Semantic Considerations in OMEGA

Omega Workshop

Grenoble - 17 February 2005

B. Josko, OFFIS

Contents

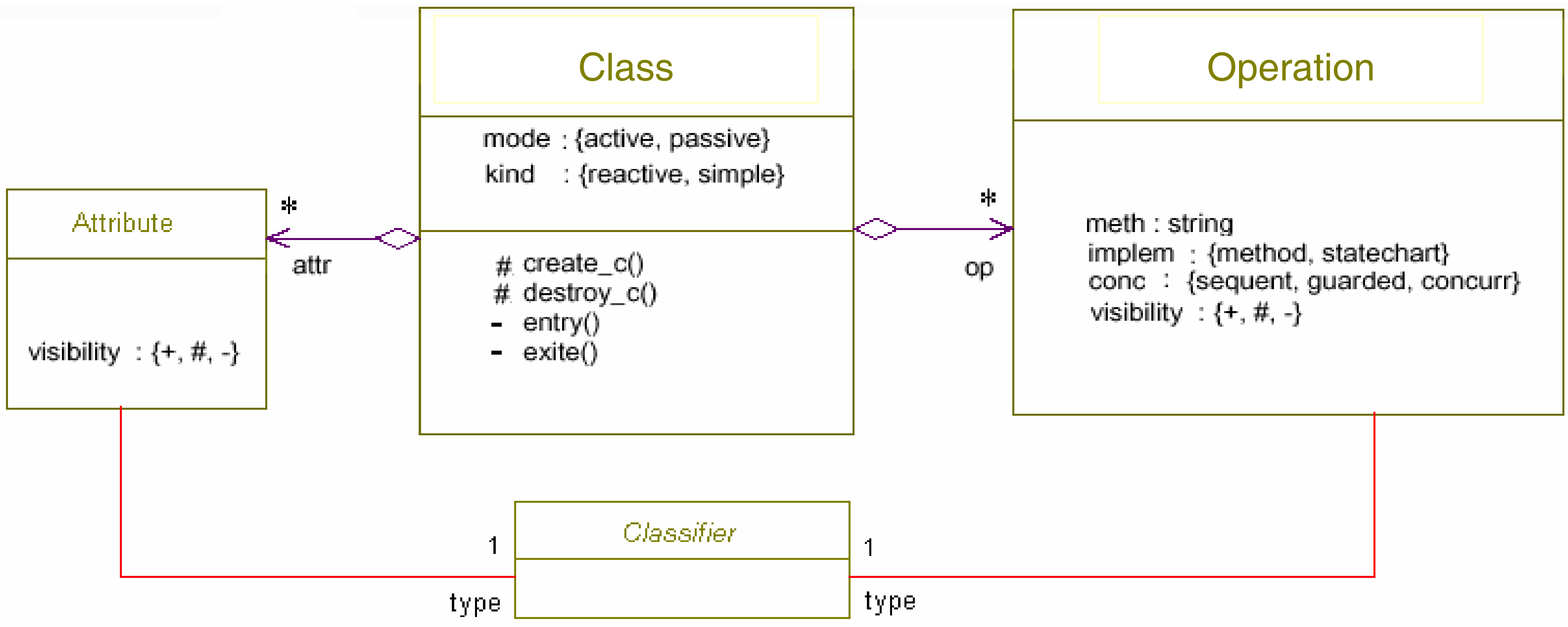
Motivation

- **OMEGA global goal**
  - Provide formal verification techniques for UML models
- **Requirements**

  "… select a sufficiently expressive sublanguage, allowing to capture real-time applications, and specify formal semantics of the chosen part of UML."
    - UML compliance
    - Expressivity for real-time embedded systems
  - Provide formal semantics
    - Basis for formal verification
    - Support effective analysis techniques
- **Approach**
  - Kernel model for untimed behavioural description
  - Time & component extensions of the kernel model
  - Abstract representation of the Omega semantics with variation points for compositional verification

UML coverage

Ω-subset: Class Constituents



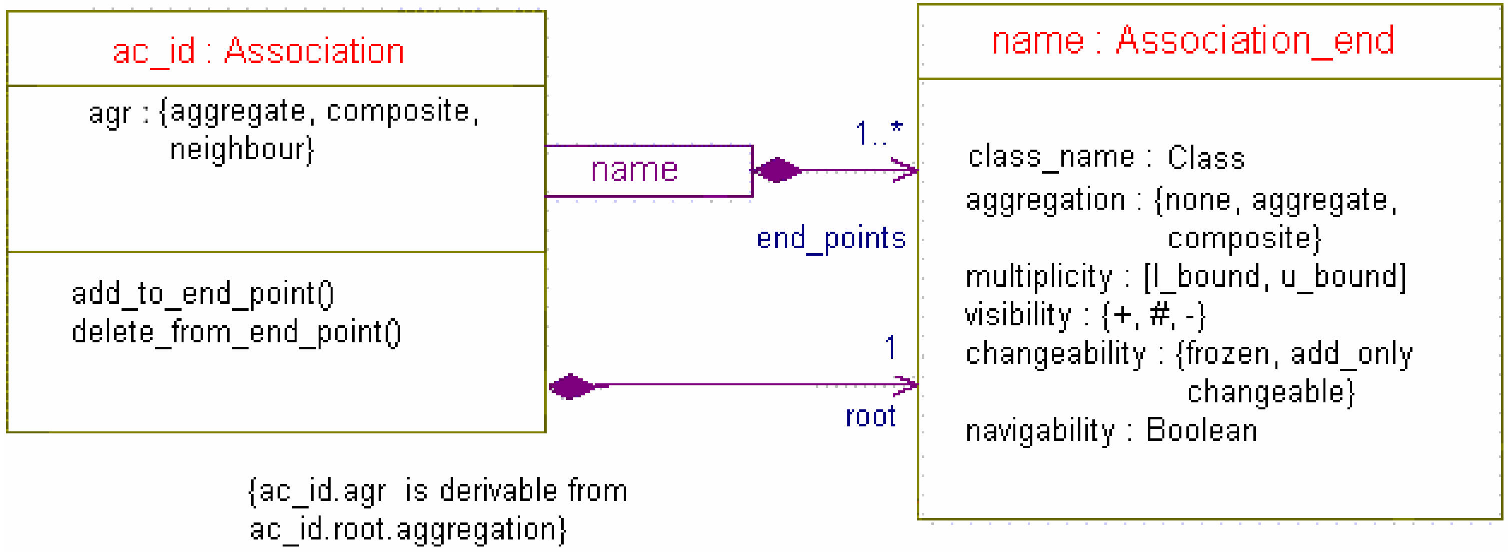
Class interface

- All public attributes and operations
- All signal receptions
- All operation calls and signals emitted to other objects

Ω-subset: Class Relations

- *Generalization* **relation (multiple inheritance) with**
  - Overriding operations and attributes (leading to polymorphism)
  - Specialisation of signals
- *Association* **relation (with different multiplicity) of the following three types:**
  - Composition (a.k.a. strong aggregation)
  - Aggregation (a.k.a. weak aggregation)
  - Neighbour (is derived from the former associations)

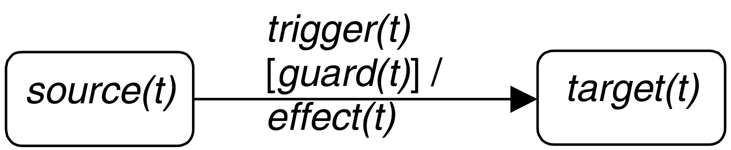
Association Definition



Kinds of multiplicity:

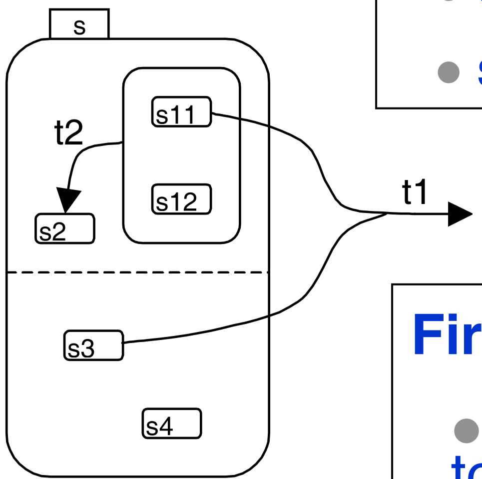
- $[n, n] = n \in \mathbf{N}$
- $[0, n], [m, n]\ \ m < n \in \mathbf{N}$
- $[0, *] = * \notin \mathbf{N}$, $[m, *]$
- $[m, m+1] = \{m, m+1\}$

Ω-subset: State Machines



2 Kinds of composite states:

- concurrent (AND-states)
- sequential (OR-states)

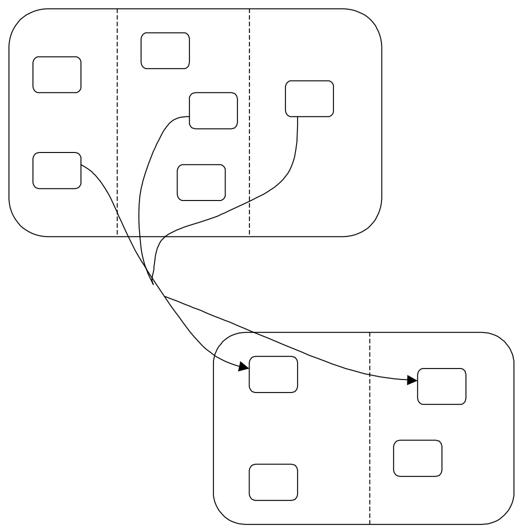


Firing enabled transitions:

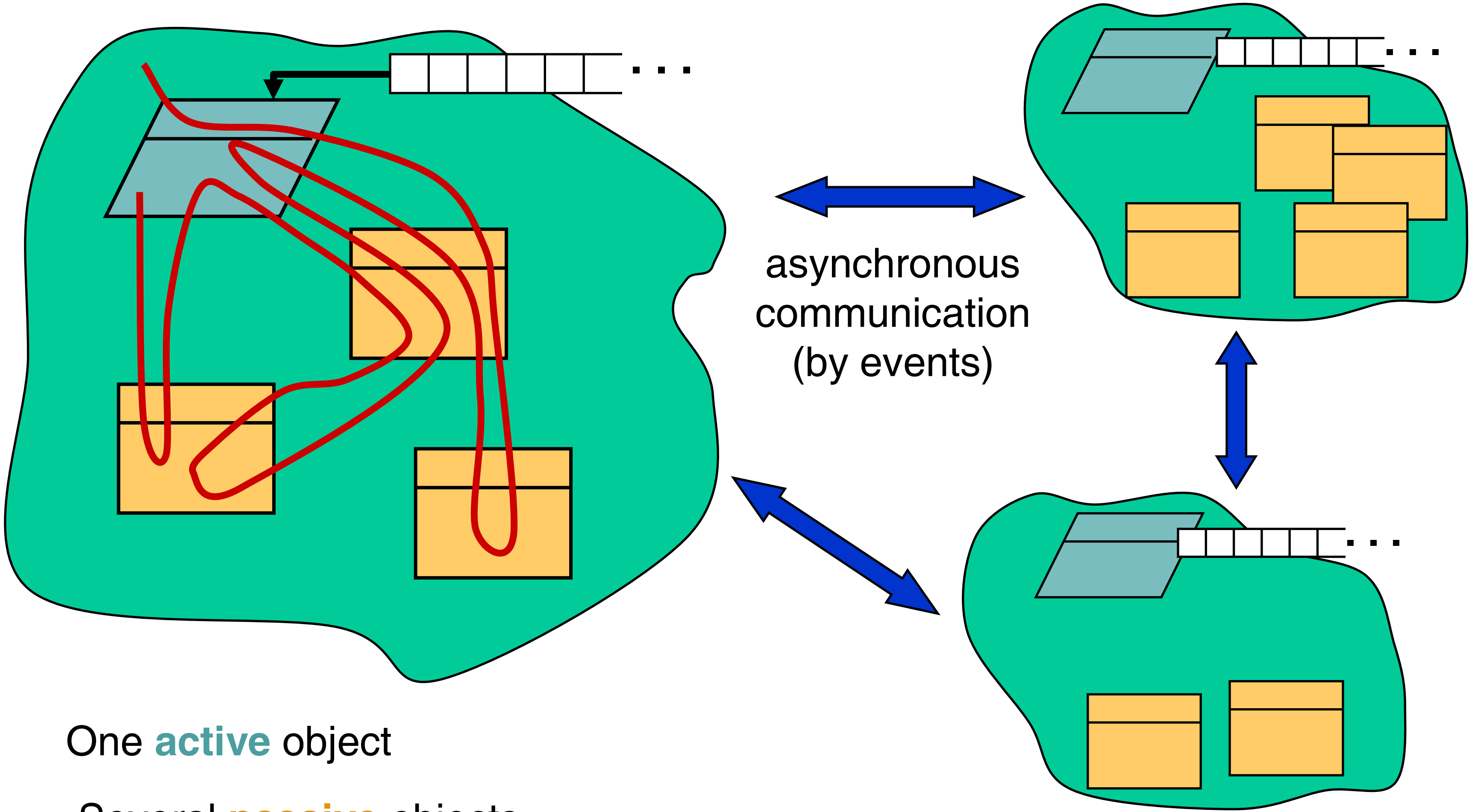
- priority from the innermost to the outermost
- non-deterministic choice between transitions with the same priority

Pseudo-states:

- history connectors
- joint and fork connectors are replaced by considering transitions with multiple sources and targets



Activity Groups



One **active** object

Several **passive** objects

One **thread** of control

Semantic Coverage

- **Object-orientation**
  object creation/destruction (with different object multiplicity), change of communication topology, inheritance and polymorphism, “multithreading”.
- **Concurrency**
  - Between activity groups, where each activity group is sequential
  - Between concurrent regions in a state machine
- **Communication**
  - Synchronous via signal events (with parameters)
  - Asynchronous via operation calls (methods or call events)
  - Access to public attributes
- **Sources of Dynamic**
  - Object creation/destruction, polymorphic operations, association changes
  - Non-deterministic choice, e.g. in transition firing or the order of the executions in concurrent regions of a state machine

Formal Semantics

UML Ω-Model

A *UML Ω-model* is a tuple

M = (C, A, Sig, c0, Assoc, Gen, sm)

- **C set of classes with interface definitions**
- **A $\subset$ C set of actors, specifies external behavior**
- **the root class *c0* is maximal under aggregation**
- **A set Sig of signals**
- **Assoc Association relations**
  - the composition relation defines a DAG
- **Gen Generalisation relations**
- **sm associates statemachines to all classes**
- **inter-object communications are compliant to the class interfaces**

Symbolic Transition System

$S = (V , \Theta, \rho)$

- V — typed set of variables
- Θ — initial condition on variables
- ρ — transition relation on variable valuations

traces(S)

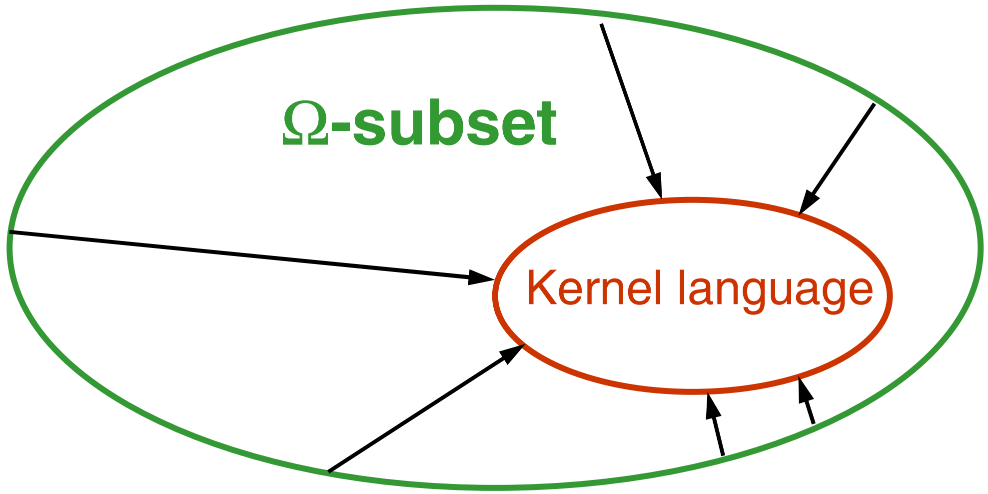
set of infinite sequences of valuations of variables satisfying:

- first valuation matches Θ
- successor valuations satisfy ρ

Semantics of the Ω subset in terms of KL

Preprocessing

- Introduction of implicit attributes and operations
- Compiling away generalisation
- Replacing complex navigation expressions
- Compiling away composition
- Inlining methods of primitive operations into state machines
- Flattening statecharts

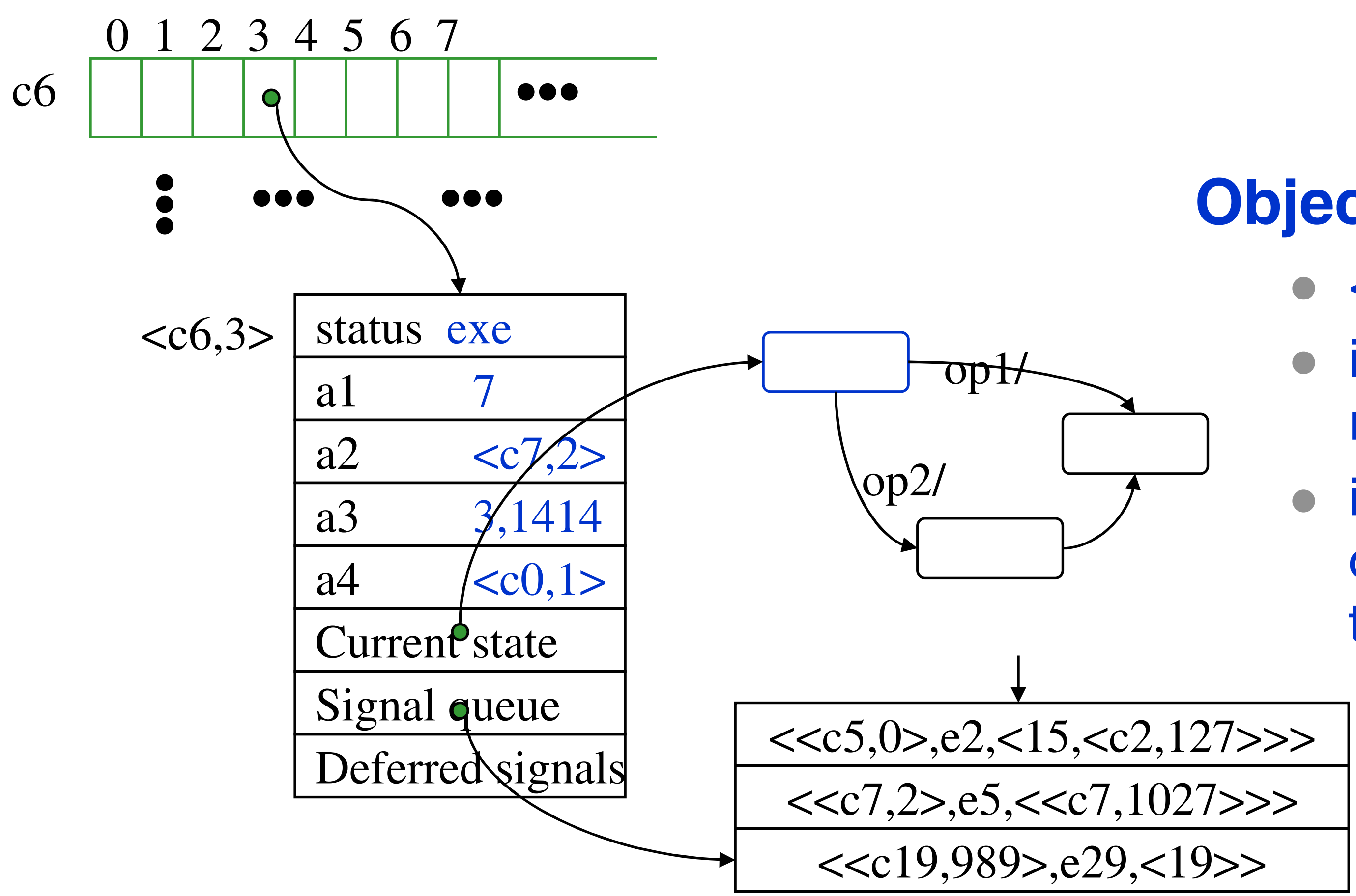


Semantics of Models

- **Given UML model M = *(C, A, Sig, c0, Assoc, Gen, sm)***
- **Associate to M a symbolic state transition diagram**
  $S_M = (V_M , \Theta_M, \rho_M)$
- **V_M is composed of**
  - sys_conf (System Configuration) contains
    - Set of objects
    - For every object
      - Values of attributes
      - Statemachine configurations
    - For active objects
      - Event queue
  - PRT (Pending Request Table)
    - Information on synchronous calls (sender, receiver, return value, status)
- Θ_M defines initial configuration
  - One object of root class with its initial values
- ρ_M the transition relation covers:
  - Effects of SM transitions
  - Object creation / destruction
  - Event disgarding

System Configuration

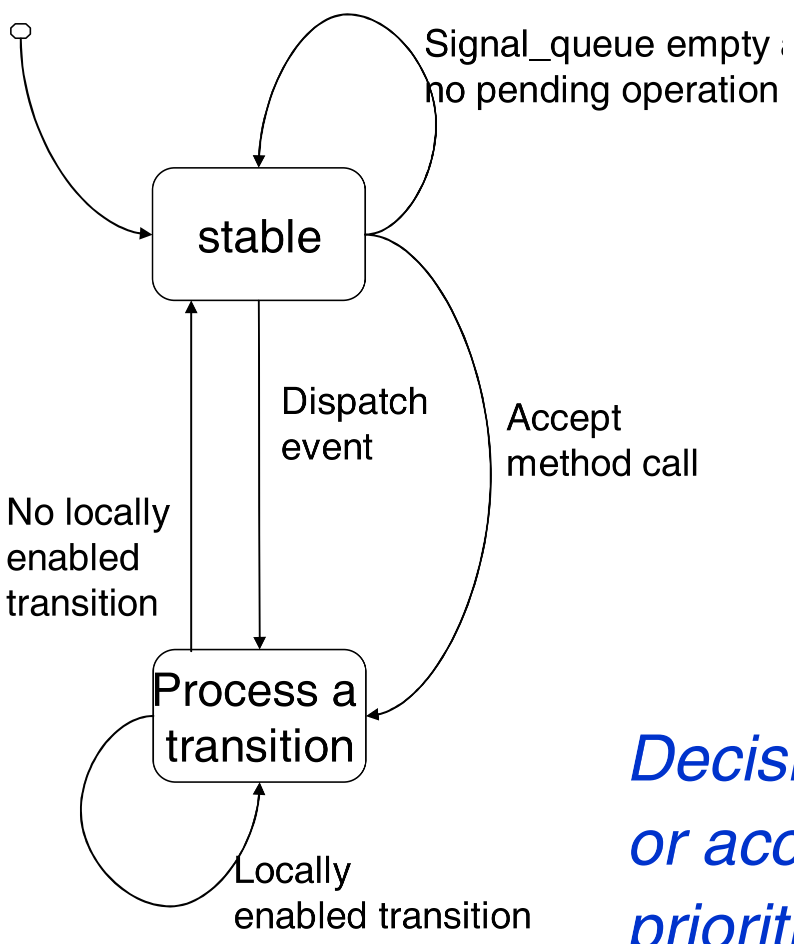
sys_conf : C → N → Valuation of object system variables



Object Identities

- <c,i> $\in$ O_id = C $\times$ N
- in formal semantics: no re-use of object id´s
- in implementation: object_id´s are pointers to memory

Execution Scheme



Run-to-Completion Step:

- at the level of one object
- at the level of activity group

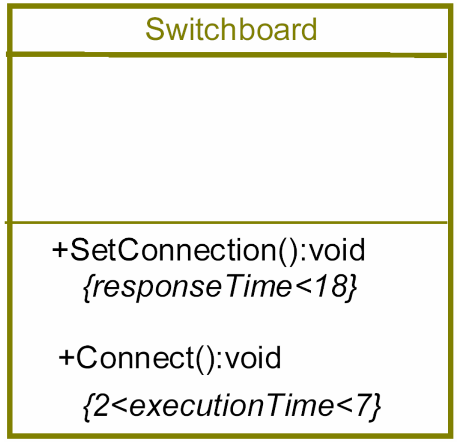
Composition:

- between objects within one activity group
- between activity groups

Decision whether to dispatch event or accept method call based on priorities (can be non-deterministic)

Adding Real Time

The timing framework



- **A specialisation of the UML SPT profile**
  - An extended subset of the standard profile anticipating on *UML 2.0*
  - Adds explicit semantics to the used concepts
  - All concepts accessible at type level

- **Global time**
  - time-related primitive types Time, Duration
- **Imperative constructs (UML 2.0) : *time dependent behavior***
  - time is external (not constraint by imperative constructs)
  - mechanisms for measuring durations: timers, clocks
  - Usage: part of action language

The timing framework (2)

- **Declarative constructs : *timed events and constraints***
  - Express constraints on time progress
  - Timed events: history of occurrences of identified state changes
    - *Sending, receiving, consuming* a signal
    - *Executing* an action / a state machine transition
    - *...*
  - Constraints on duration between event occurrences
    - **Basic** time constraints (used as axioms)
    - **Derived** time constraints (requirements to be verified)
  - Usages
    - Local constraints of classes and global constraints
    - Event matching mechanism used in specialized <<observer>> classes

To describe the semantics formally: State Transition Systems are extended to **Timed Automata**

Conclusion

Achievements

- **Definition of the Ω subset of UML**
  - Rich subset of UML suitable for real time embedded systems
- **Definition of the kernel UML model**
  - A simple and expressive operational subset of UML
  - Formal semantics for the kernel language
- **Time extension for the kernel model**
  - Simple and expressive time concepts and mechanisms
  - Semantics for the time notions
- **Component model**
  - Presentation in the kernel model of both internal and external view of component
  - Inter-component coordination mechanisms: small prototype implementations
- **Abstract semantics**
  - Abstracting from the tool implementation details
  - Allow compositional reasoning

Comparison to other approaches

- **UML standard**
  - *Incomplete*, we fully define the semantics of the selected subset
- **Semantics implemented in UML CASE tools**
  - Tools: semantics choices rarely made explicit and deviations from standard are frequent
  - OMEGA: explicit semantics and close to UML standard
- **ACCORD UML (S. Gerard / F. Terrier)**
  - Little focus on semantics, mainly a methodology on how getting an implementation
- **pUML UML formal semantics in Z**
  - Addresses only the static part of UML
  - Does not mention real-time
- **UML semantics in ASM (I. Ober)**
  - Does not treat statemachines
  - Does not mention real-time

Thank you for your attention