State/Behavior and the CIM Model

9 October 2007

Karl Schopmeyer

Chair, State & Behavior WG

Version 1.1, 22 October 2007

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change. The Standard Specifications remain the normative reference for all information.
- For additional information, see the Distributed Management Task Force (DMTF) Web site.
- http://www.dmtf.org/standards

The DMTF was formed to lead the development, adoption and unification of management standards and initiatives for desktop, enterprise and internet environments

Agenda

- What do we mean by Behavior and State ?
- Growth of behavior integration in modeling and CIM
- A CIMState Extension to the CIM concepts

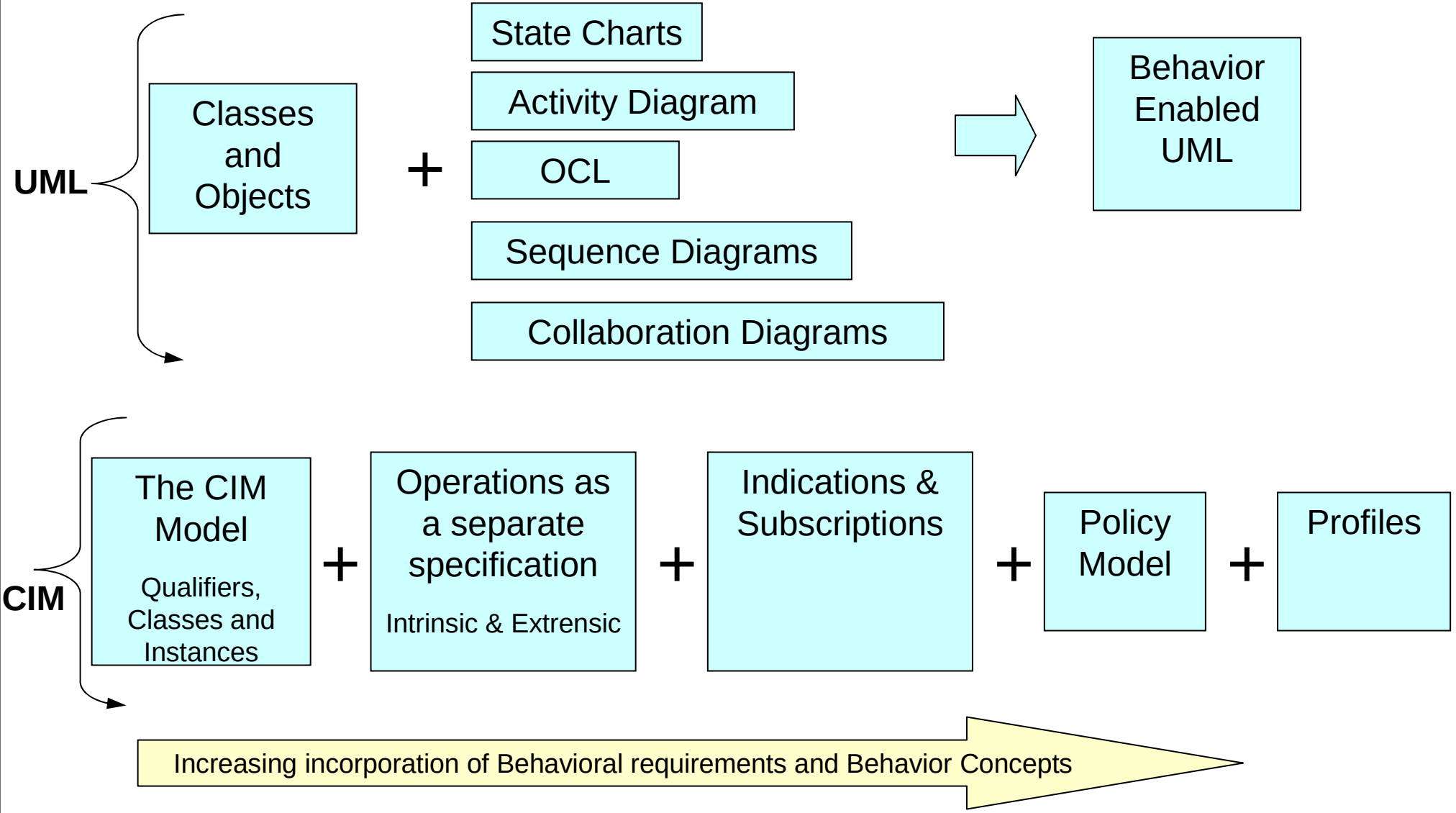
What is Behavior?

- Important question because the conceptions of state, state machines, etc. vary widely
- Behavior means different things to different people
- Definitions
  - Behavior - A behaviour of an object is a collection of *actions* that the object may take part in, together with the set of constraints on when those actions can occur. The object model does not constrain the form or nature of object behaviour. The actions can be interactions of the object with its environment or internal actions of the object
  - State - The state of an object is the condition of the object at a given instant that determines the potential future sequences of actions that object may be involved in. At the same time, actions bring about state changes and, hence, the current state of an object is partly determined by its past behaviour.

Behavior and UML

- Behavior is a core concept in UML
  - Structural Diagrams (class, Object, Composite Structure, Component, Deployment, …)
  - Behavioral Diagrams(Use case, Activity, Interaction, State Machine, Protocol State Machine, OCL)
    - The behavioral diagrams define how the UML modeled resources interact and how they execute their capabilities.

The Growth of Behavioral Concepts in Modeling & CIM



Growth of Behavior in CIM

- The original developers of CIM tried to minimize behavioral characteristics in the development of the CIM Information Model
- A few early CIM models introduced behavior (Ex. Application Model state classes)
- Policy Model was direct introduction of model behavior through adding model components (rules define conditions -> actions)
- Basic state introduced to the CIM Model with properties in selected classes
  - CIM_EnabledLogicalElement: EnabledState and RequestedState
  - CIM_ManagedSystemElement: OperationalStatus
- The indication/subscription model added significant behavior because specific behavior was required relating subscriptions to indications. However, only the model components were formally defined initially
- Large scale users recognized the limitations and defined behavior through profiles
  - Define the usage of a group of classes (services provided)
  - Define/extend the classes and instances involved
  - Define the interaction between the Classes/instances of the classes
  - Define the required behavior of the resource in terms of the classes/Instances
  - Largely free text today
- OCL introduced as a possible behavior definition tool (OCL Qualifier)

State and State Machines

- State and State Machine Abstractions are a useful tool to define object behavior
- Historically state used in hardware and protocol design
- Adopted by OMG as core behavior definition tool
- Uneven use in software development
- State concepts are part of several management technologies
  - ISO management (Common state machine mechanism)
    - Single state model for operational, Usage, Administrative state
  - JSR 77 – J2EE management
    - Common state model for J2EE Management objects
  - Goal driven or desired state management solutions
    - Management automation based on state as condition and actions
    - Requires concepts of state, managed component hiearchy, state hiearchy
    - A number of solutions use this concept but it was never really standardized
    - User sets top level desired state. Management environment attempts to move all elements in the defined hierarchy so that the result state matches the desired state.
    - Often used to change operational state in complex system systems requiring start/stop of multiple subsystems to change overall state.
  - CIM
    - Incorporates Operational State into model today

Issues of State as behavior in CIM

- State and behavior in defining models and profiles
  - Using state to define behavior of CIM Objects in response to external input
  - This is a behavioral mechanism useful in the definition of models
  - Using actions to define the operations one object can execute on another object as part of the behavior of the object.
    - Example. Object A sets a property to a defined state in object b
    - This can be integrated into a state machine so that state changes cause actions.
- Define and use a common managed element state management mechanism. Provide a common state model for CIM Objects with concepts like:
  - Predefined states (ex. Enabled, Disabled, Shut Down, No Change, Offline, Test, Deferred, Quiesce, Reboot, Reset
  - Provide mechanisms so that hierarchies of state change can be defined. Allows rolling up state changes from one element to a higher level element.
- Clients want to see state as status

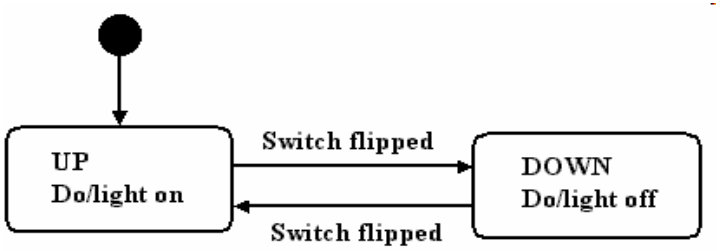
Objectives of the Behavior & State Working Group

- Today CIM is still today largely an information model
- Defining behavior within CIM is difficult
  - Behavior of CIM objects in relation to external inputs
  - Between objects (actions from one object to another)
  - Model today incorporates State variables but without means to define behavior of the objects themselves
  - Much of management implementation specs(ex SMIS) is definition of behavior of managed objects and between managed objects
  - Concepts like starting and stopping in reality are object interdependent and represent state
  - Work should be consistent with UML concepts
- Objectives
  - Define mechanisms that would enhance behavior of CIM objects and between objects to be defined.
  - Clarify the use of State in the CIM models
  - Extend CIM to allow programmatic use of state
  - Extend use of management of state

What is Finite State machine?

- Described By:
  - An initial state or record of something stored someplace
  - A set of possible input events
  - A set of new states that may result from the input
  - A set of possible actions or output events that result from a new state

A state machine is any device that stores the status of something at a given time and can operate on input to change the status and/or cause an action or output to take place for any given change.



FSMs model how objects respond to external events

A finite state machine can be used both as a development tool for approaching and solving problems and as a formal way of describing the solution for later developers and system maintainers.

FINITE STATE MACHINES

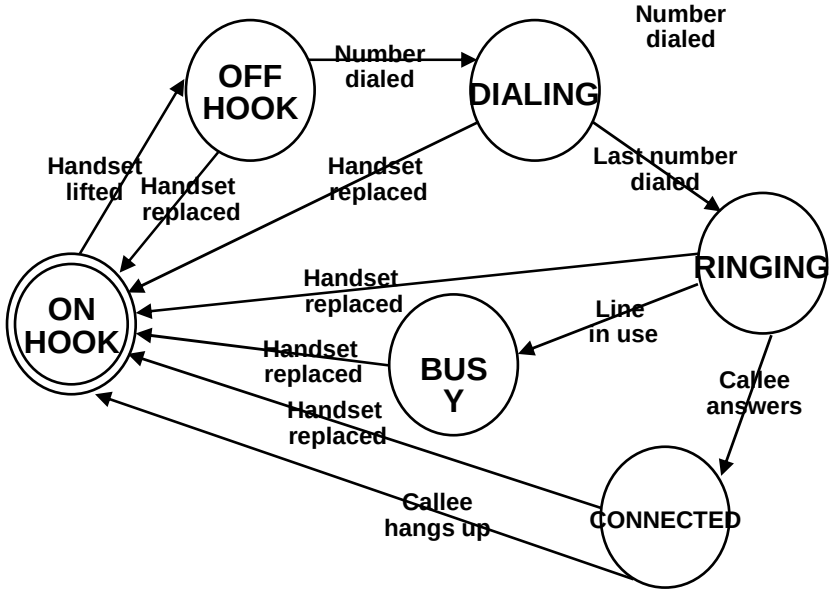
- Many systems have the characteristic of "memory". That is, past behavior can influence future behavior. For example, a pay phone must remember how much money has been inserted to know how much service to provide
- Typical technique for modeling these situations is with a *finite state machine*. The states serve as the memory
- Abstract machines with memory are called sequential machines to distinguish them from "memoryless" combinatorial machines like those represented by decision tables

- State transition Diagrams
  - One mechanism for representing finite state machines (FSMs) is the state transition diagram
  - A state is denoted by a rectangle containing the state's name
  - The machine is always in exactly one state. When the system obtains a new input, the machine moves to another state and performs an action, which may be used to produce an output
  - States are connected by labeled directed arcs denoting the transitions among the states
  - The labels on the arcs consist of two parts. The first is the input/event/stimulus provoking the transition. The second is the action/response performed. The two parts are separated by a short horizontal line

State Machine Advantages in Behavior Definition

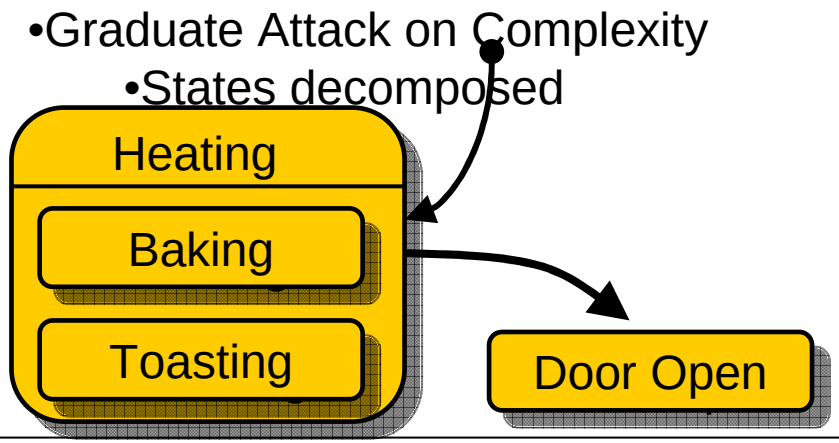
- Concise form for defining reaction of object to input
- Capable of extending reaction to inter-object actions (action of one object on another object)
- Clearly bounded description of change to input
- Clear mathematical form that simplifies design
- Precise form of definition that is usable for code generation

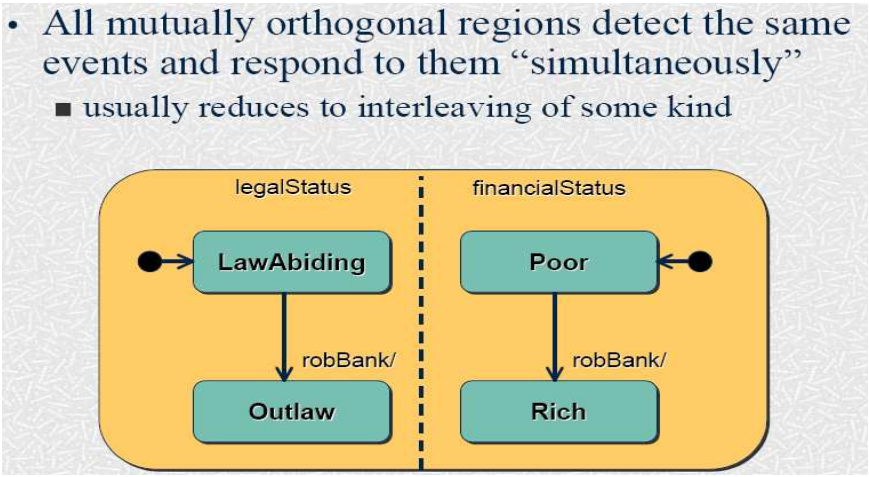
  - Example
    - A telephone is a device that at any given time is in one of several states. For example, the state can be busy, ringing, dialing, in use, or on hook.
    - User actions cause the current state to change. For example, a user picking up a ringing telephone causes the current state to change from ringing to in use.
    - State transition diagrams can be used to model software with user interfaces.



UML STATE CHARTS

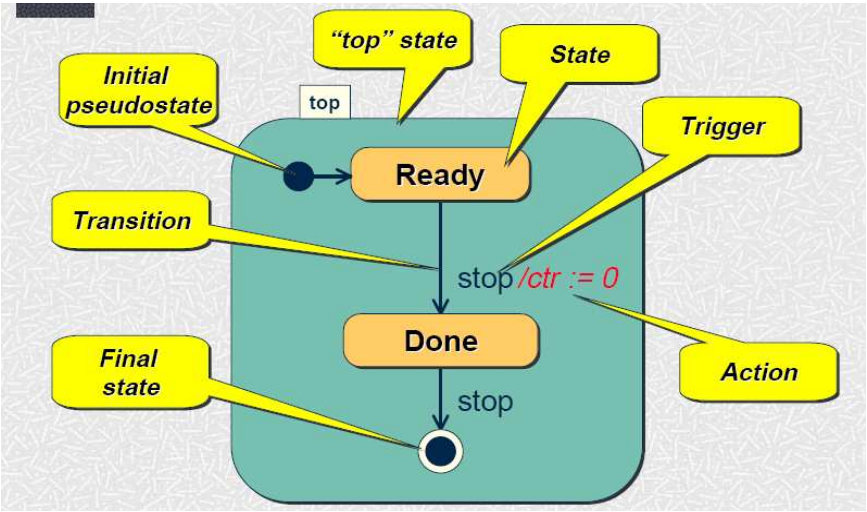
- One problem with finite state machines was the explosion in the number of states that occurs when several independent activities are going on simultaneously
- Traditional finite state machines provide no mechanisms for abstraction; the grouping together of related states that are relatively independent from the other states in a system
- David Harel developed an extension to finite state machines called *state charts* to overcome these difficulties
  - Core additional concepts were **hierarchical states** and concurrent states (**orthogonal regions**)





State and UML – The StateChart

- State Charts are part of UML
- One of The UML behavioral elements
  - Shows how parts of a UML model changes over time
  - Objects change state in response to events and time
- The UML state diagram captures state changes
- State is defined for a single UML object, and sequences of messages between objects.



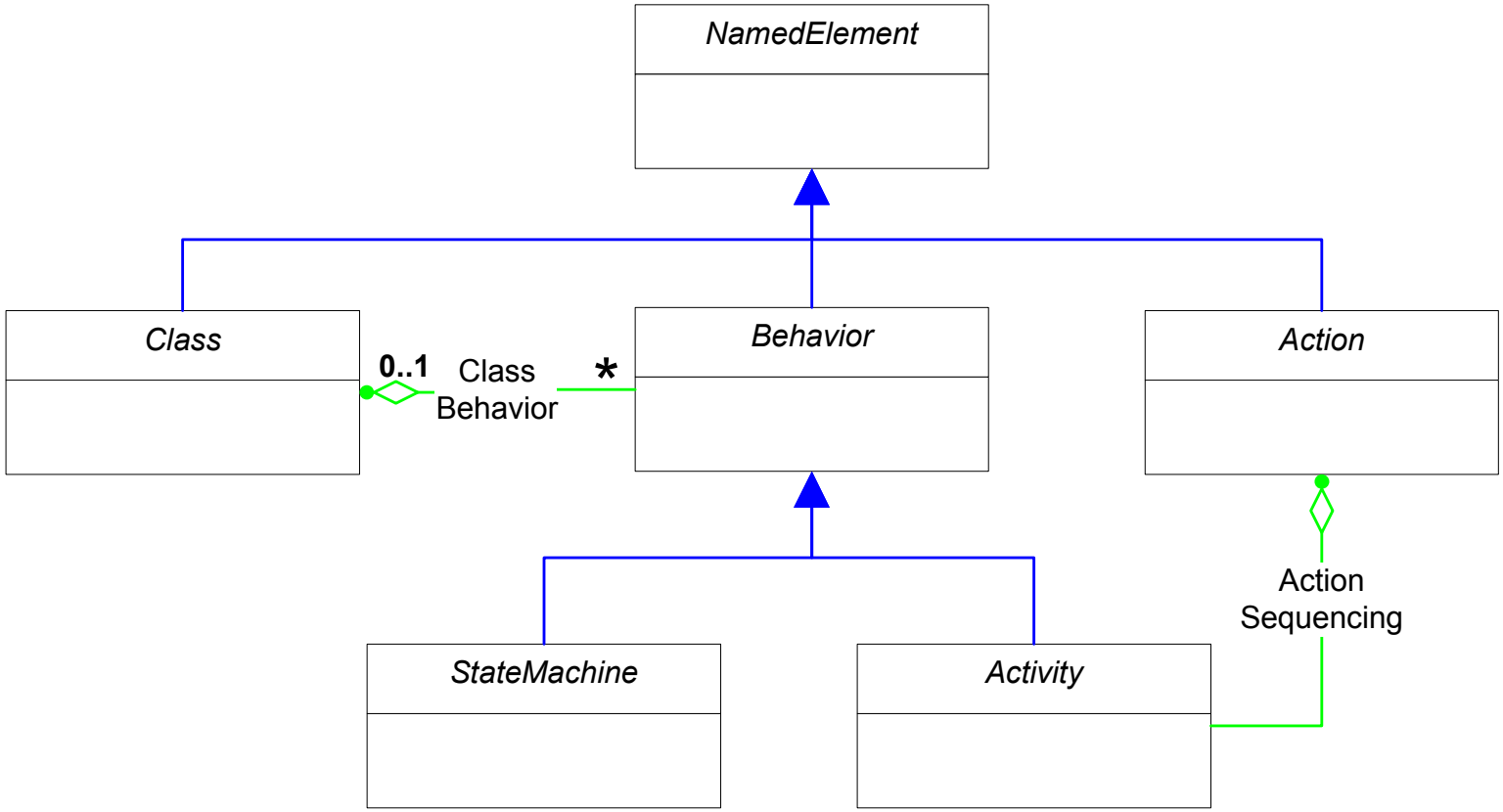
- Hierarchical State Model
  - Hierarchical States
  - Hierarchical State Transitions
- Based on event processing architecture
- Features
  - Guards
  - Entry and exit actions
  - Orthogonal Regions - orthogonal regions detect the same events and respond to them "simultaneously"

Using UML State Model for CIMState

- Issues
  - Complexity of UML StateCharts
    - Hierarchical states, orthogonal regions, etc. create a complex development/modeling environment
  - UML Diagram based, no text language (ex. MOF ⇔ UML)
  - Concepts of Actions and Constraints which do not translate cleanly to CIM
    - CIM has no single condition and action language
- Solution
  - Extend the CIM metamodel to incorporate behavior concepts
  - Define simplified, text-based abstract notation for State Transitions
  - Offer alternatives for Actions/Conditions

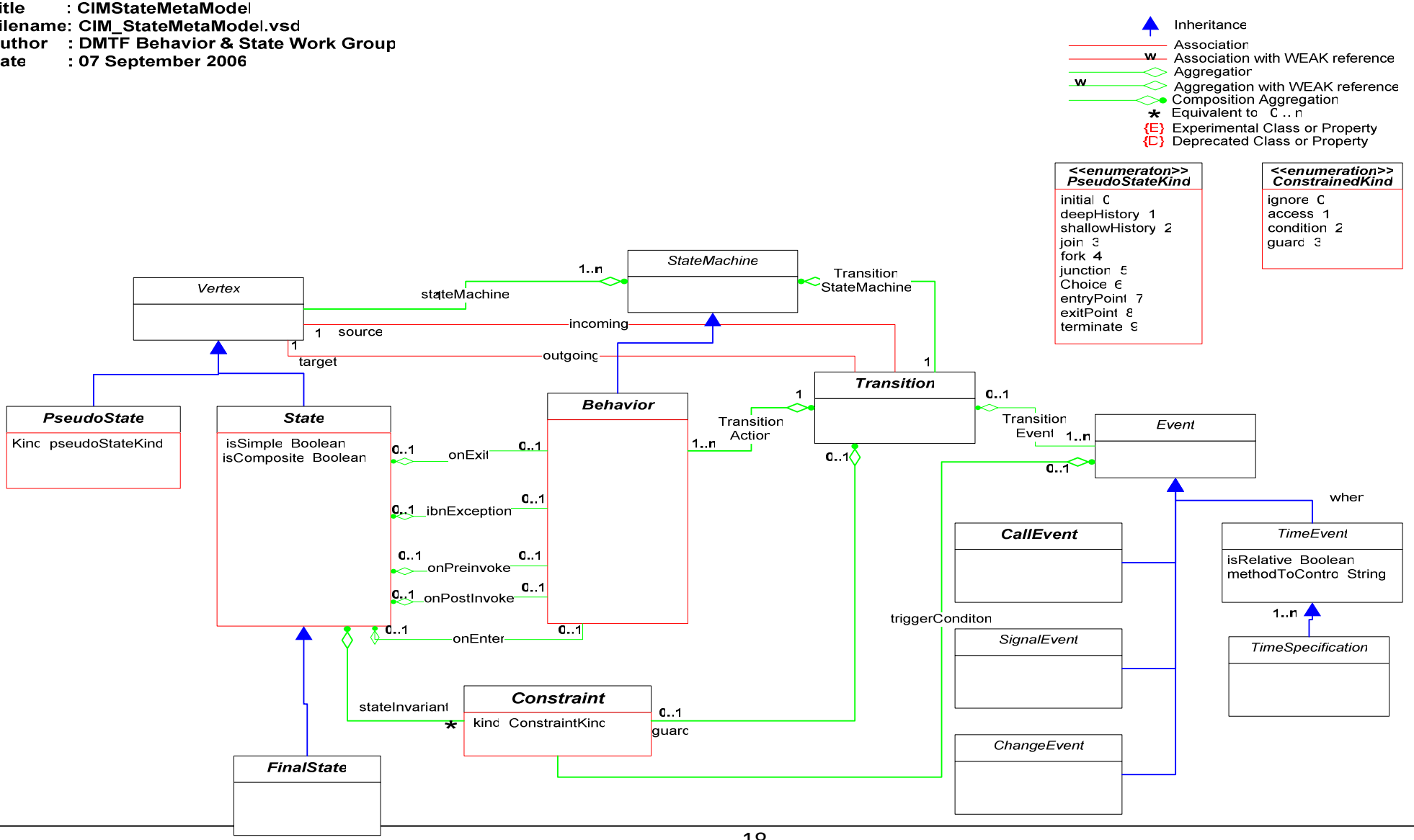
Behavior Meta-Model

- Based ON UML State Meta Model
- Extends CIM meta-model



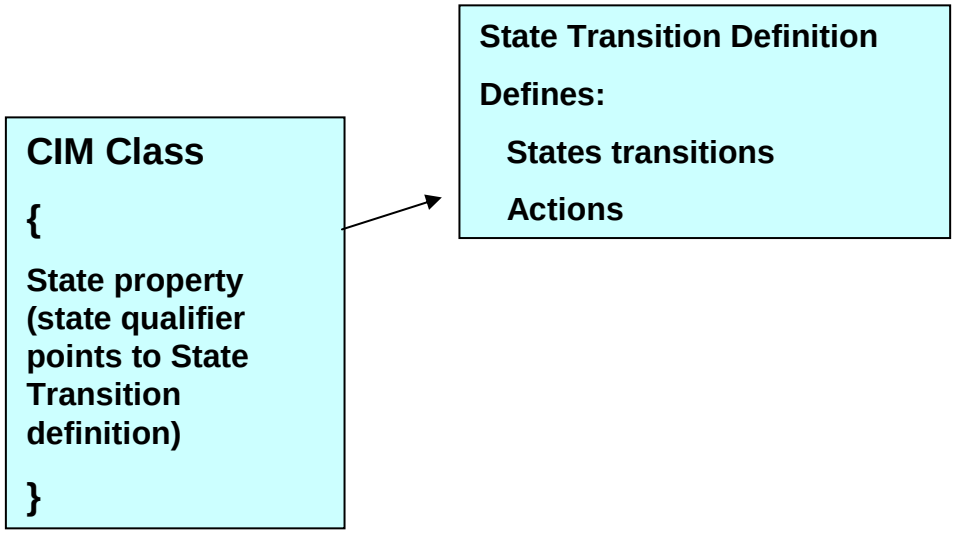
Full Meta Model extension (in process)

Title : CIMStateMetaModel
Filename: CIM_StateMetaModel.vsd
Author : DMTF Behavior & State Work Group
Date : 07 September 2006

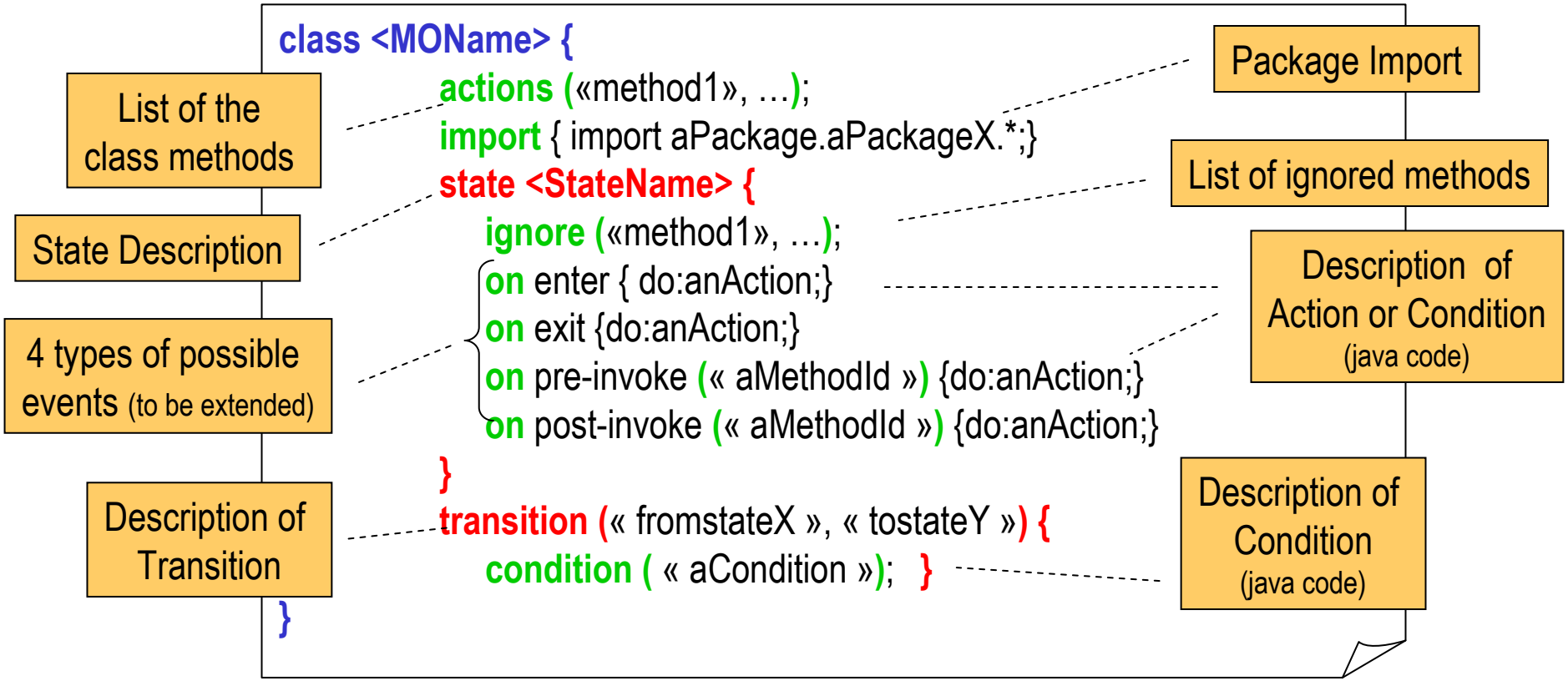


Features of CIMState

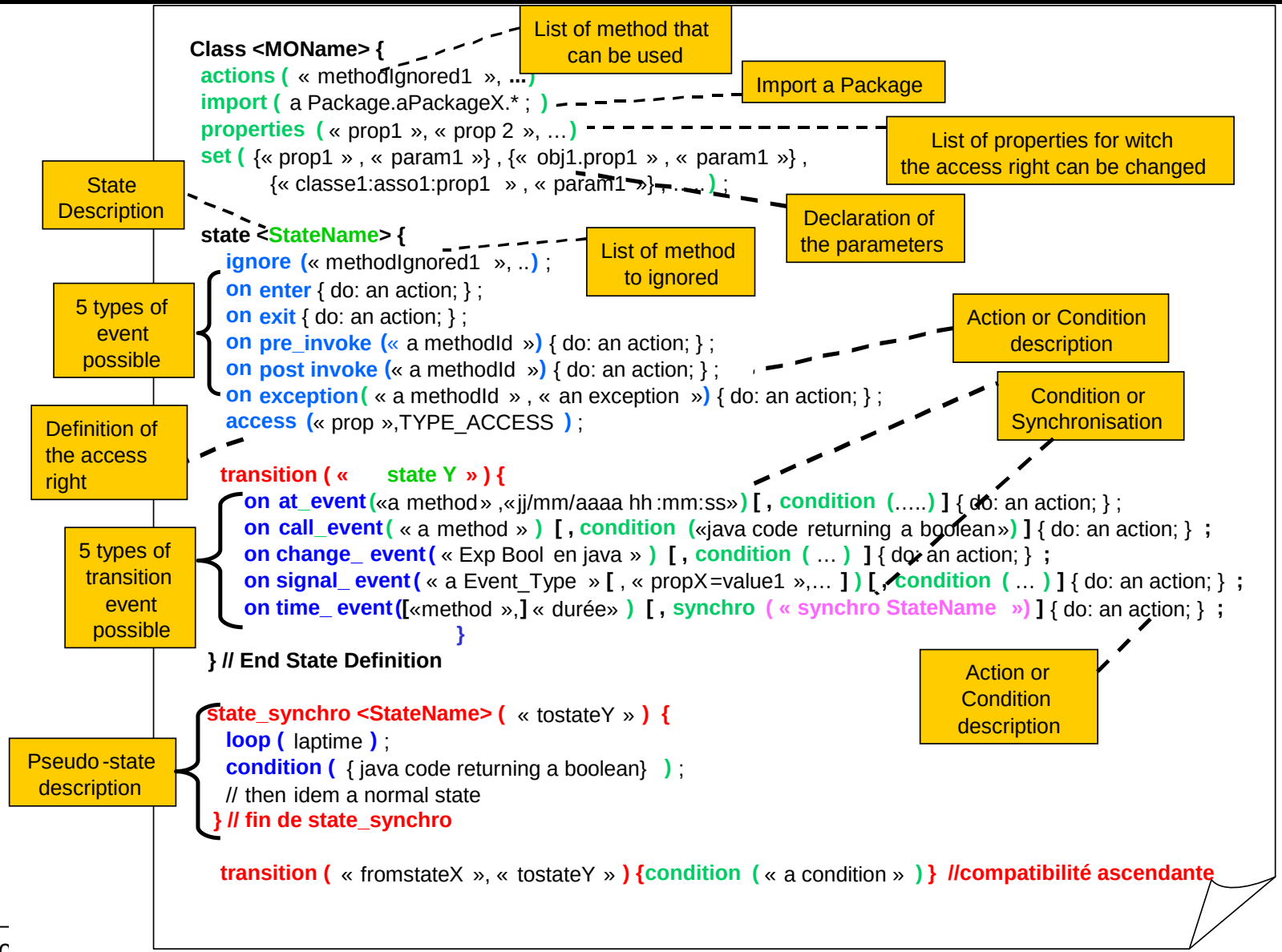
- Define UML StateChart based State Machine definitions
  - Events
  - State Transitions
  - Conditions
  - Actions
- Connect State Machine to Class Model
  - CIMState machines can be attached to Classes (qualifier)
- Define Actions through multiple languages
  - CQL, SPL, OCL, implementation specific languages

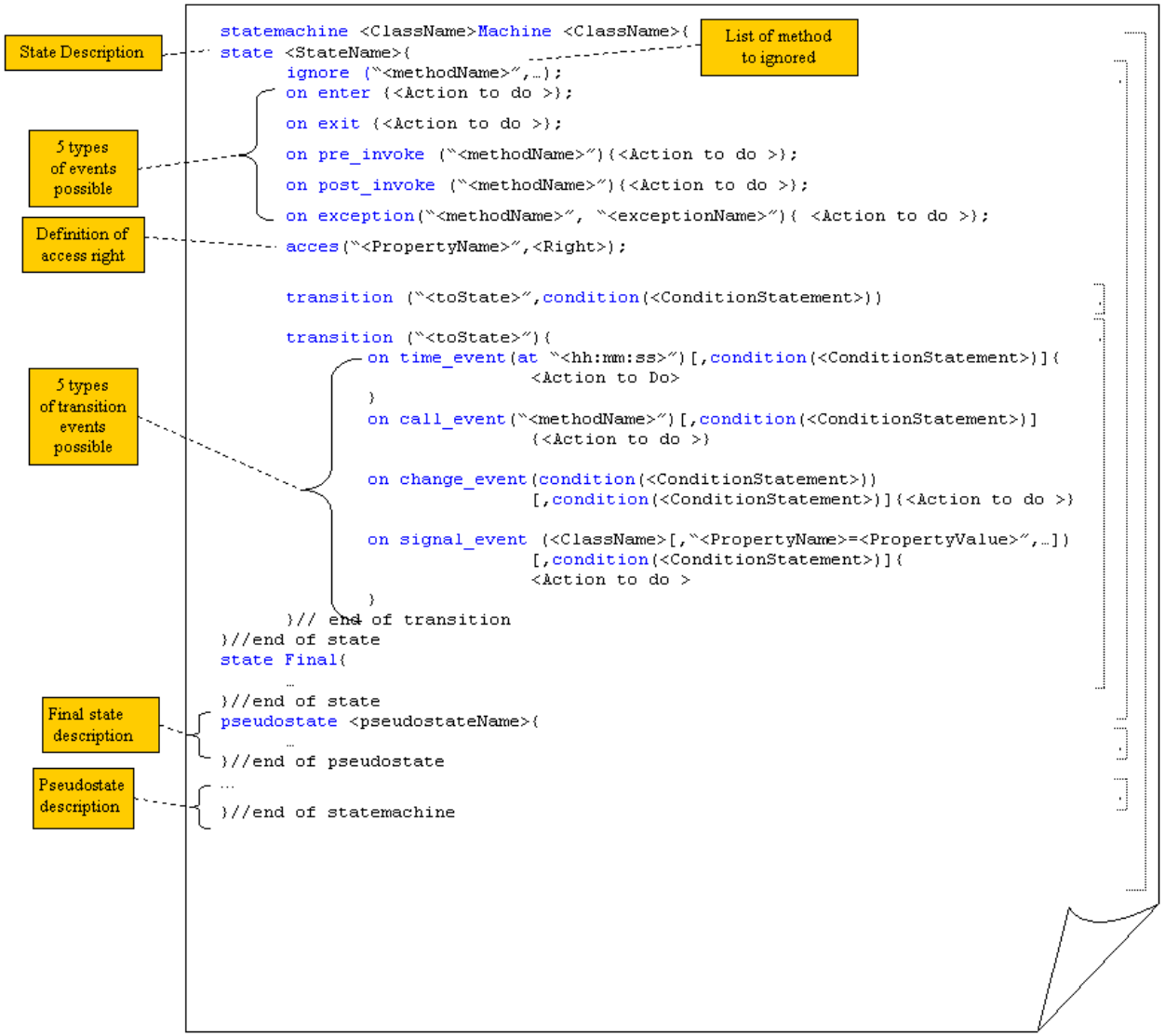


Overview of a CIMState Grammar



CIMState Grammar Overview





Issues today

- Defining Actions and Conditions
  - No single language in CIM for action and condition definition
  - Alternatives include
    - CQL
    - Objects and defined in Policy Model
    - SPL
    - Implementation specific languages (ex. Java for Java implementations)
- Point of Integration into the CIM Models and Profiles
  - This is unclear without more working examples

Expectations for this Work

- Extension to the Meta Model for behavioral elements
- Text based notation representing State models
- Usage
  - It is clear that it is not logical to generally apply state concepts generally at the DMTF CIM Model level (attaching state definitions to CIM Classes in a model release). Models are still largely information models
  - Useful in defining behavior specific to profiles. Behavior becomes essential to profile definition.
  - Useful as component of tools for development of providers.
  - Useful in clarifying the general issues of CIM general model of managed state and managed status concepts, rollup of state and status, etc.
- This work is in process today in the DMTF Behavior and State Work Group

Questions?

DMTF: http://www.dmtf.org/

EMAIL: k.schopmeyer@swbell.net