UML 2.0 Diagrams, Petri Nets and Development of an Executable Architecture to predict performance

A Systems Engineering Methodology

Paul Barr Ph.D., PE, **Jody Pettis MSE**
poobarr@mitre.org jpettis@mitre.org
The MITRE Corporation
260 Industrial Way West
Eatontown, N. J. 07724

Agenda

- **Architecture Defined**
- **Modeling & Simulation**
- **Diagrams in UML 2.0**
- **What's New in UML 2.0**
- **What is a Petri Net**
- **Addition of Petri-Net to examine Performance**

Software Architecture Defined

From the book *Software Architecture in Practice (2nd edition)*, (Bass, Clements, Kazman; Addison-Wesley 2003:

- **The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them**
  - **"Externally visible" properties refers to those assumptions other elements can make of an element, such as its provided services, performance characteristics, fault handling, shared resource usage, and so on.**
  - ***Architecture defines elements*.**
    - **The architecture embodies information about how the elements relate to each other.**
  - **Architecture specifically *omits* certain information about elements that does not pertain to their interaction.**
  - **An architecture is foremost an *abstraction* of a system that suppresses details of elements that do not affect how they use, are used by, relate to, or interact with other elements.**

What Is A Model ?

- **A Representation of an object, a system, or an idea in some form other than that of the entity itself. (Shannon)**
  - **Mathematical:**
    - **Analytical queuing models,**
    - **Linear programs,**
    - **Simulation**
  - **A Simulation of a system is the operation of a model, which is a representation of that system.**

What is UML 2.0

- The **OMG** specification states:
  - *"The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.*
  - *The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components."*

Diagrams in UML 2.0

- **UML 2.0 defines thirteen types of diagrams, divided into three categories .**

1. **Structure Diagrams**
   - **include the Class Diagram, Object Diagram, Component Diagram, Composite Structure Diagram, Package Diagram, and Deployment Diagram.**
2. **Behavior Diagrams**
   - **include the Use Case Diagram, Activity Diagram, and State Machine Diagram.**

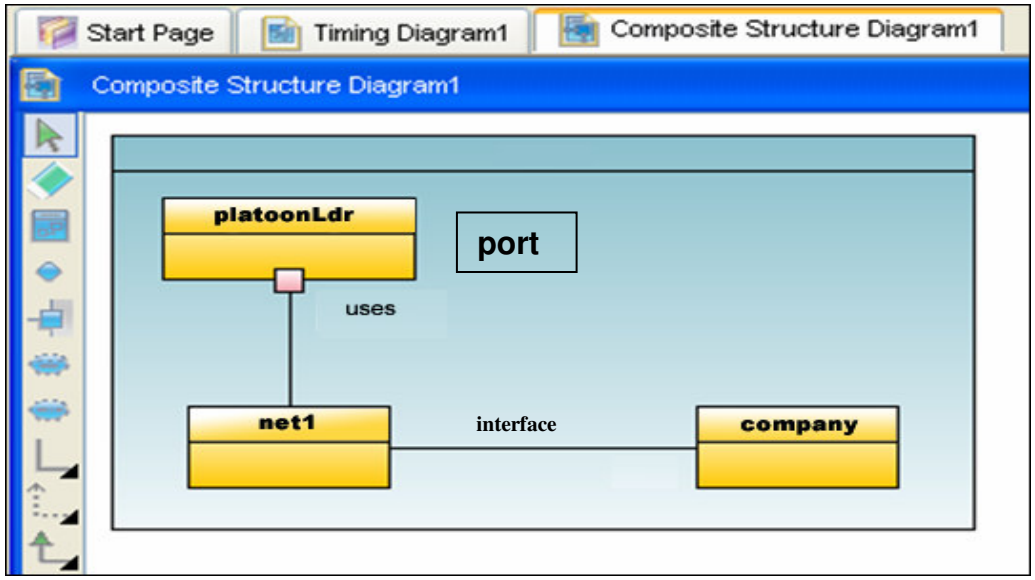
3 **Interaction Diagrams,**

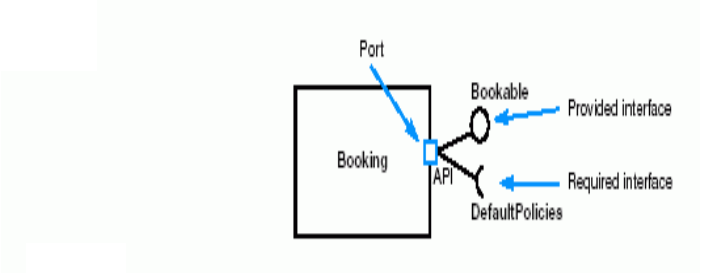
- **all derived from the more general Behavior Diagram, include the Sequence Diagram, Communication Diagram, Timing Diagram, and Interaction Overview Diagram.**

What's new in UML 2.0

- **Composite Structure Diagram (New Type of Class Diagram**
  - **This diagram lets you define how a class is defined by a further structure of classes (objects) and the communication paths between these parts.**

  Part: The concept of parts makes possible the description of the internal structure of a class.
  Port: A *Port* is a named interface on a component, it defines a set of operations and events that are *provided* by a component or that are *required* from its environment.
  Connector: A connector Specifies a link (an instance of association) that enables communication between two or more parts.



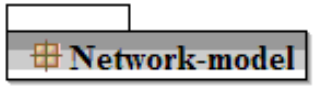


UML 2.0 Diagrams

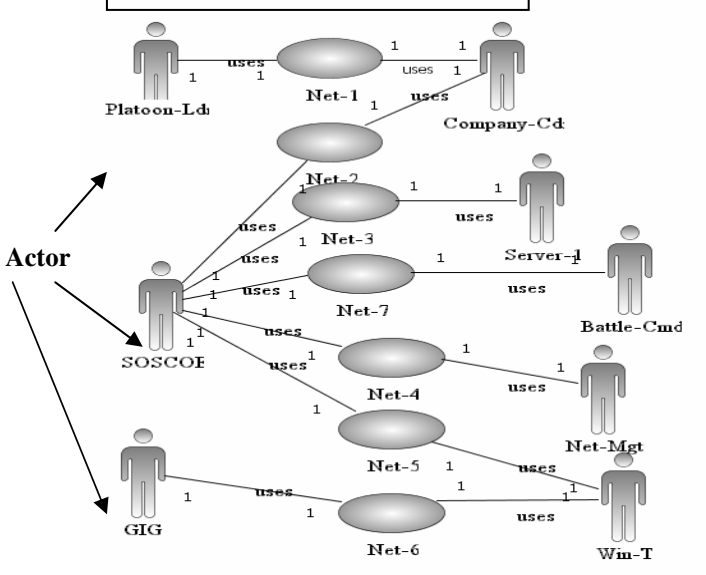
An object is a software bundle of related state and behavior

Package Diagram

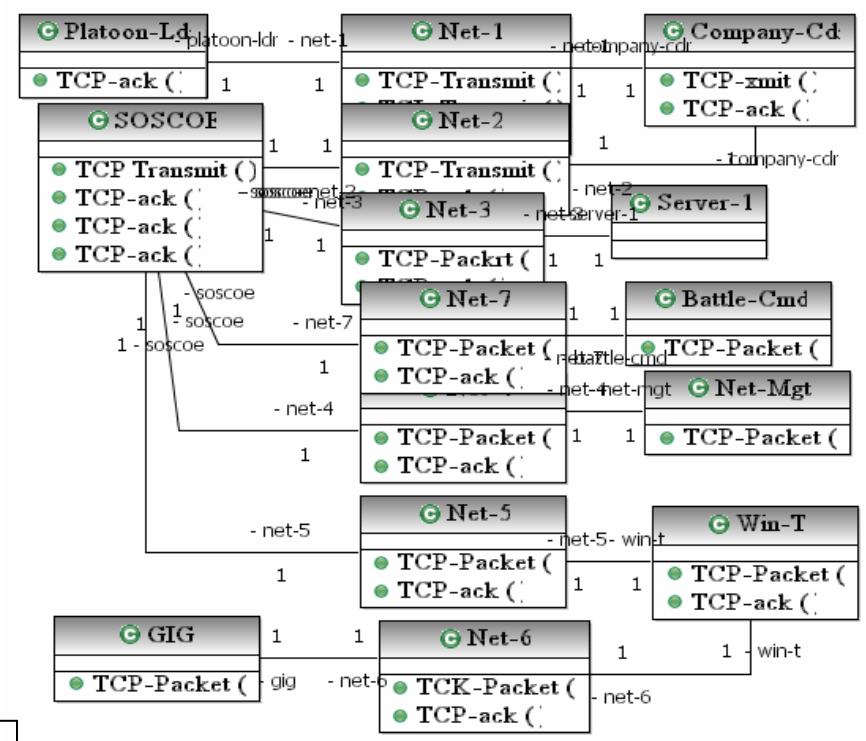
A package is a namespace for organizing classes and interfaces in a logical manner.



Use Case Diagram



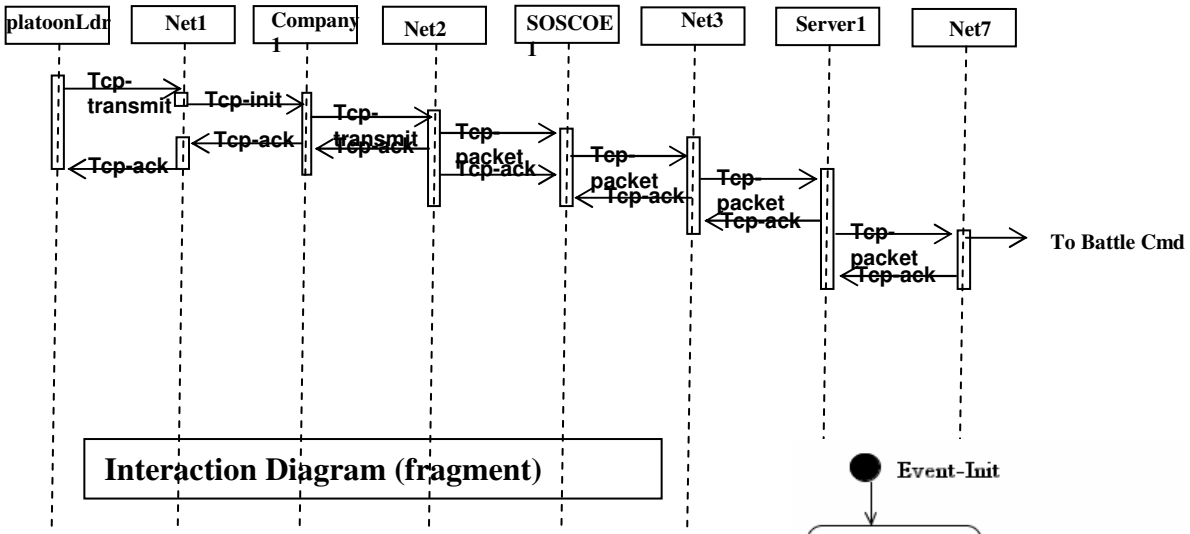
- **A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.**
- **An actor is a person, organization, or external system that plays a role in one or more interactions with your system.**
- **An association exists whenever an actor is involved with an interaction described by a use case.**

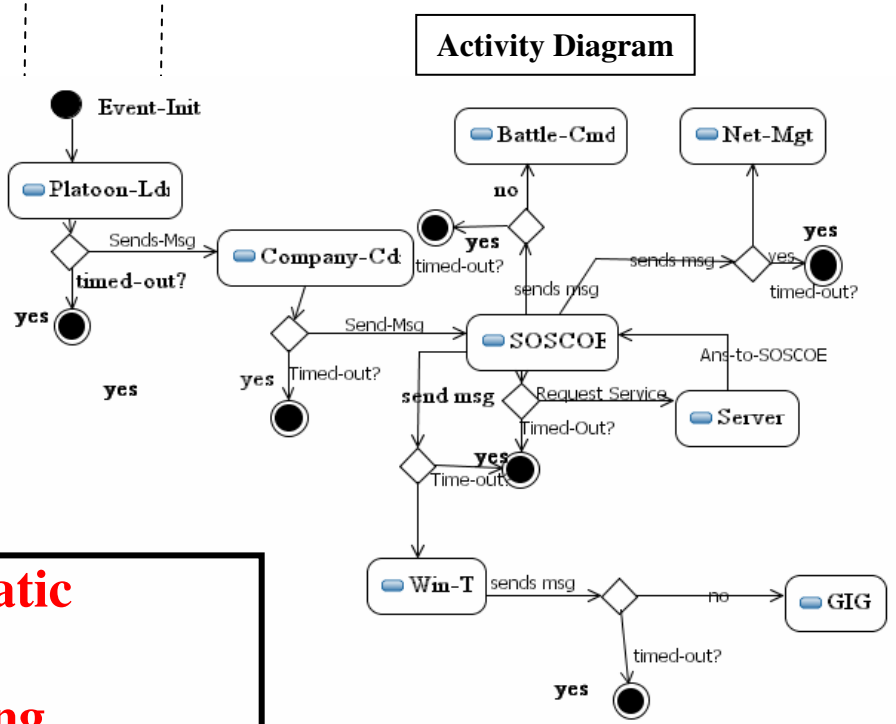


Class Model

A class is a blueprint or prototype from which objects are created.

UML 2.0 Diagram



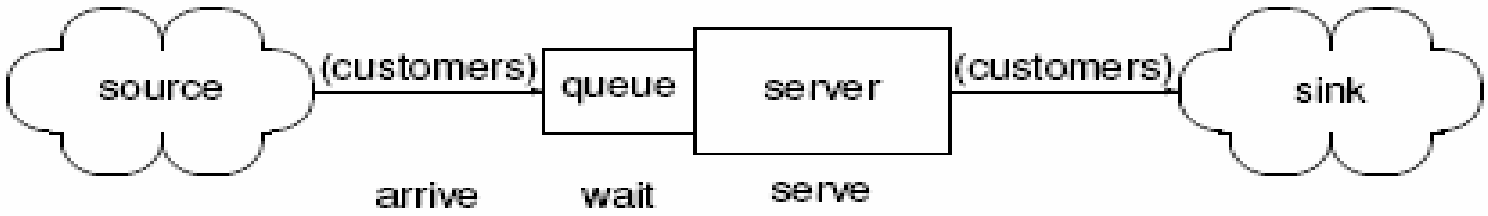


•UML is *NOT* executable. It is a Static Structure

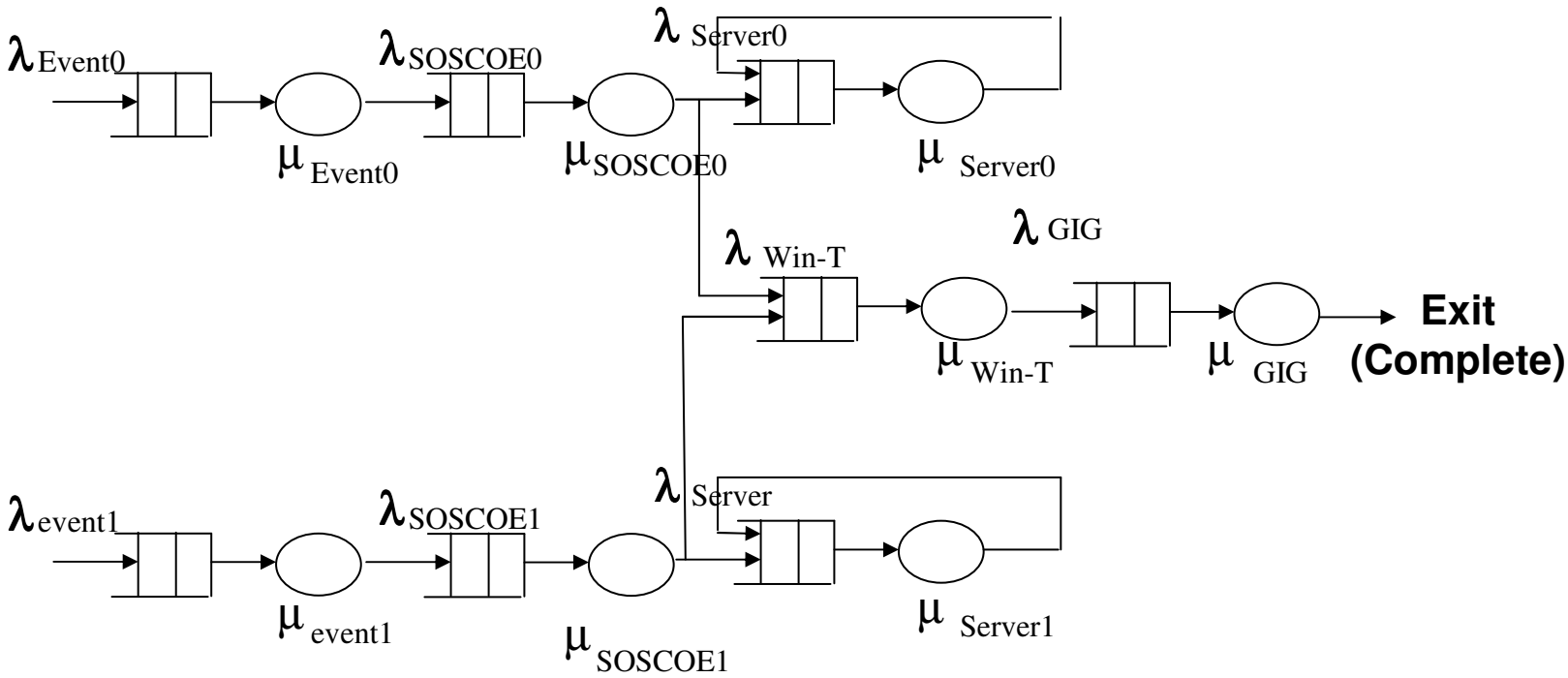
- It does not deal with examining performance issues.

Characteristics of Queuing Models

- **Time: proceeding from event to event;**
- **Object: at least three different kinds of objects, namely Producer, Consumer, and Queue, and, additionally, an agenda which keeps track of the events and schedules them**
- **Stochastic: arrival and serving times are in-deterministic**
- **Dynamic: states of Producers, Consumers, and Queues depend on past states.**
- **Simulation means queuing simulation of queuing models prove to be an adequate modeling paradigm.**



SOSCOE Queuing Model

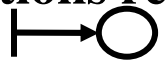


- **Using the aggregate bandwidth of the JTRS, a variable message size and variable background traffic**
- **Execute the model in a open source Petri-Net tool set to examine the behavior**

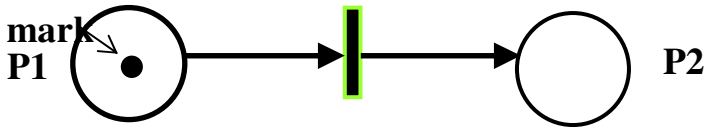
Definition of Petri Net (PN)

• A PN is a n-tuple (P,T,I,O,M) P is a set of places, T is a set of transitions, I input arcs, O output arcs and M marking

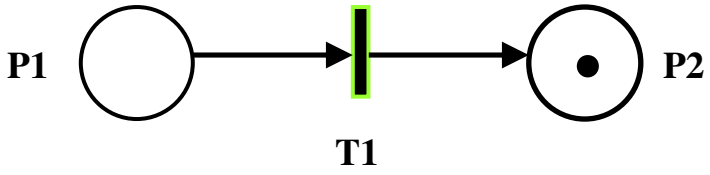
- Input arcs are directed arcs drawn from places to transitions, representing the conditions that need to be satisfied for the event to be activated 

- Output arcs are directed arcs drawn from transitions to places, representing the conditions resulting from the occurrence of the event 

• When input places of a transition have the required number of tokens, the transition is enabled.



• An enabled transition may fire (event happens) removing one token from each input place and depositing one token in each of its output place.



Renew2.1 Petri Net

- **Renew is an open source Java-based high-level Petri Net editor/simulator that provides a flexible modeling approach based on reference nets.**
  - **Reference nets separate the concepts of nets and net instances, thereby allowing object-based modeling on the net level.**
  - **Reference nets allow communication between nets via synchronous channels. This provides a powerful abstraction mechanism.**
- **The simulation engine is capable of true concurrency and supports symmetric multi-processor architectures. Transitions with an extended firing time are executed in the background concurrently with other transitions, if required**

http://www.informatik.uni-hamburg.de/TGI/renew/renew.html

Renew2.1 includes Logging with log4j

- **Log4j is an Open Source logging API developed under the Jakarta Apache project.**
  - **It provides a robust, reliable, fully configurable, easily extendible, and easy to implement framework for logging Java applications for debugging and monitoring purposes.**
  - **Log4j allows developers to insert log statements in their code and configure them externally.**
- **Log4j handles inserting log statements in application code and managing them externally without touching application code, by using external configuration files.**

Petri Net –Distributed SOSCOE

Company 1

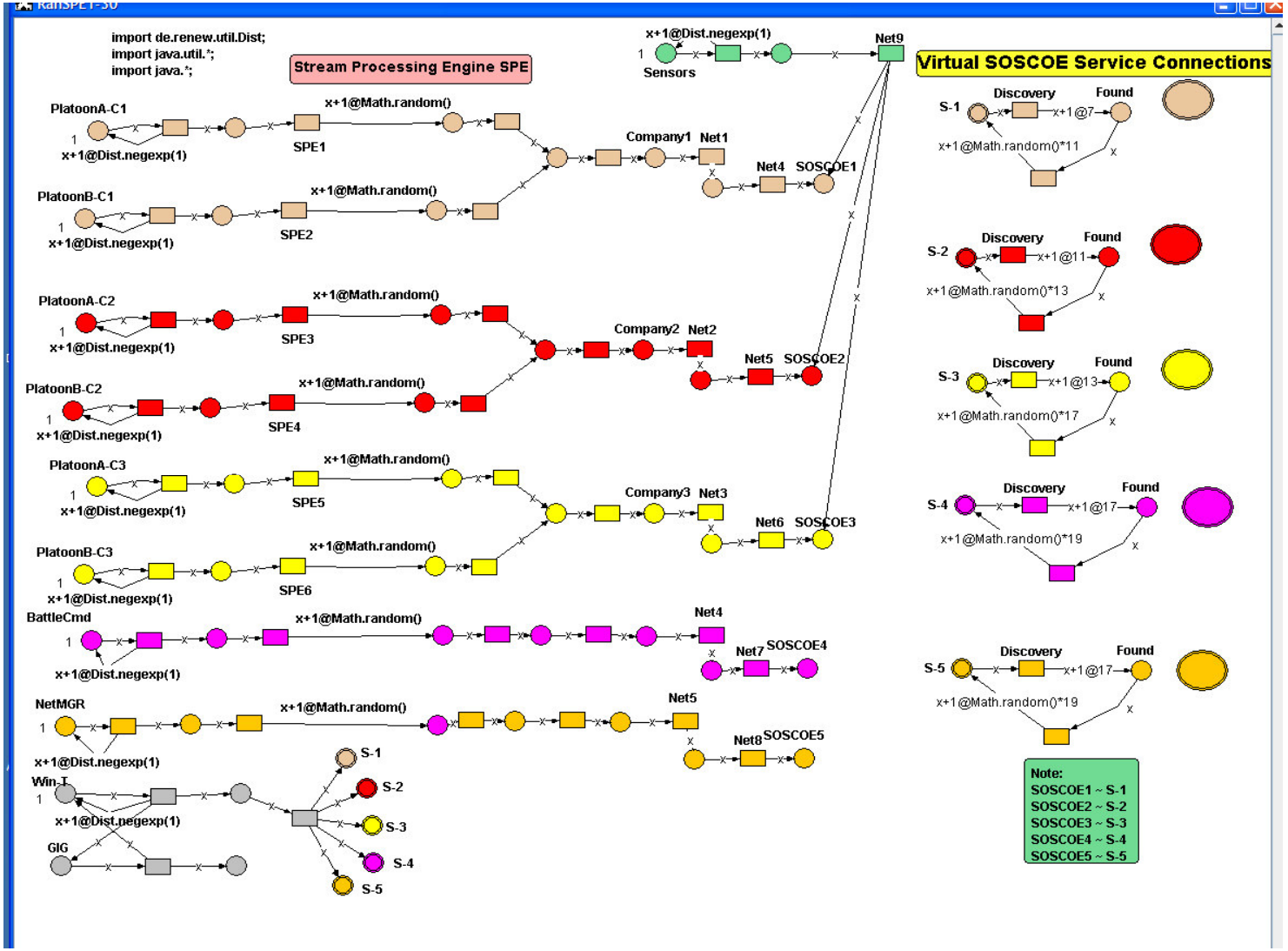
Company 2

Company 3

BattleCmd

NetMgr

Win-t & GIG



Petri Net –Distributed SOSCOE Examining Bottleneck Queues

Company 1

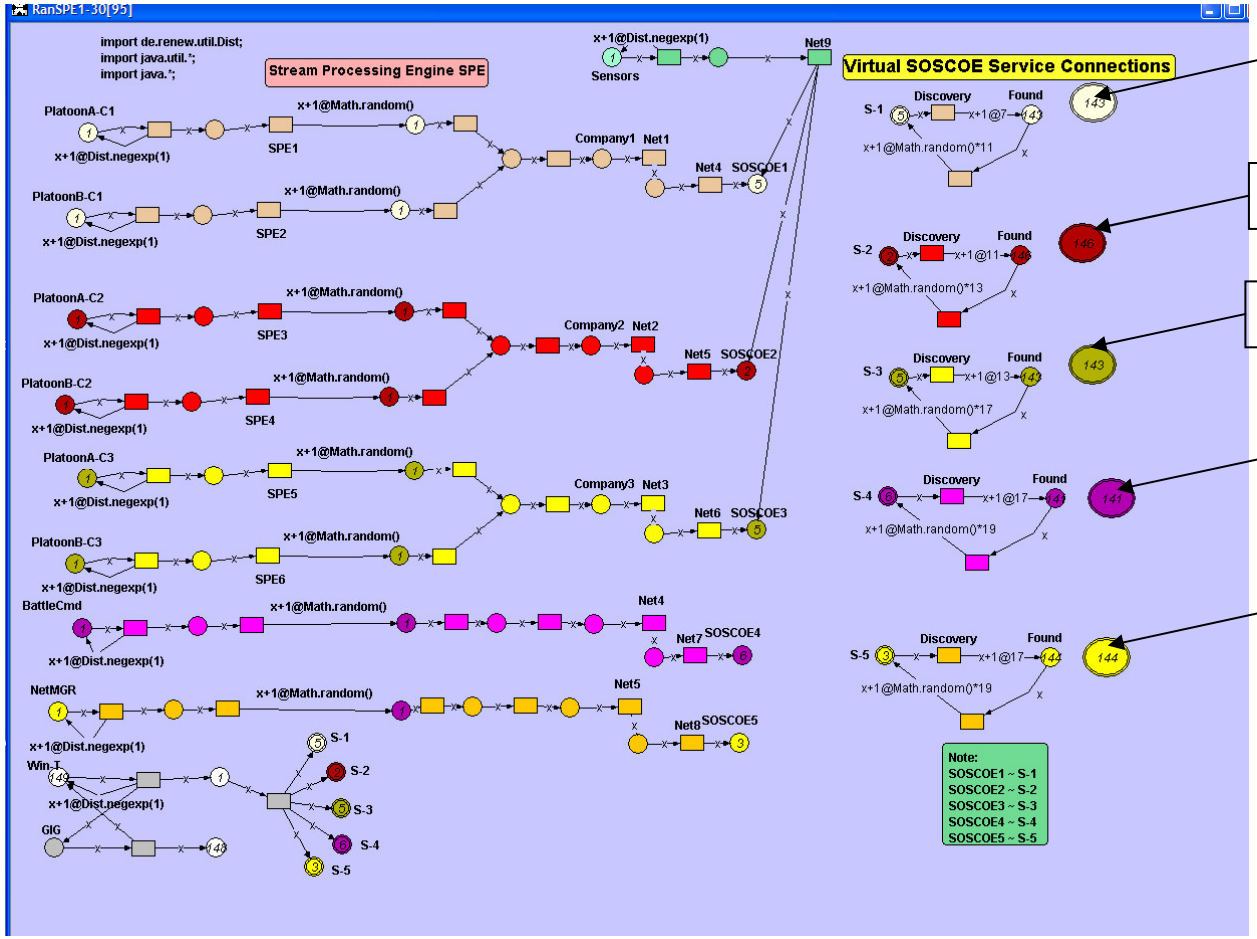
Company 2

Company 3

BattleCmd

NetMgr

Win-t & GIG



145 SOSCOE1

146 SOSCOE 2

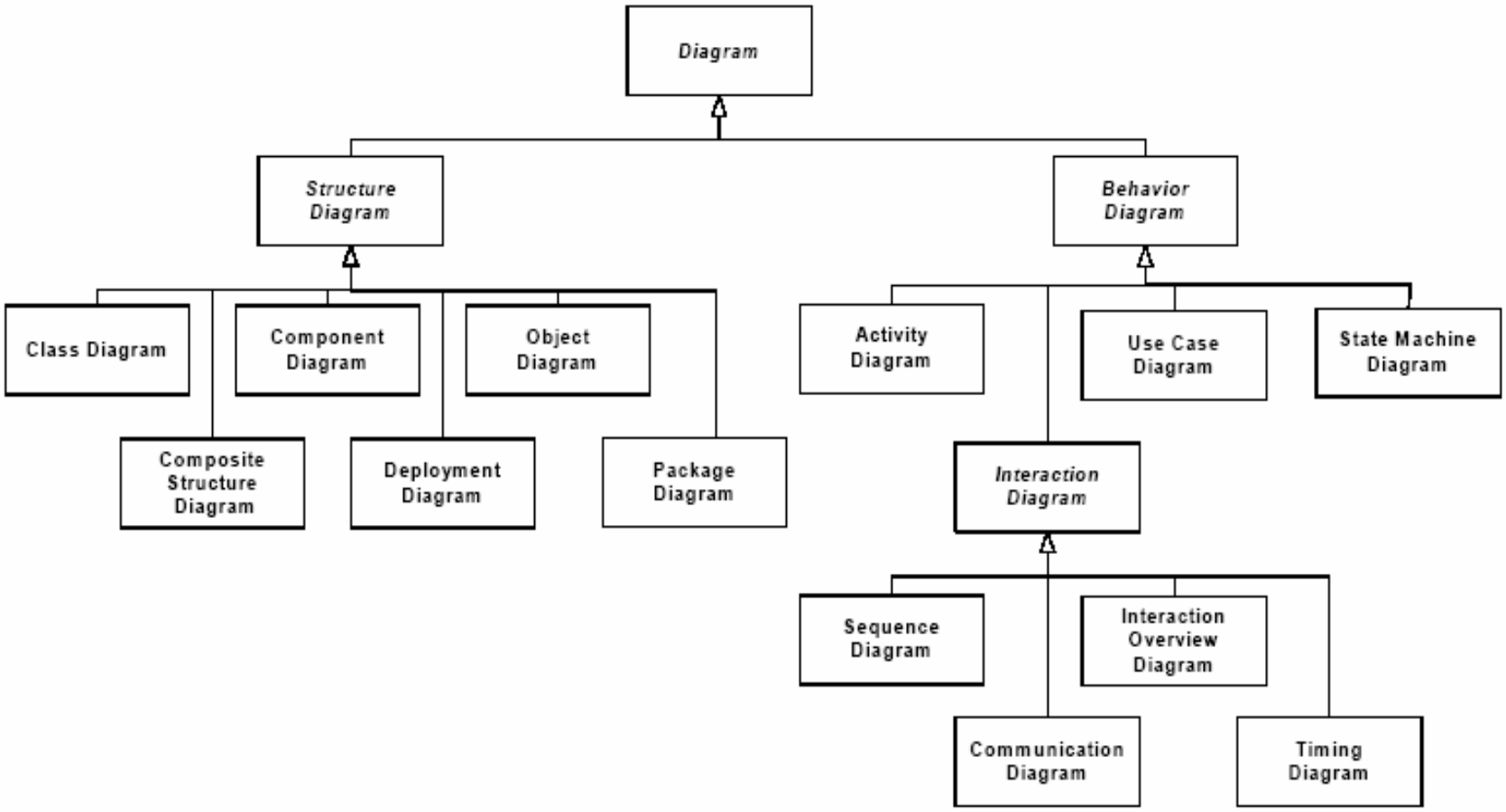
143 SOSCOE 3

141 SOSCOE 4

144 SOSCOE 4

Questions ?

UML 2.0 Overall Structure



What's new in UML 2.0 continued…….

- Some new core constructs such as Parts, Ports and Connectors are introduced.
  - Part: The concept of parts makes possible the description of the internal structure of a class.
  - Port: A *Port* is a named interface on a component, it defines a set of operations and events that are *provided* by a component or that are *required* from its environment.
  - Connector: A connector Specifies a link (an instance of association) that enables communication between two or more parts.

Figure C.47
Ports