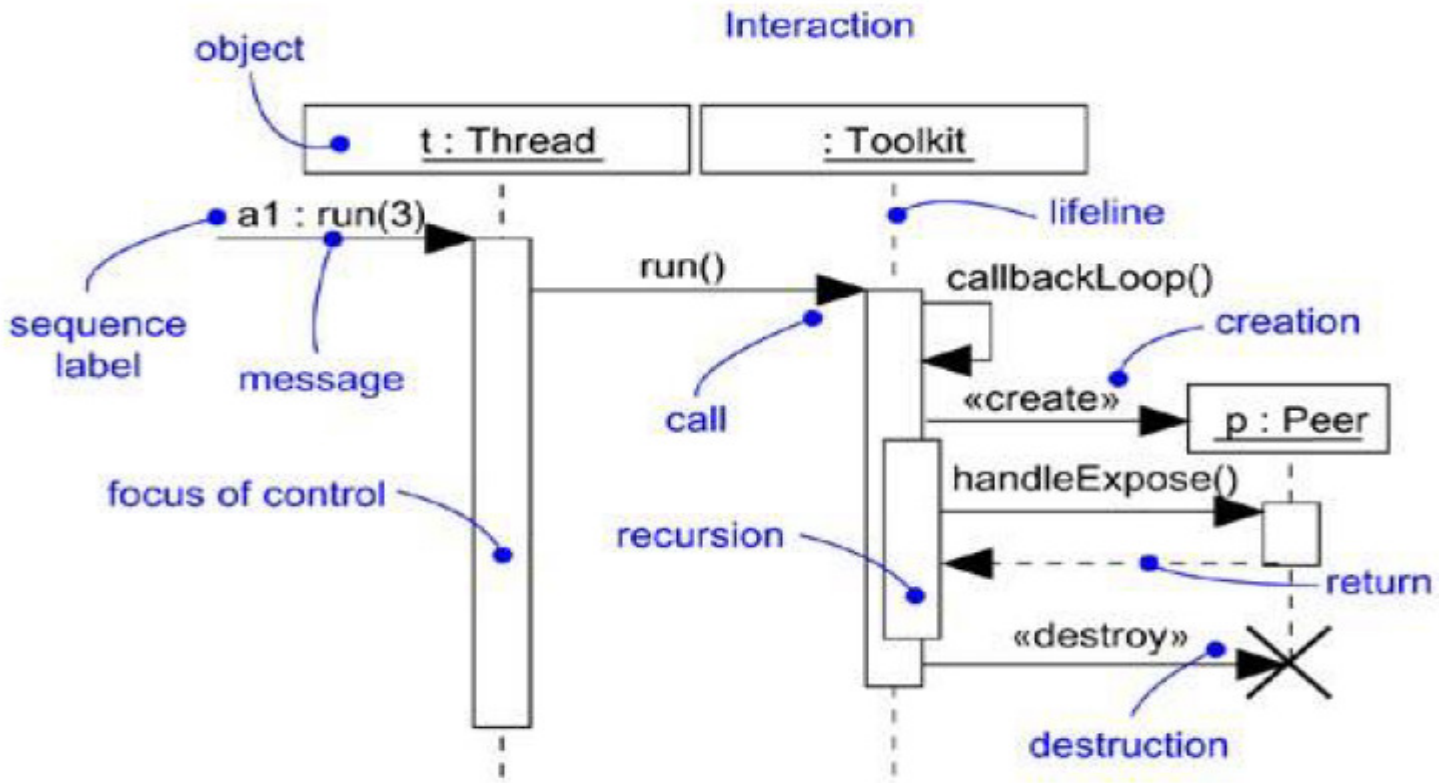




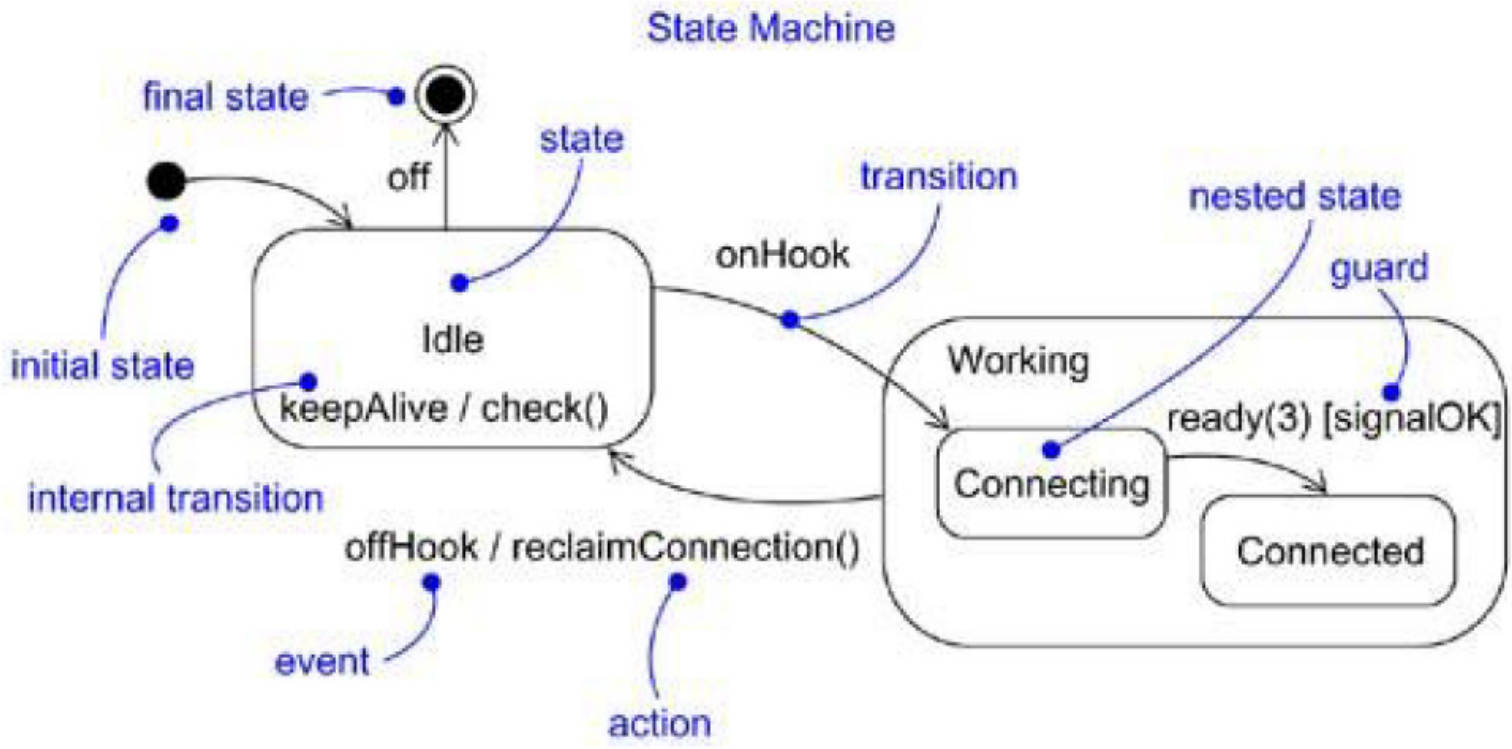


Behavioral things (1)

- Entire model (sequence diagram): contains both building blocks and relationships



Behavioral things (2)



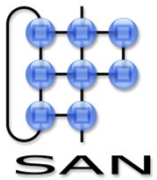
Diagrams (Model kinds)

- Structural diagrams
  - Class diagram
  - Object diagram
  - Component diagram
  - Deployment diagram
- Behavioral diagrams
  - Use case diagram
  - Interaction diagram (sequence & collaboration)
  - State machine diagram
  - Activity diagram

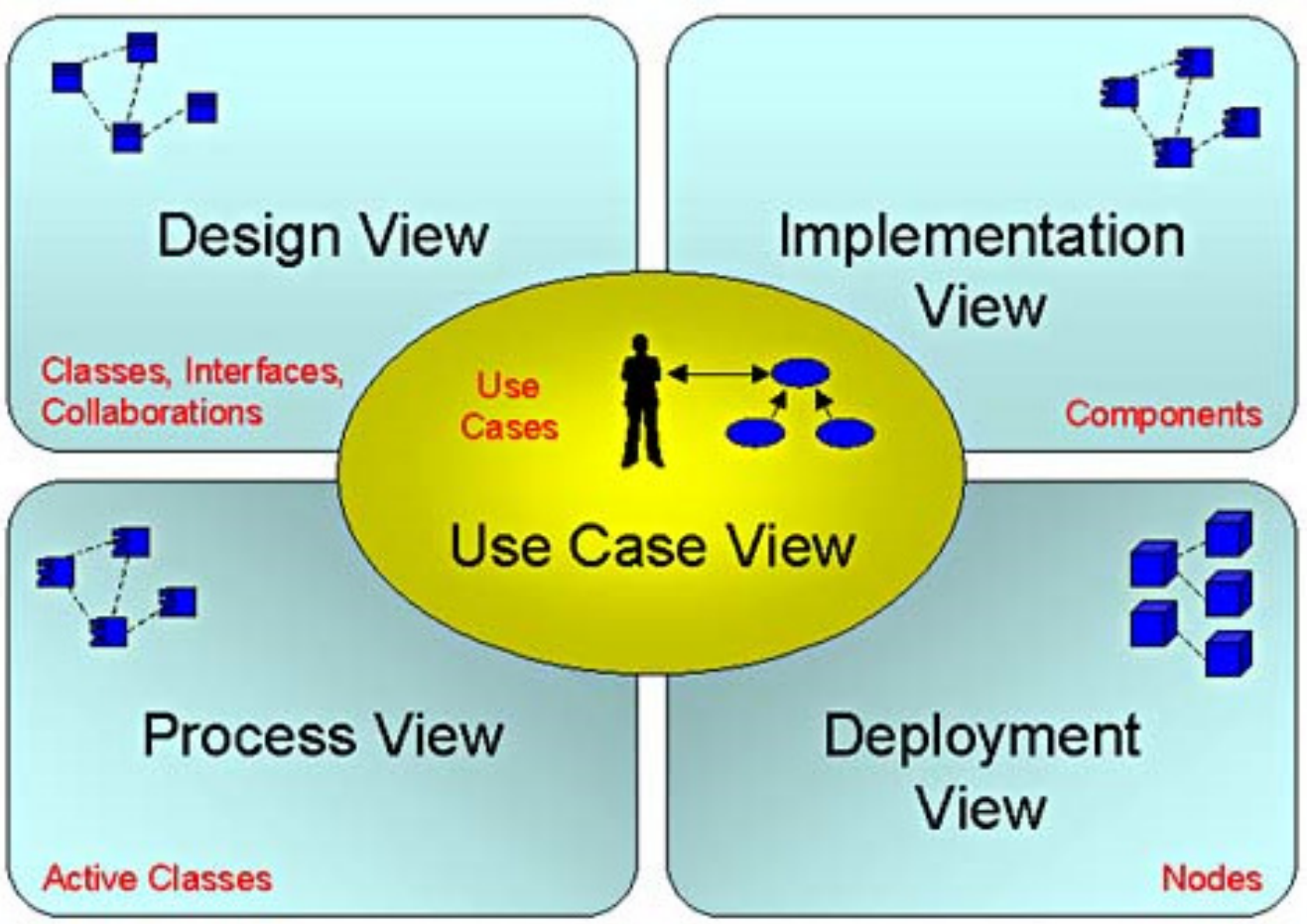
This is not all:

see book by Booch, Rumbaugh, Jacobson,

or tutorials on the WWW

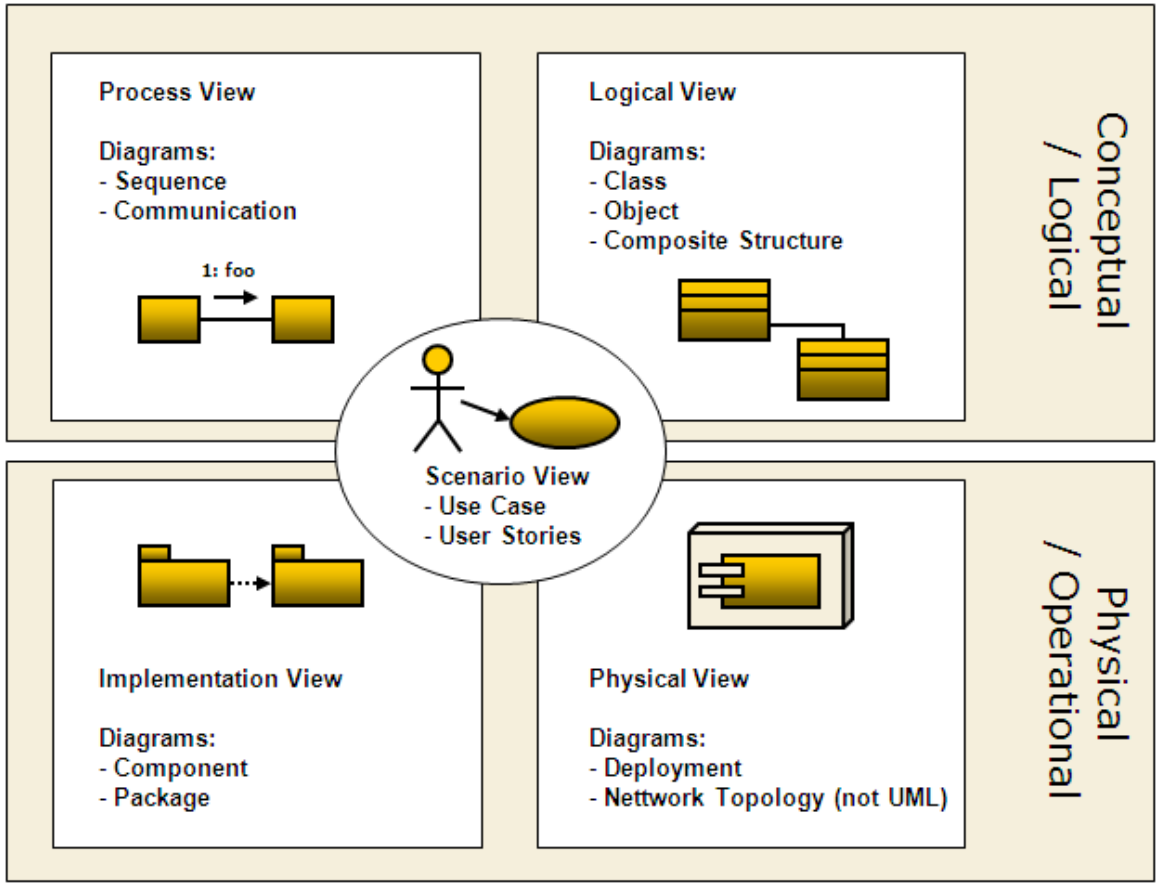


UML version: Kruchten 4+1 view



Specifies UML “things” used to indicate AEs.
Uses UML vocabulary for the views

UML version: Kruchten 4+1 views

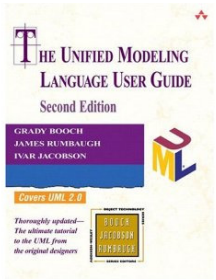


Just to warn you that the vocabulary used to identify views varies!

Information on UML

Alexander Serebrenik, Lectures slides from course 2IW80 *Software specification and architecture*, 2014-2015, URL http://www.win.tue.nl/~aserebre/2IW80/2014-2015/

Grady Booch, James Rumbaugh, Ivar Jacobson, *The Unified Modeling Language User Guide, Addison-Wesley, 1999.*

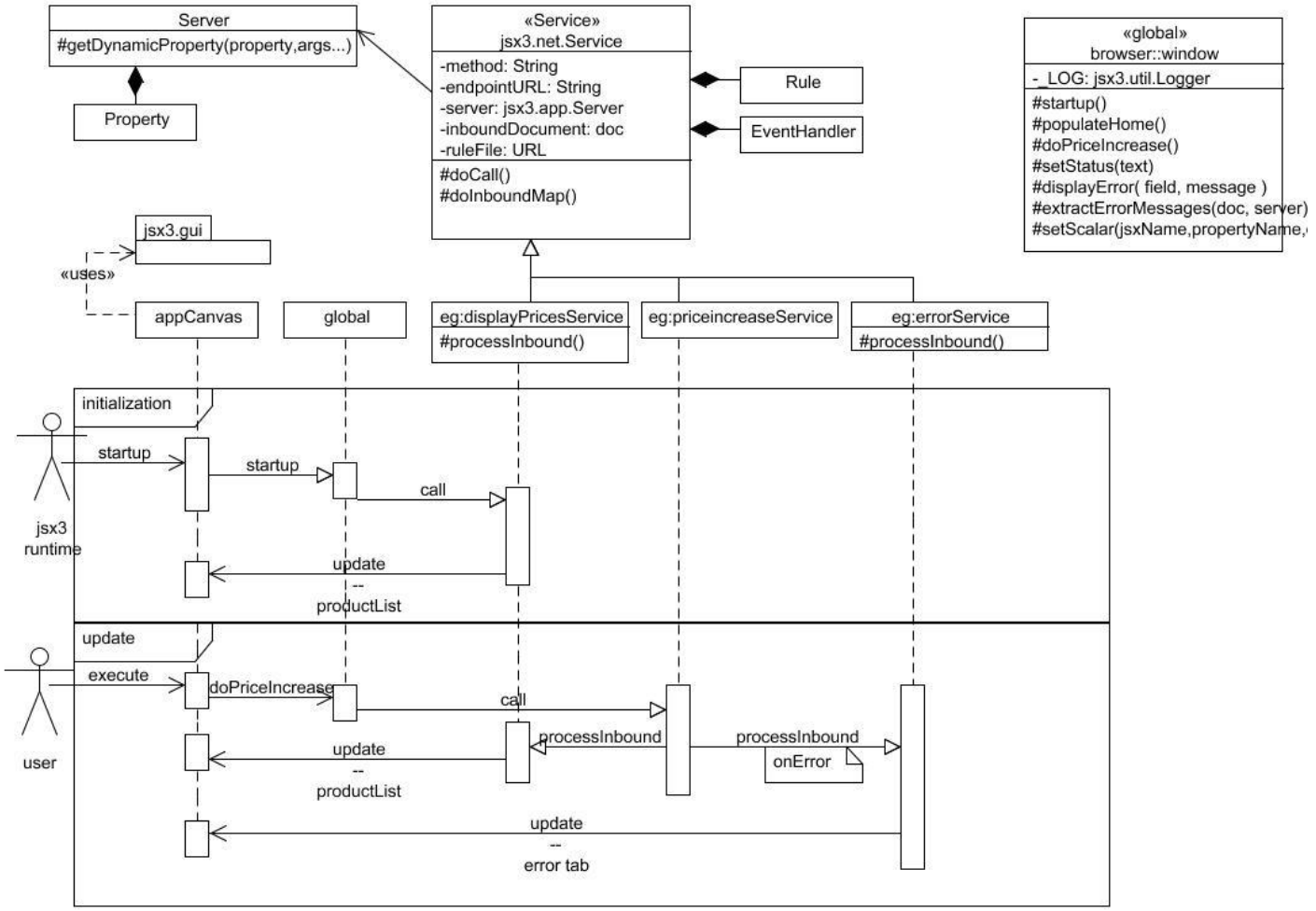


On the web there is an abundance of information and tutorials on UML of varying quality. In the end, the authoritative source is

OMG UML standard and related documents

Taken from:

http://i1.wp.com/media.techtarget.com/tss/static/articles/content/AjaxandSpring

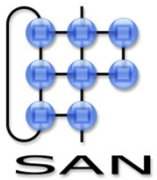


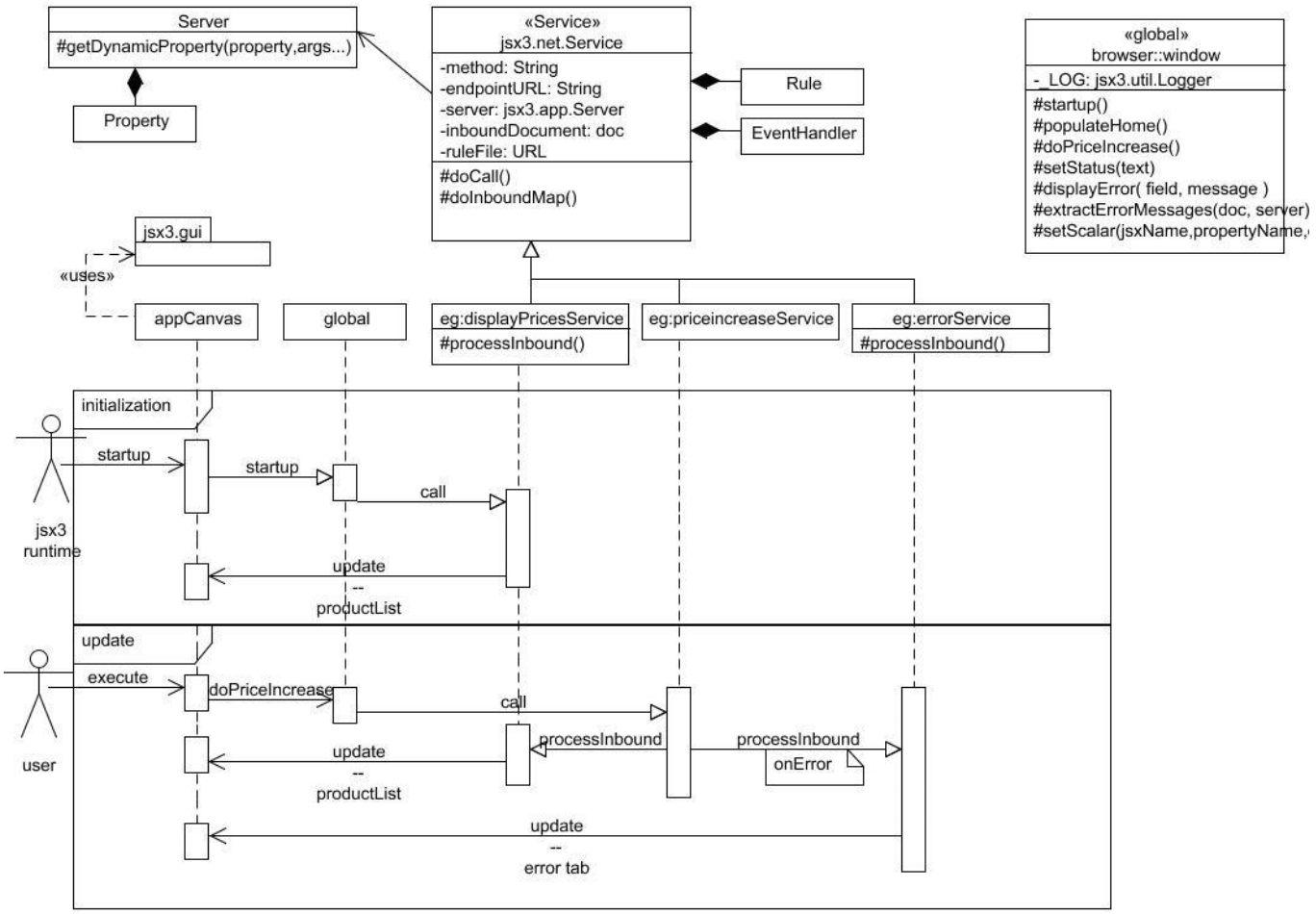
Questions about the model

TU/e

1. What building blocks do you see? What do they represent? Are they conceptual or physical?
2. Same questions as 1, but now for connectors?
3. To which view(point)s does the model belong?
    1. Motivate why, and identify corresponding stakeholders and their concerns.
4. Which of the following EFRs are addressed (Y + motivation | N)?
    1. Performance/scalability, availability/reliability, security, maintainability, other?
5. Is there a concept of distribution (Y + motivation | N)?
6. Comment on the clarity/semantics of the diagram
    ☺ | 😐 | ☹ , plus motivation

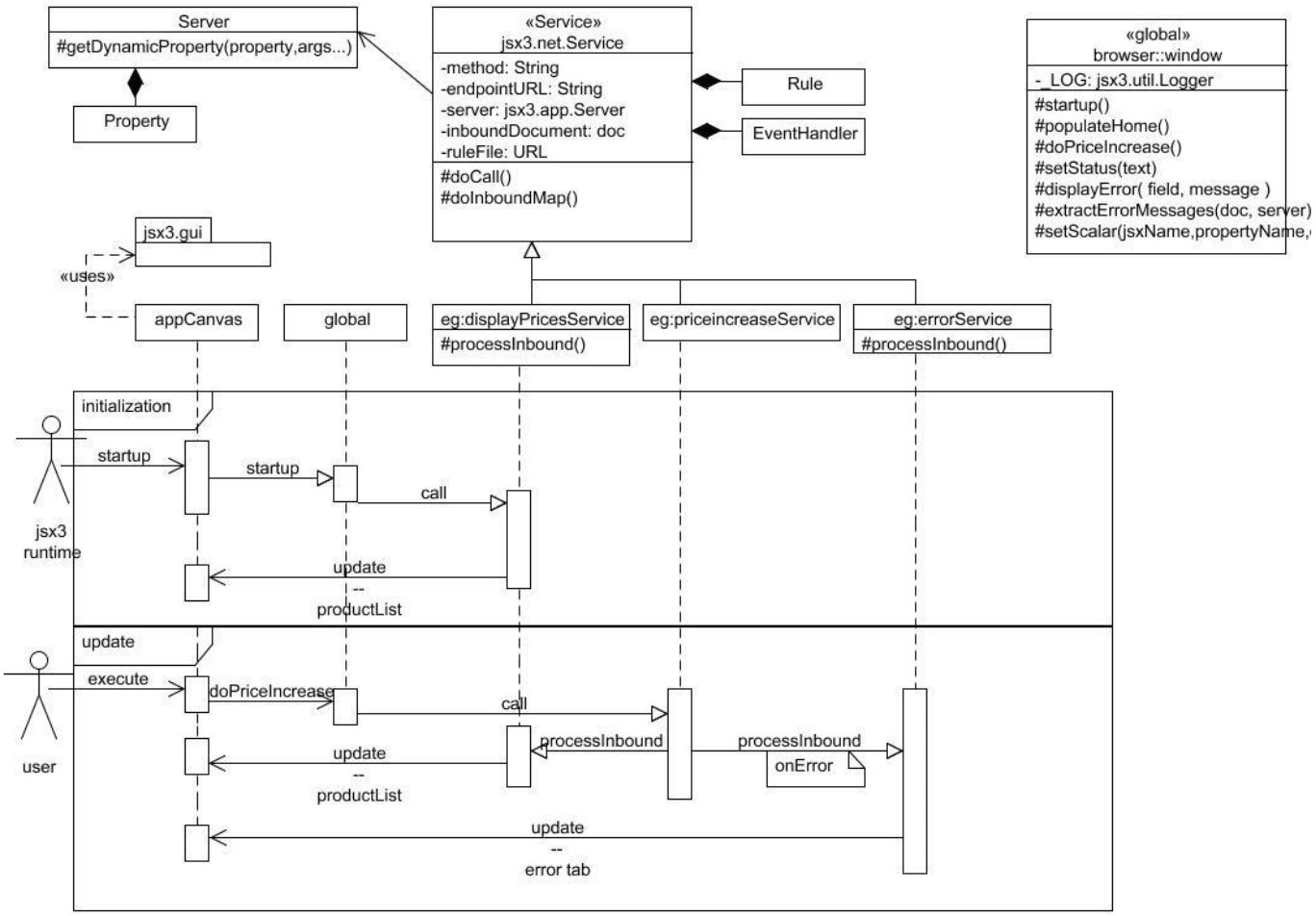
Keep you answers crisp!





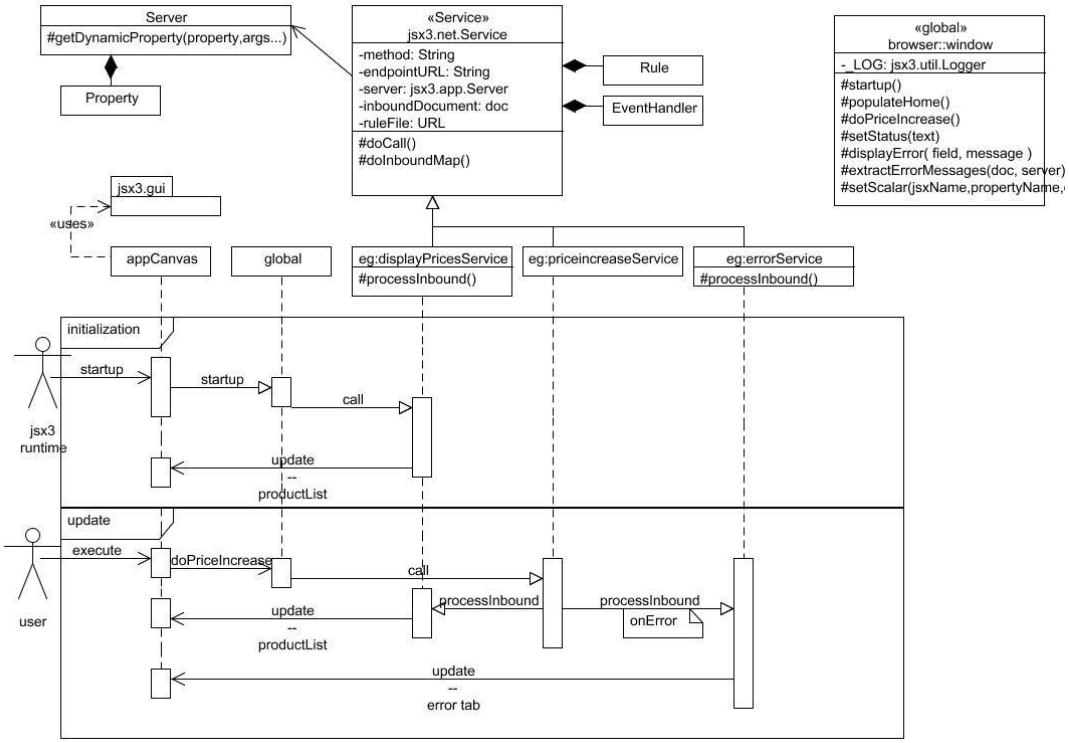
Building blocks:

- Services, timelines, two scenarios (initialization and update) (C)
- A global library (C)
- Actors: human and runtime environment (C) and a server (C or P)



Connectors: all (C)

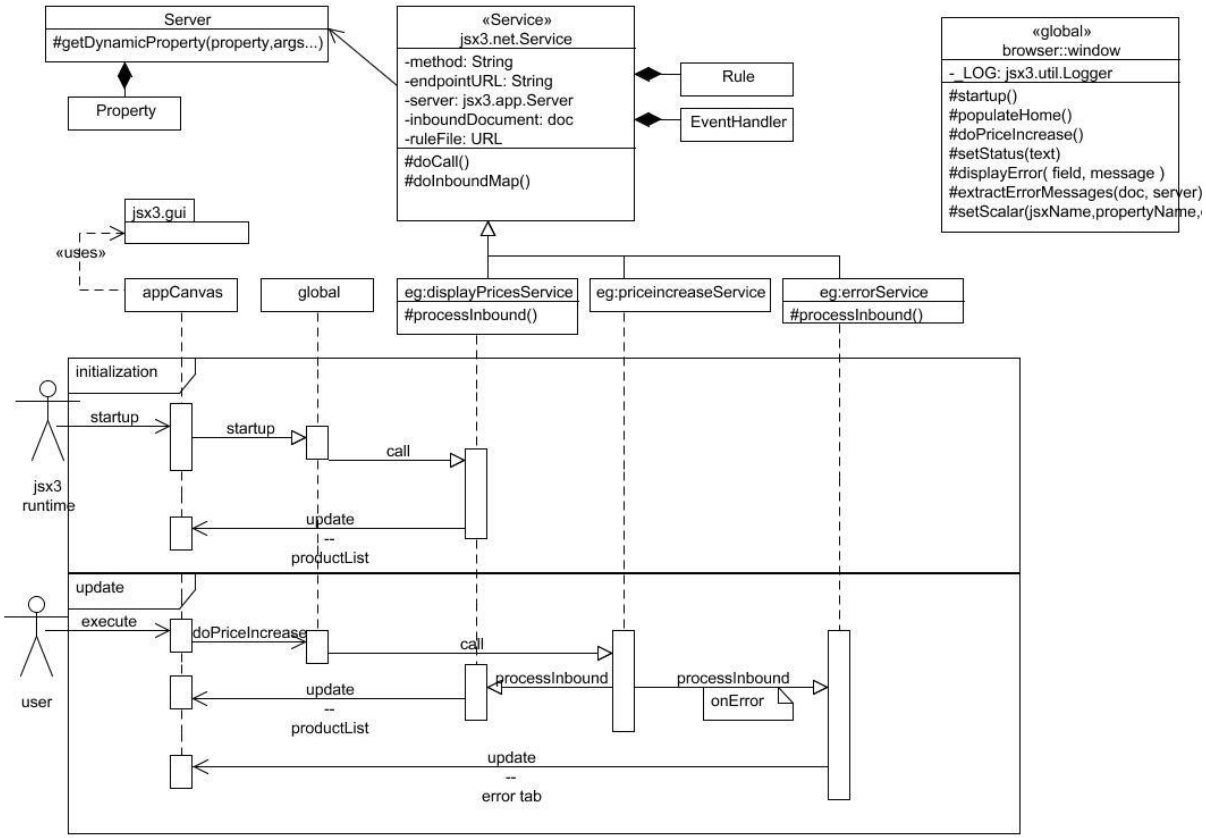
- Message, method invocation
- Is_a relationships, to indicate special services
- Part-of relationships (black diamond arrows)
- Dependency relationship (uses)



View – concern – stakeholder (1..*):

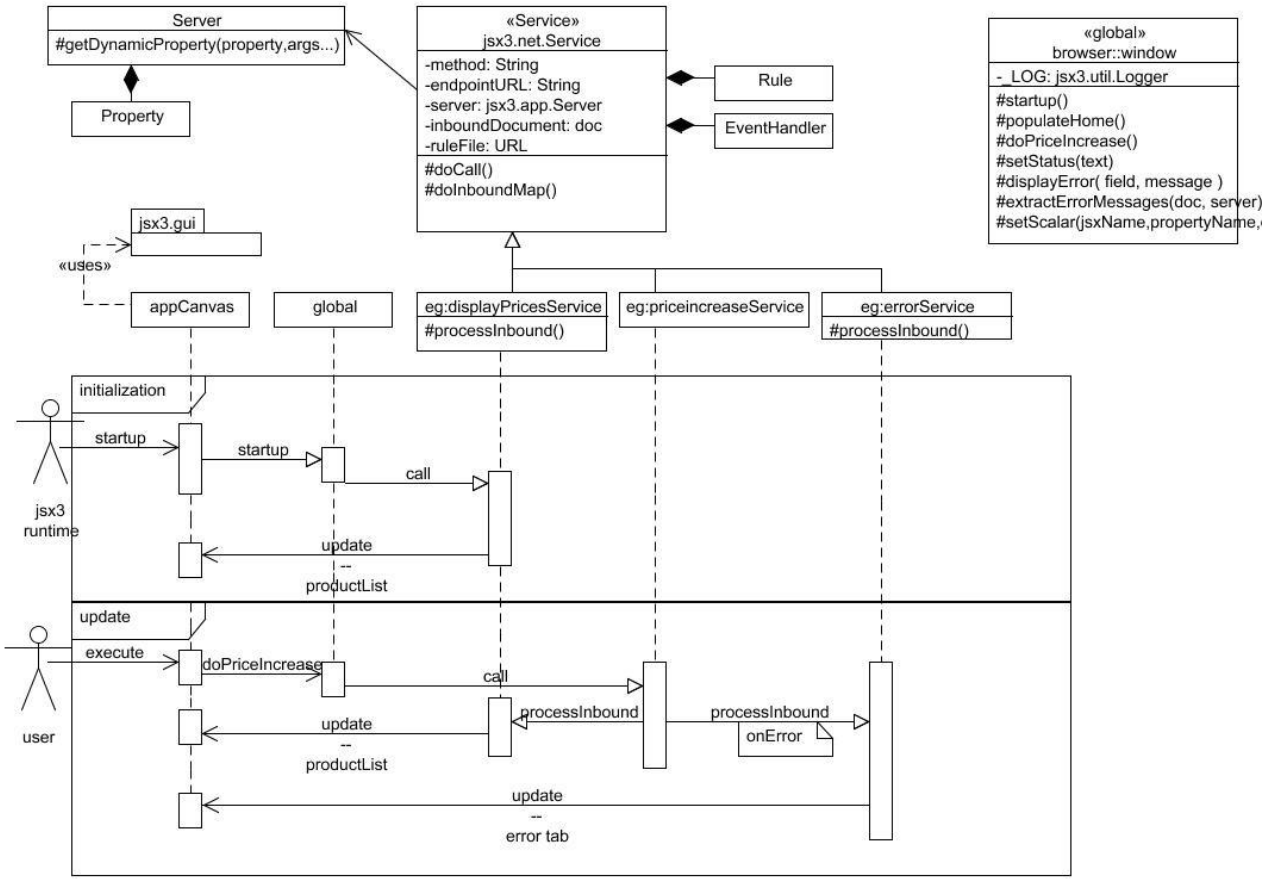
- Development view : the class structure of some entities is shown which is of interest to programmers.
- Process view: system integrators, testers can see how the various parts interact. Developers (programmers) can see which class methods are used.
- Scenario: testers can run these to test interaction. Users can see how to interact with the system. Because of the latter, it can be argued that the model also belongs to the logical view.

There are even some indications of a deployment view, since a technology is mentioned (jsx3.net), as are a runtime elements such as a server



Extra-functional requirements (Y + motivation) /N :

- Security: N
- Availability & reliability: Y
  - There are error methods exhibited
- Maintainability: Y
  - Class structure is shown at a level useful for implementation and modification
- Performance and scalability: N



Distribution (Y + motivation) /N:

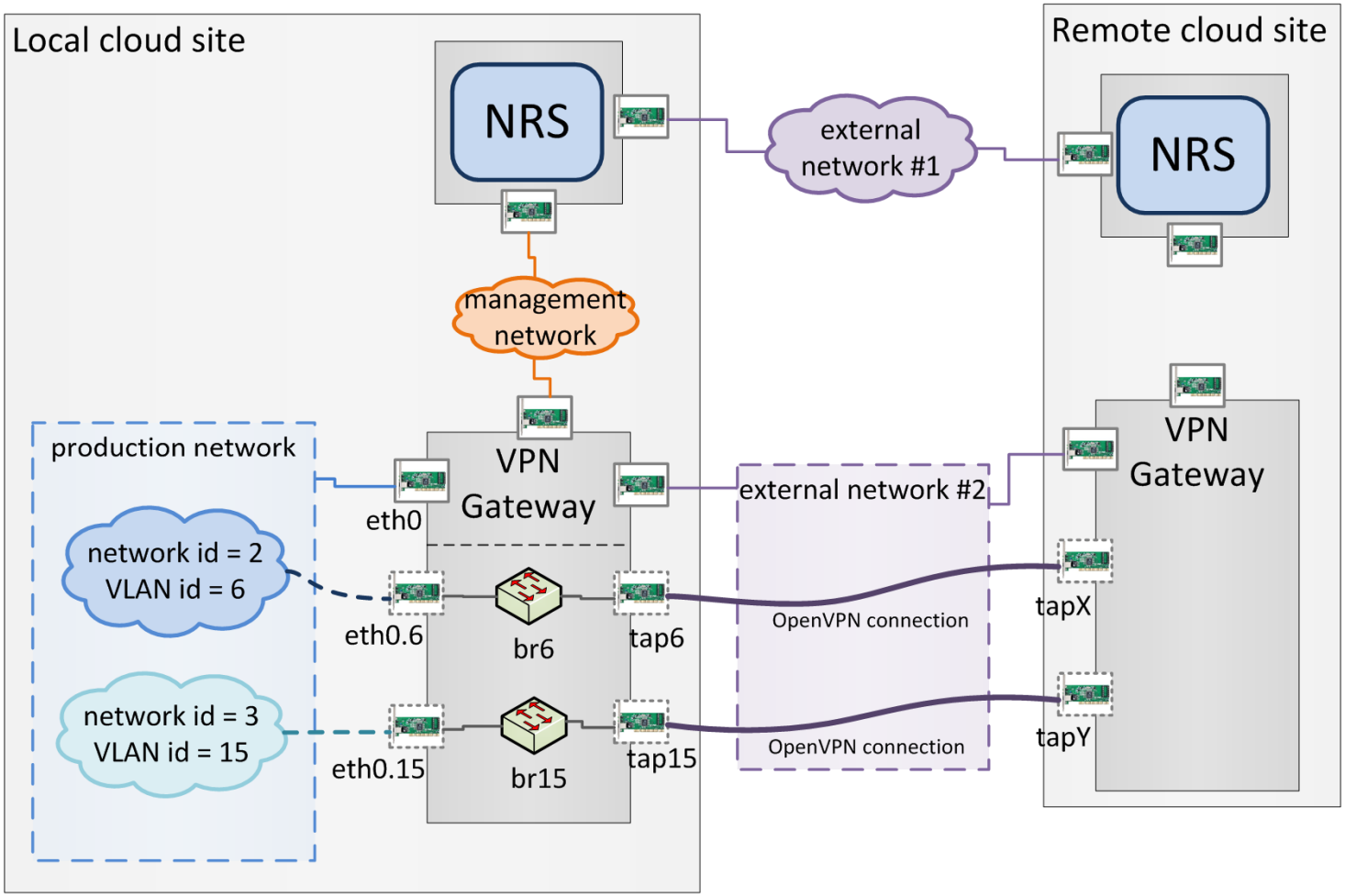
Clarity/Semantics (☺ | 😐 | ☹) + motivation:

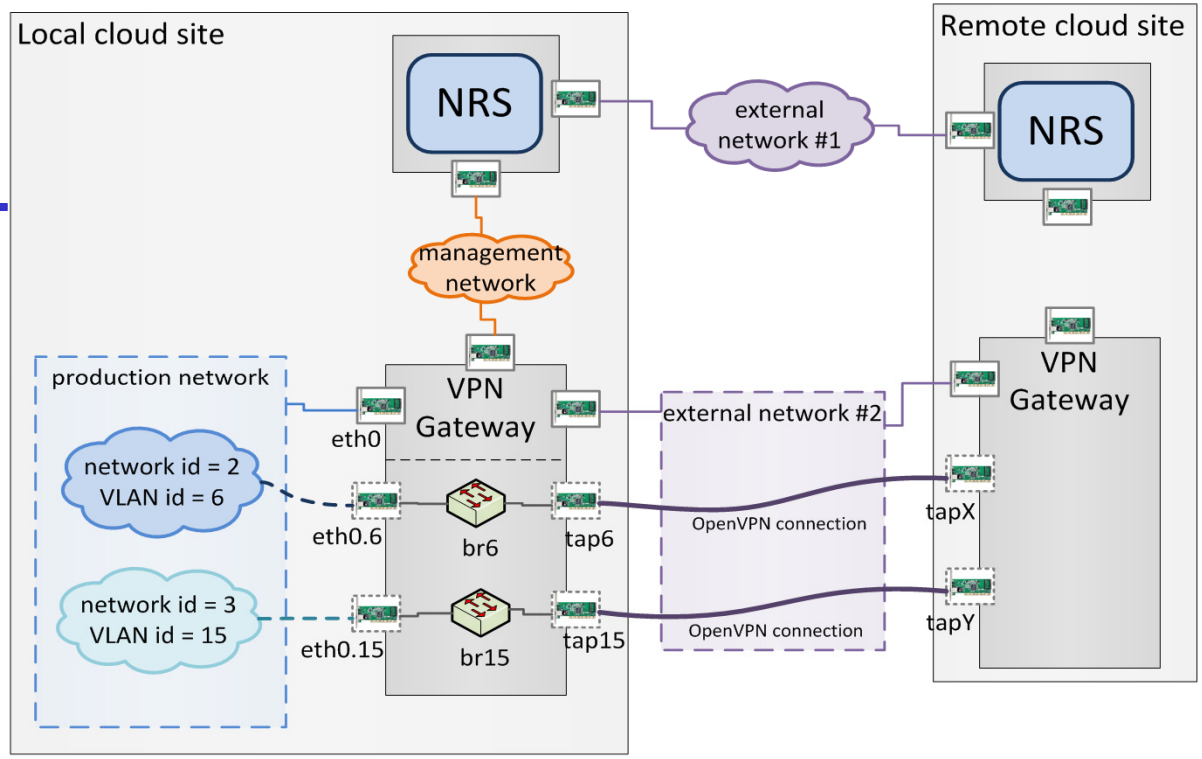
- A mix of a class diagram and a sequence diagram. So on the one hand, a lot of information crammed into a single model, but on the other hand this guarantees consistency between development and process view models.

Taken from:

NRS: a system for automated network virtualization in iaas cloud infrastructures

http://dx.doi.org/10.1145/2465829.2465832



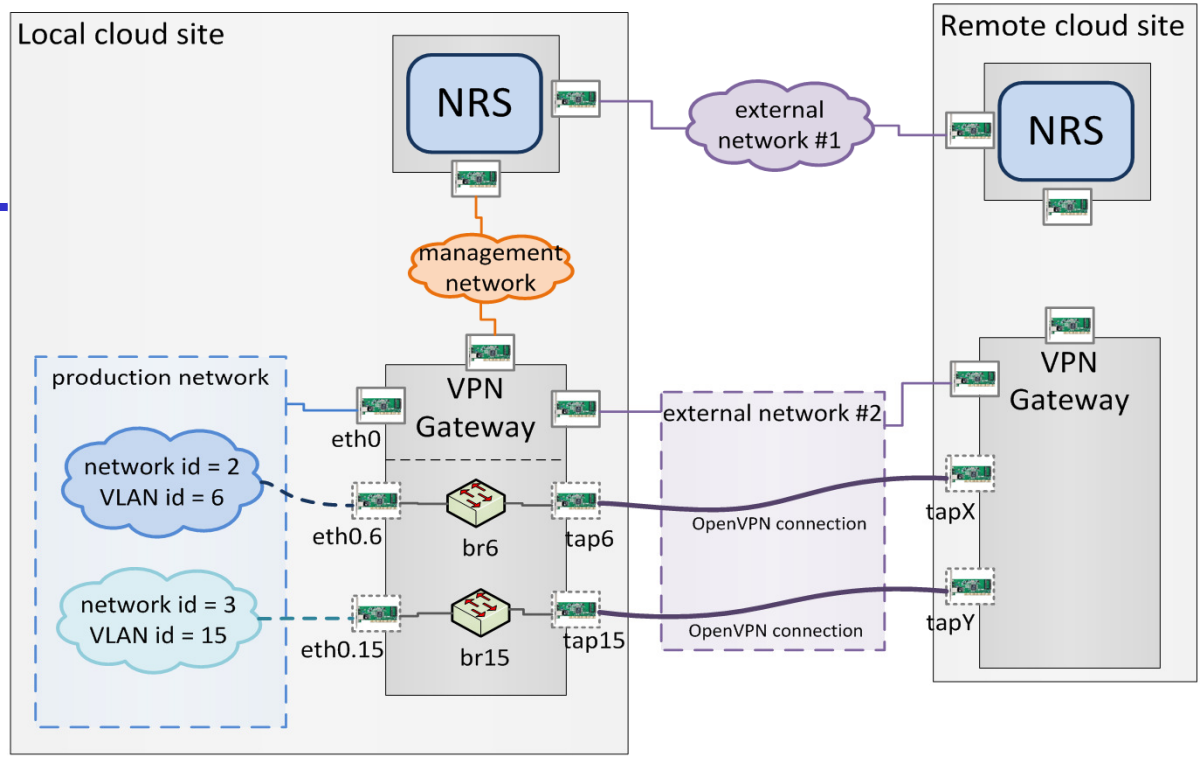


Building blocks:

- Sites, machines, gateways, bridges, network cards, networks (LAN, WAN) (all P)
- VLANs (C)
- NRS controller process (C)

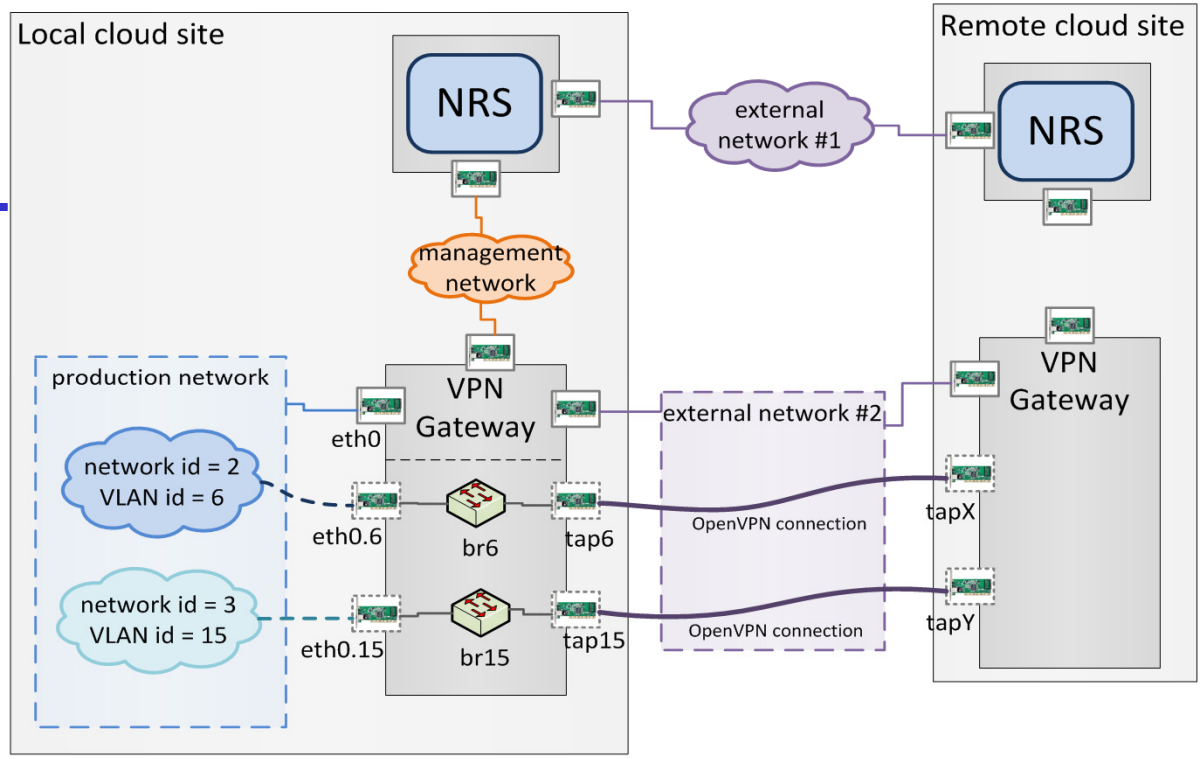
Connectors: all (C)

- Bindings of network cards (interfaces) to networks ,
- or other cards (OpenVPN connections)



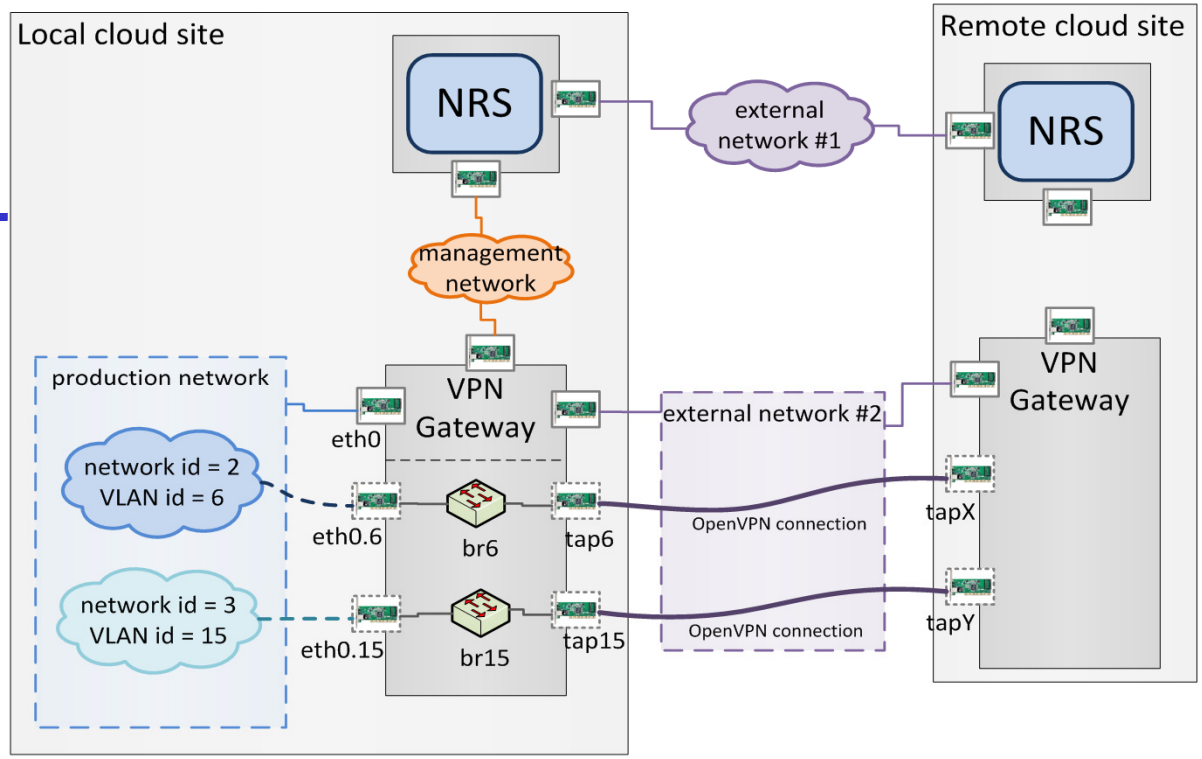
View – concern – stakeholder

- Physical view, because we see devices and machines. Important for suppliers, system engineers and network operators and administrators who have to put the system together and operate it
- Logical view, because it gives an overview of intra- and inter cloud network services. Useful system acquirers and developers. It also indicates a management network which is used for configuration of the network and therefore of interest to the network operators and administrators.



Extra-functional requirements (Y + motivation) /N :

- Security: Y
  - because VLANs isolate network traffic of various cloud tenants
- Availability & reliability, maintainability, performance & scalability: N
- Others:
  - Operability and configurability, through the management network



Distribution (Y + motivation) /N:

- Because we can see sites, multiple machines, multiple networks

Clarity/Semantics (☺ | 😐 | ☹) + motivation:

- No clear drawing conventions
- Physical configuration clear, but not much context on the intended purpose of the system. Cloud tenants require their applications to run on isolated networks that have no interference from other applications. Therefore VLANs have to be configured/deployed on a physical network.