Non-Functional Analysis for UML Models

Model Processing for Analysis

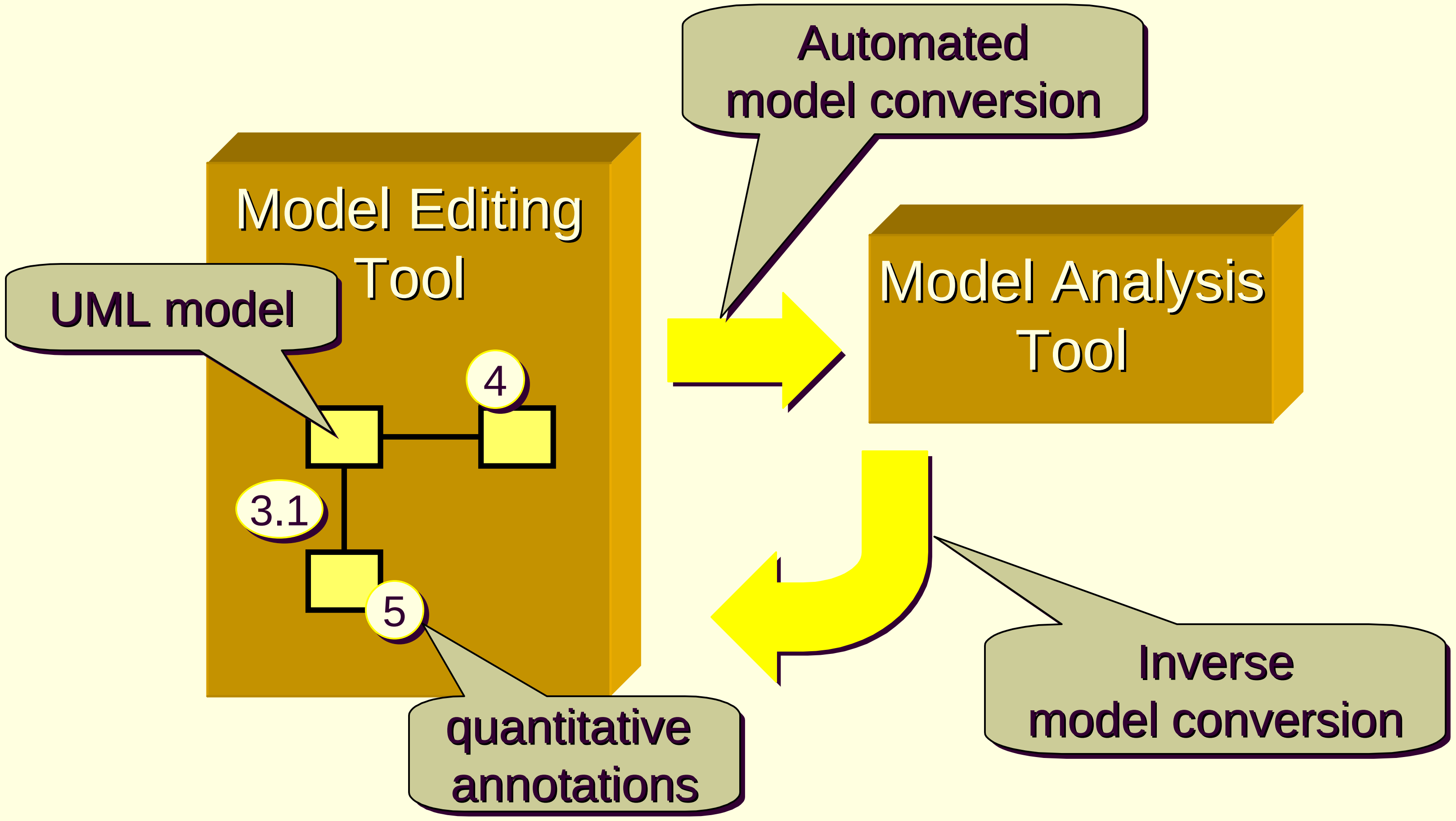
Ben Watson
Tri-Pacific Software, Inc.
watson@tripac.com

The (So Called) Real-Time UML Profile

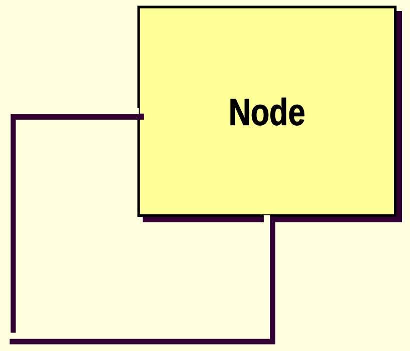
- Officially, it is *The UML Profile for Schedulability, Performance and Time*
- The profile was adopted at the September OMG meeting in Toronto
- The profile addresses the time related non-functional characteristics of a UML model
  - Models for time, resources, concurrency
  - Sub profiles (and models) for schedulability and performance
  - Software and hardware infrastructure and their mapping
  - Specific notations for the above where necessary
    - Stereotypes
    - Tagged values

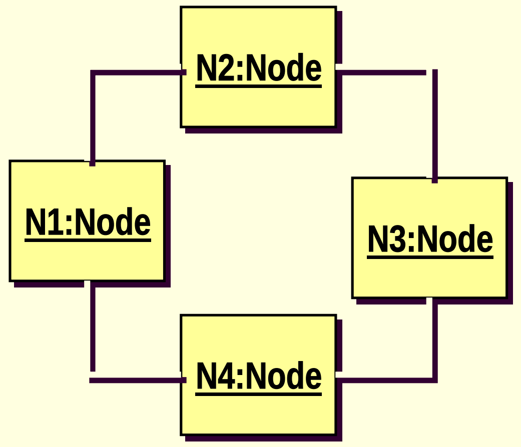
Desired Development Model

- Seamless integration of technologies and tools based on standards for real-time modeling



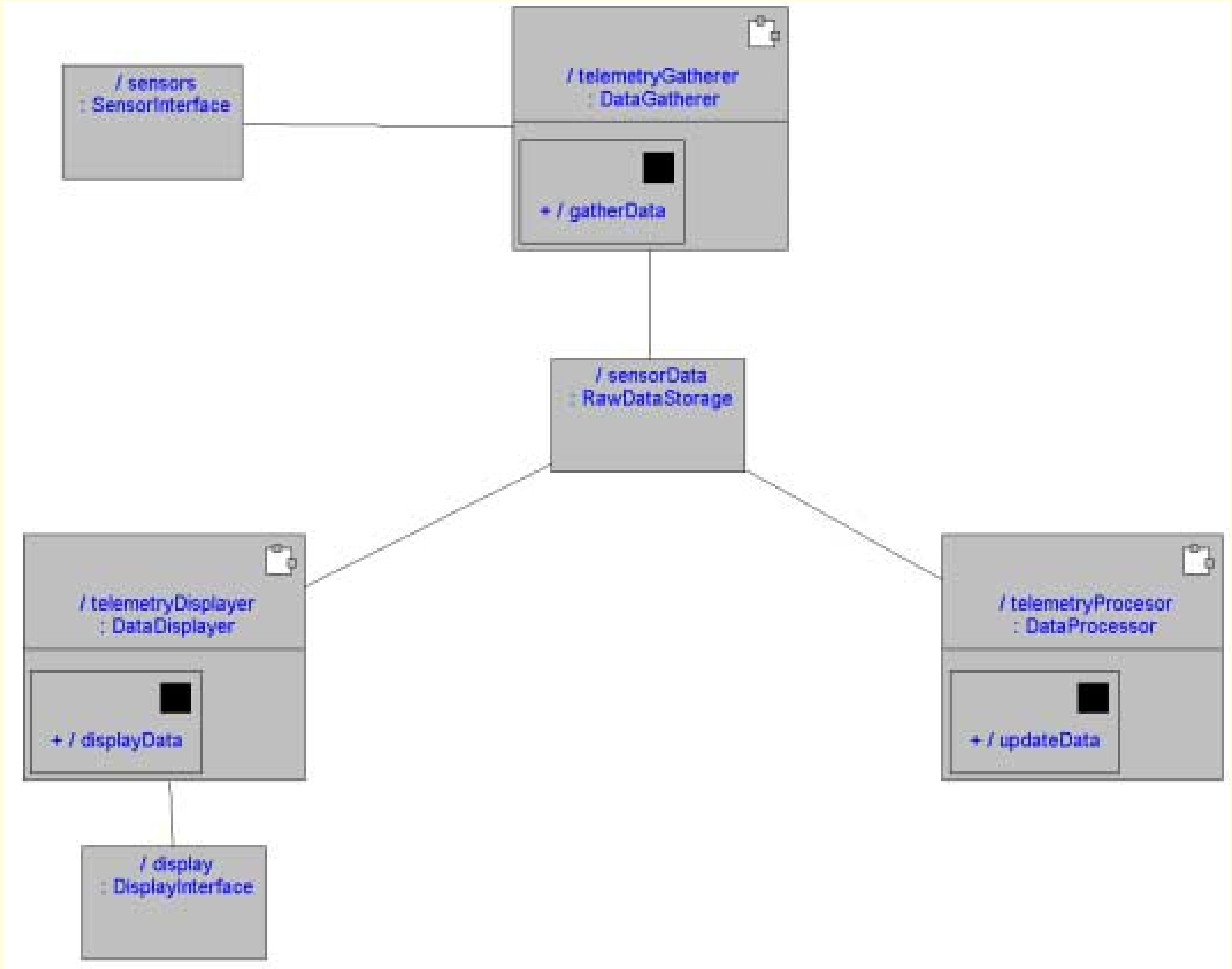
Instance- vs Class-Based Models



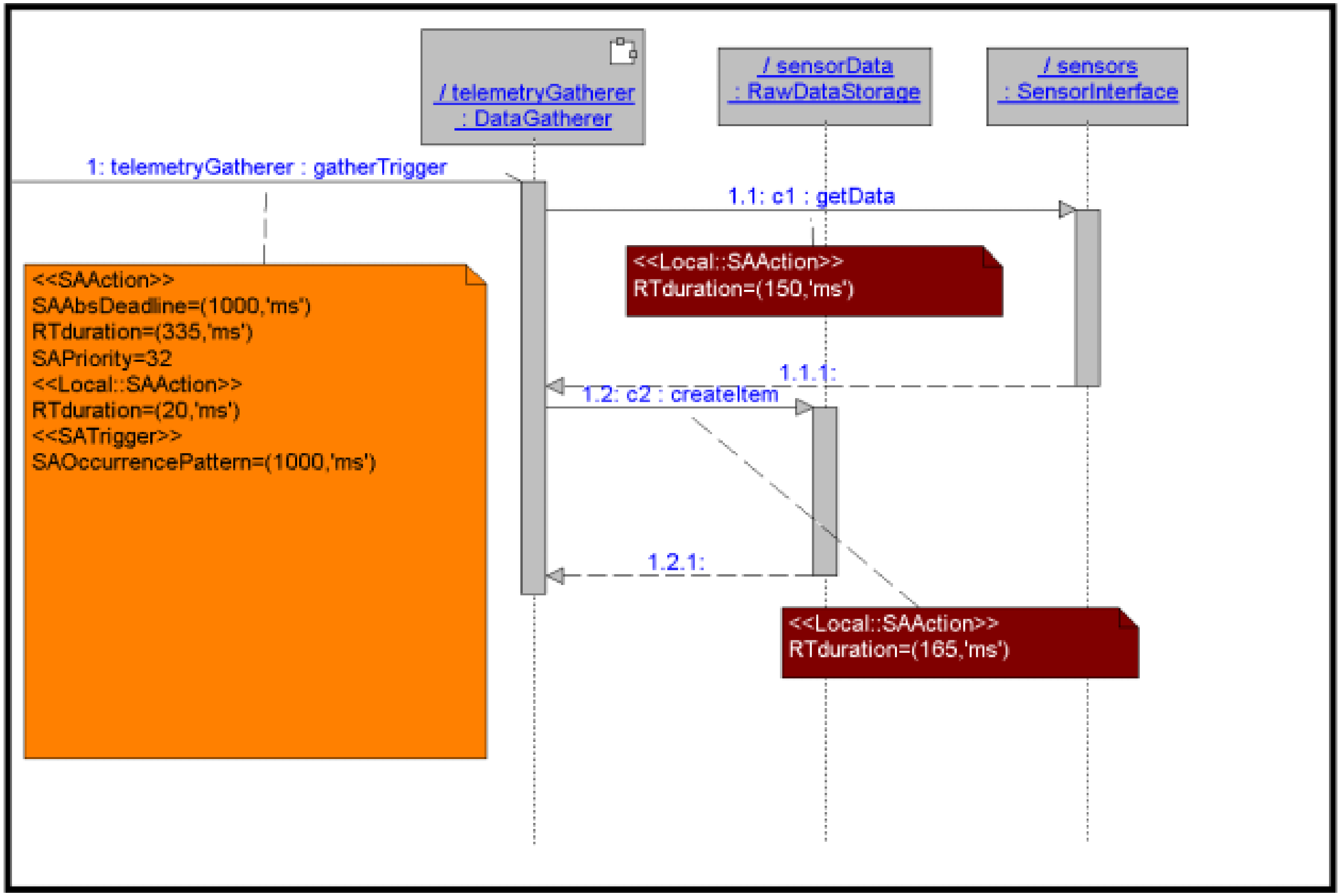


- Practically all analysis methods are concerned with instance-based models
- However, it is often useful to associate QoS characteristics with classes
  - Used to define default values that may be overridden for specific instances
- Need to apply a stereotype to both spec elements and instance elements

Example: Collaboration



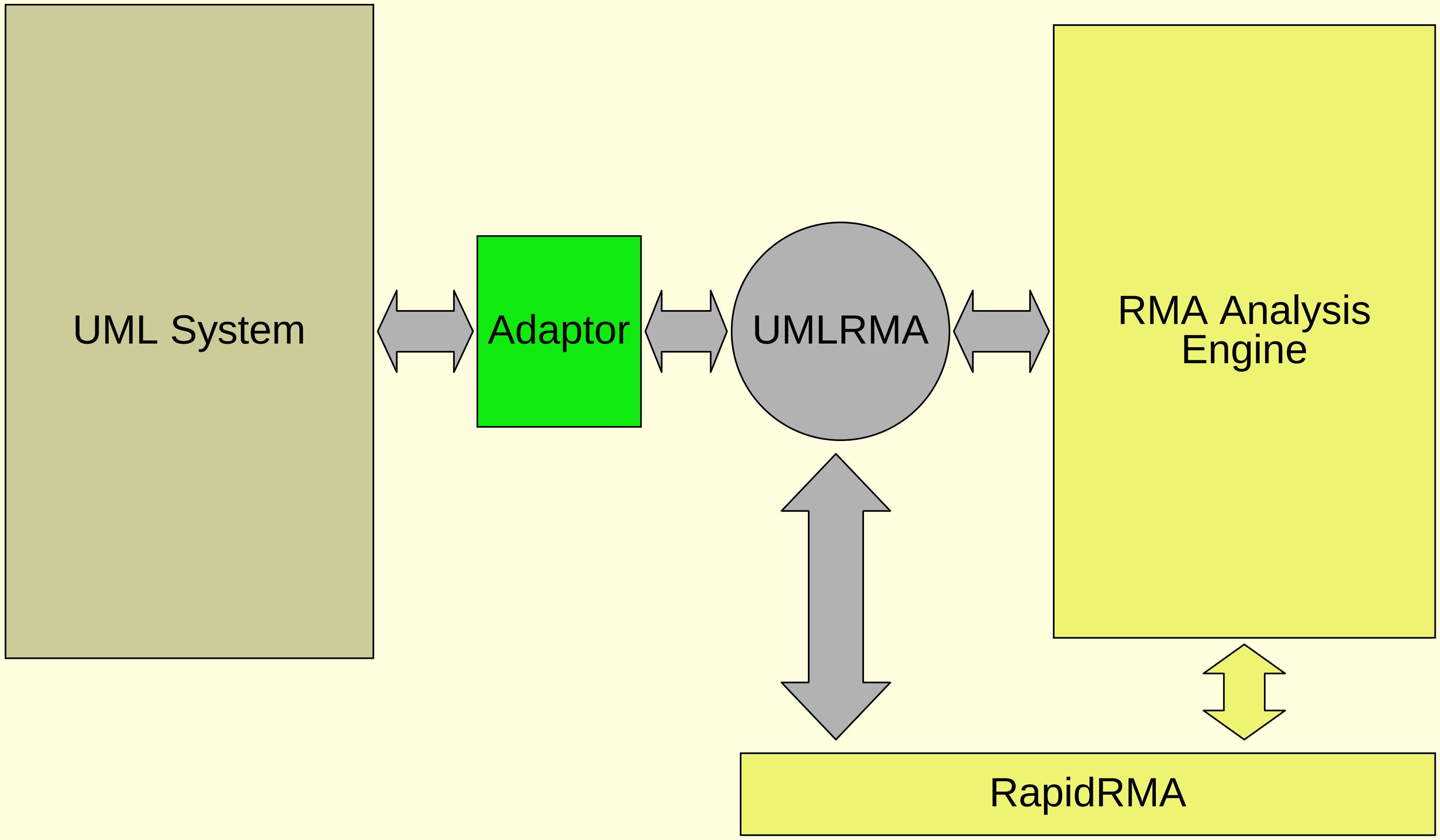
Example: Annotated Sequence



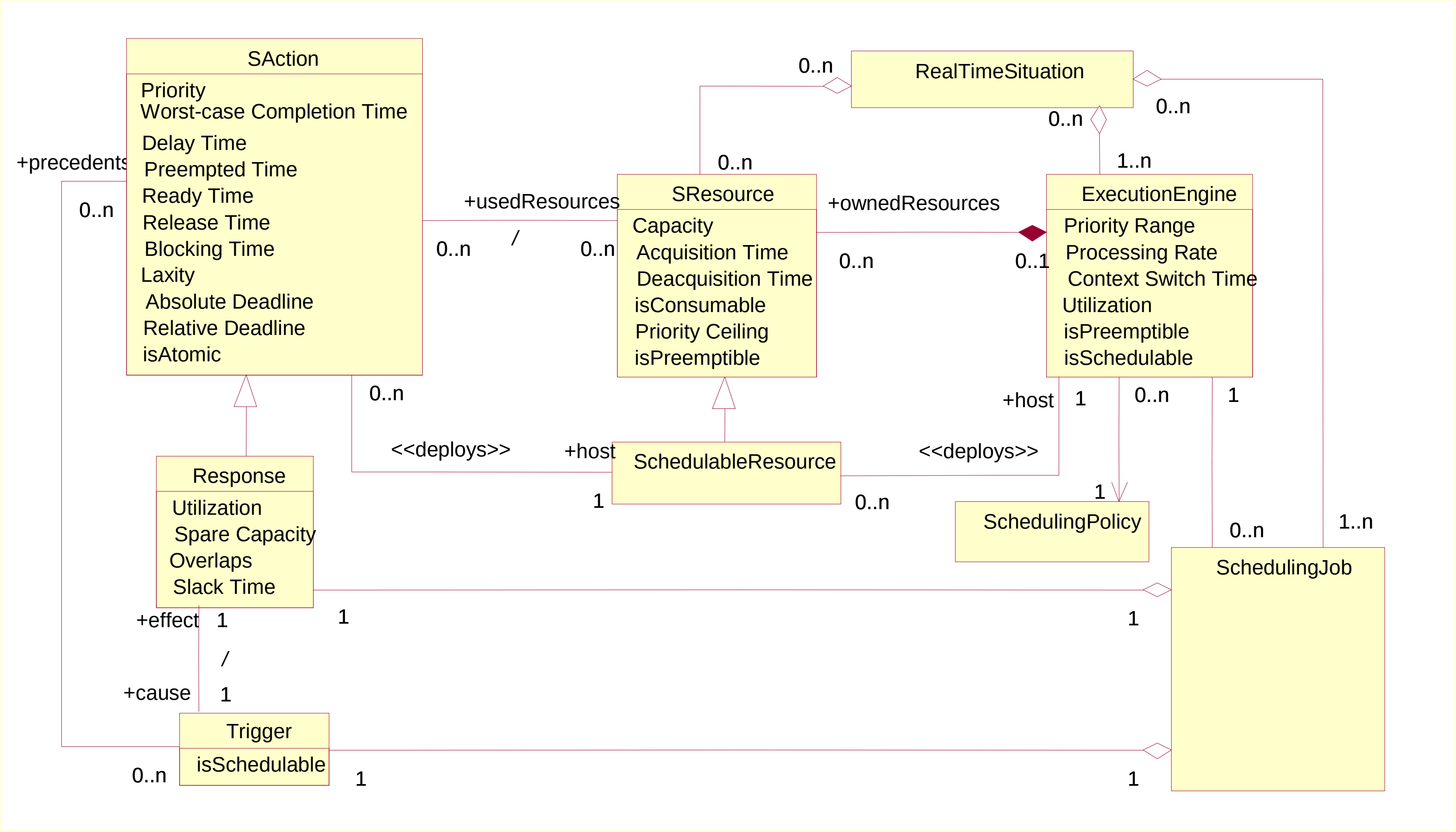
What We Needed to Build

- A schedulability analysis model processor
- Two issues to address
  - The program architecture
  - Extracting a timing model from the UML model
- Starting point for the model processor was RapidRMA, our Rate Monotonic Analysis (RMA) tool
- Our goal: To make the integration with multiple UML tools as seamless as possible
  - Make it unobtrusive (look like the host application)
  - Provide complete RMA tool capability
  - Do it interactively

Program Architecture



Schedulability Analysis Sub-Profile



Defined Stereotypes (1 of 3)

Stereotype	Applies To	Tags	Description
«SAAction» (subclass of «RTaction» and «CRAction»)	Action, ActionExecution, Stimulus, Action, Message, Method…	SAPriority [0..1] SAActualPty [0..1] SABlocking [0..1] SAReady [0..1] SADelay [0..1] SARelease [0..1] SAPreempted [0..1] SAWorstCase [0..1] SALaxity [0..1] SAPriority [0..1] SAAbsDeadline [0..1] SARelDeadline [0..1] SAusedResource [0..1] SAhost [0..1]	An action
«SAEngine»	Node, Instance, Object, Classifier, ClassifierRole	SASchedulingPolicy [0..1] SAAccessPolicy [0..1] SARate [0..1] SAContextSwitch [0..1] SAPriorityRange [0..1] SAPreemptible [0..1] SAUtilization [0..1] SASchedulable [0..1] Saresources [0..1]	An execution engine

Defined Stereotypes (2 of 3)

Stereotype	Applies To	Tags	Description
«SAOwns» (subclass of «GRMrealize»)	Abstraction		Identifies ownership of resources
«SAPrecedes»	Usage		A precedence relationship between actions and triggers
«SAResource»	Classifier, ClassifierRole, Instance, Object, Node	SAAccessControl [0..1] SAConsumable [0..1] SACapacity [0..1] SAAcquisition [0..1] SADeacquisition [0..1] SAPtyCeiling [0..1] SAPreemptible [0..1]	A resource of some kind
«SAResponse» (subclass of «SAAction»)	Action, ActionExecution, Stimulus, Action, Message, Method…	SAUtilization [0..1] SASpare [0..1] SASlack [0..1] SAOverlaps [0..1]	A response to a stimulus or action
«SASchedulable» (subclass of «SAResource»)	Classifier, ClassifierRole, Instance, Object, Node		A schedulable resource

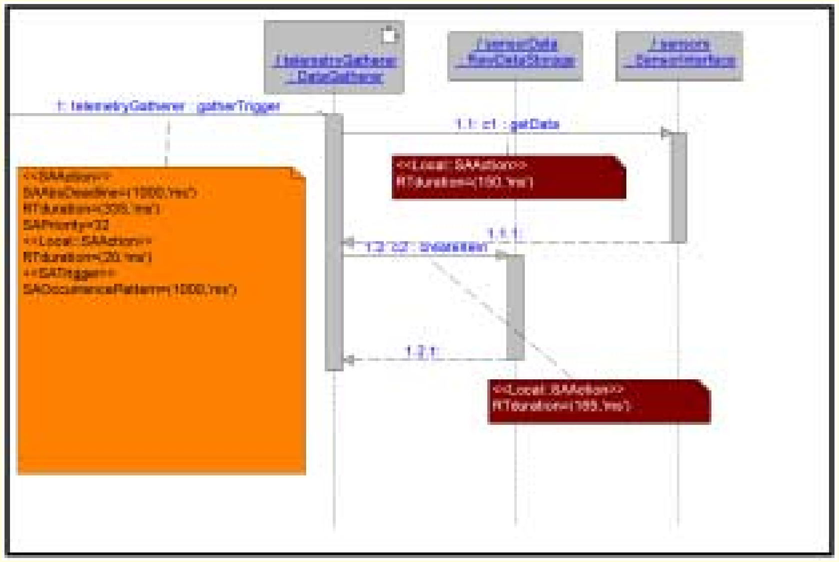
Defined Stereotypes (3 of 3)

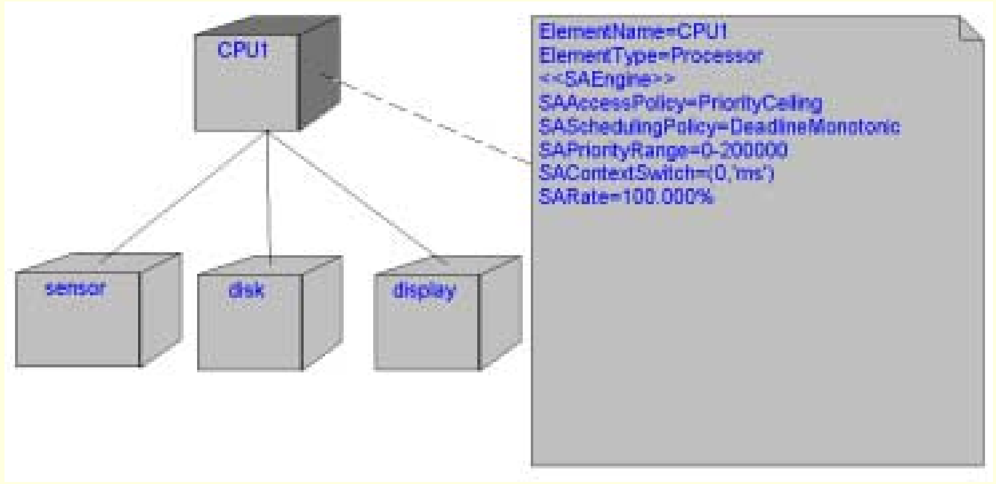
Stereotype	Applies To	Tags	Description
«SAScheduler»	Classifier, ClassifierRole, Instance, Object	SASchedulingPolicy [0..1] SAExecutionEngine [0..1]	A scheduler
«SAPrecedes»	Usage		A precedence relationship between actions and triggers
«SASituation»	Collaboration, CollaborationInstance, ActivityGraph		A schedulability analysis context
«SATrigger» (subclass of «SAAction»)	Message, Stimulus	SASchedulable [0..1] SASAprecedents [0..1]	A trigger
«SAusedHost»	Usage		Identifies schedulable resources used for execution of actions
«SAUses»	Usage		Identifies sharable resources

Minimum Annotations for Schedulability

- External signals and time triggered internal signals
  - Occurrence pattern
  - Deadline
- Actions that process the signals
  - Execution time
  - Action sequence
    - Precedence
    - Synchronous / asynchronous
- Deployment
  - Processor
  - Device
  - Instance

Minimum Annotations





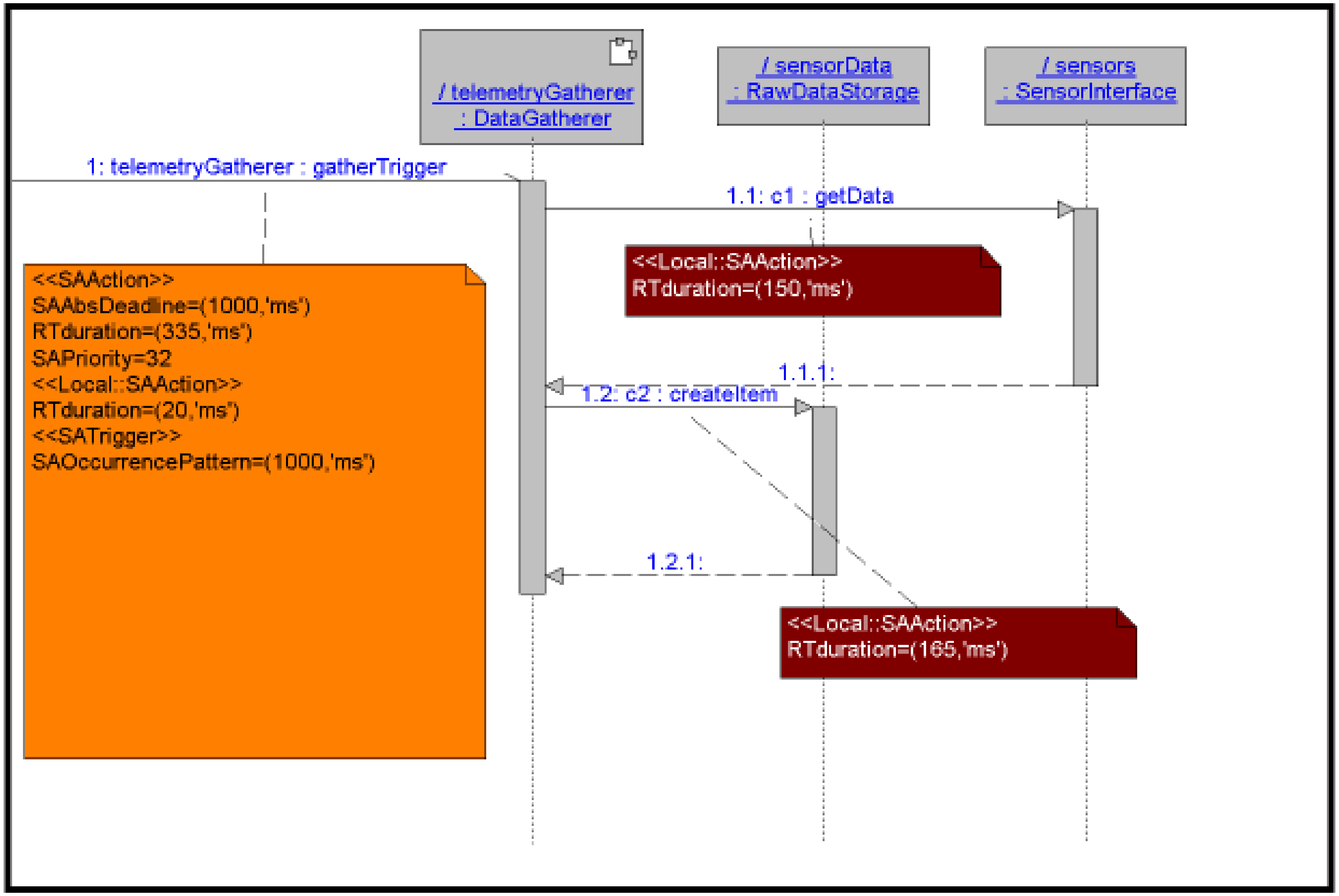
Classifiers and Instances

- All schedulability analysis is instance-based
- Annotations on a classifier are permitted
  - Default value for the entire class
  - An annotation on an instance overrides the classifier annotation
- Weak support for instances in UML tools
  - No method to correlate instances on different sequence diagrams
  - Adopt the convention that identical instance names refer to the same instance
- It is important to know when actions belong to the same instance of an object due to run-to-completion semantics

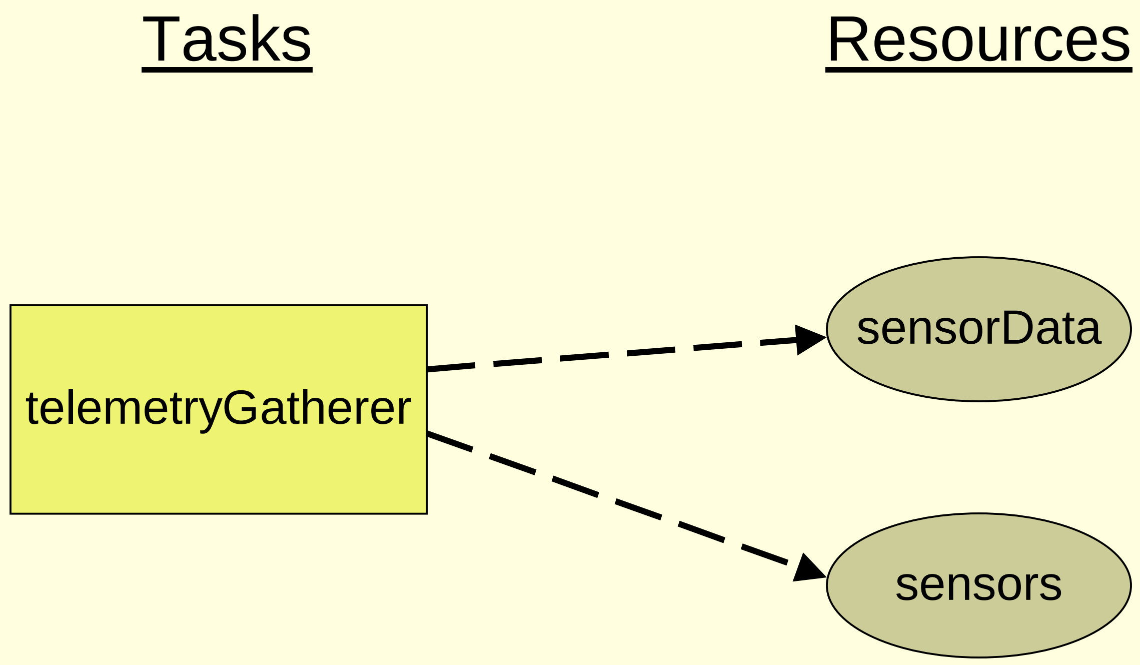
Rules to Extract Timing Model

- The sequence diagrams determine the timing model
- Locate all external signals
  - Incoming from the environment
- Determine arrival pattern and deadline from <<SATrigger>> and <<SAAction>> stereotypes
- Determine the action that is the response to the trigger event
  - Single action
  - Action sequence (precedence)
  - <<SAAction>> and <<local::SAAction>>
  - Action sequence inherits the trigger occurrence pattern
  - End-to-end deadline
- Determine tasks and resources
  - Synchronous vs asynchronous messages

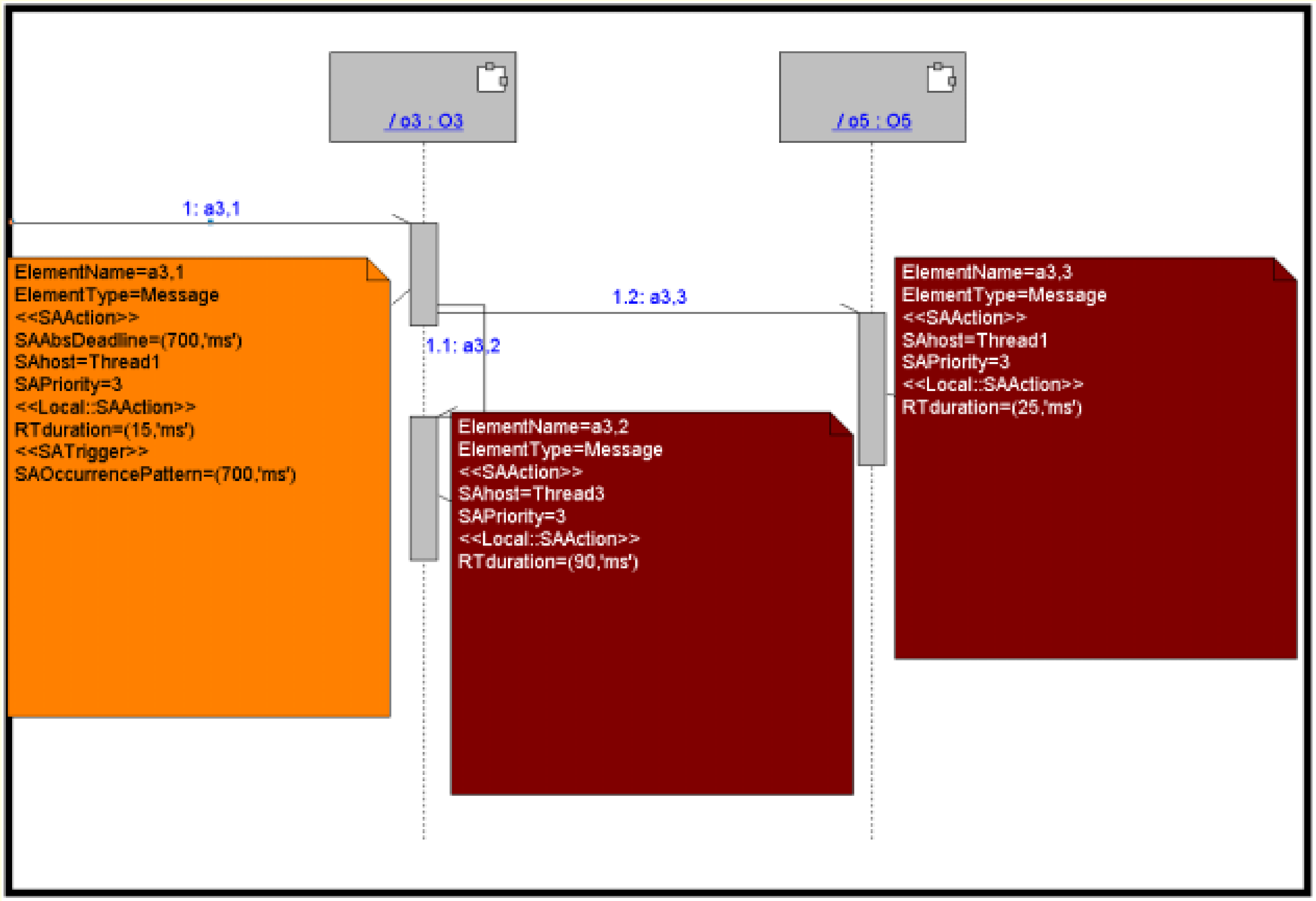
Example Sequence Diagram



Timing Model



Another Example



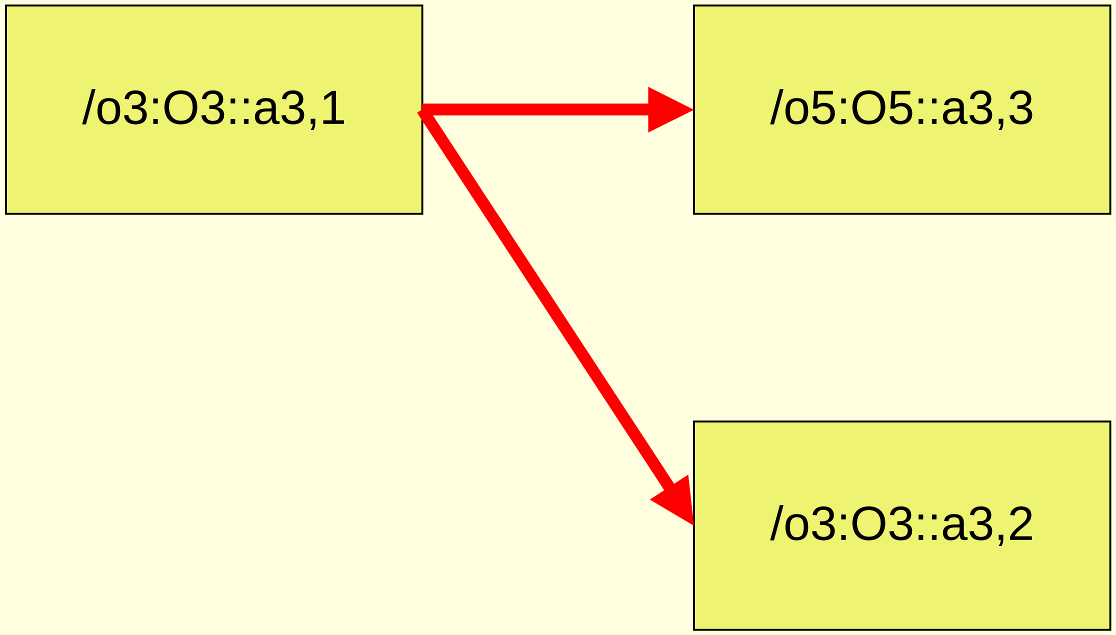
ElementName=a3,1
ElementType=Message
<<SAAction>>
SAAbsDeadline=(700,'ms')
SAhost=Thread1
SAPriority=3
<<Local::SAAction>>
RTduration=(15,'ms')
<<SATrigger>>
SAOccurrencePattern=(700,'ms')

ElementName=a3,2
ElementType=Message
<<SAAction>>
SAhost=Thread3
SAPriority=3
<<Local::SAAction>>
RTduration=(90,'ms')

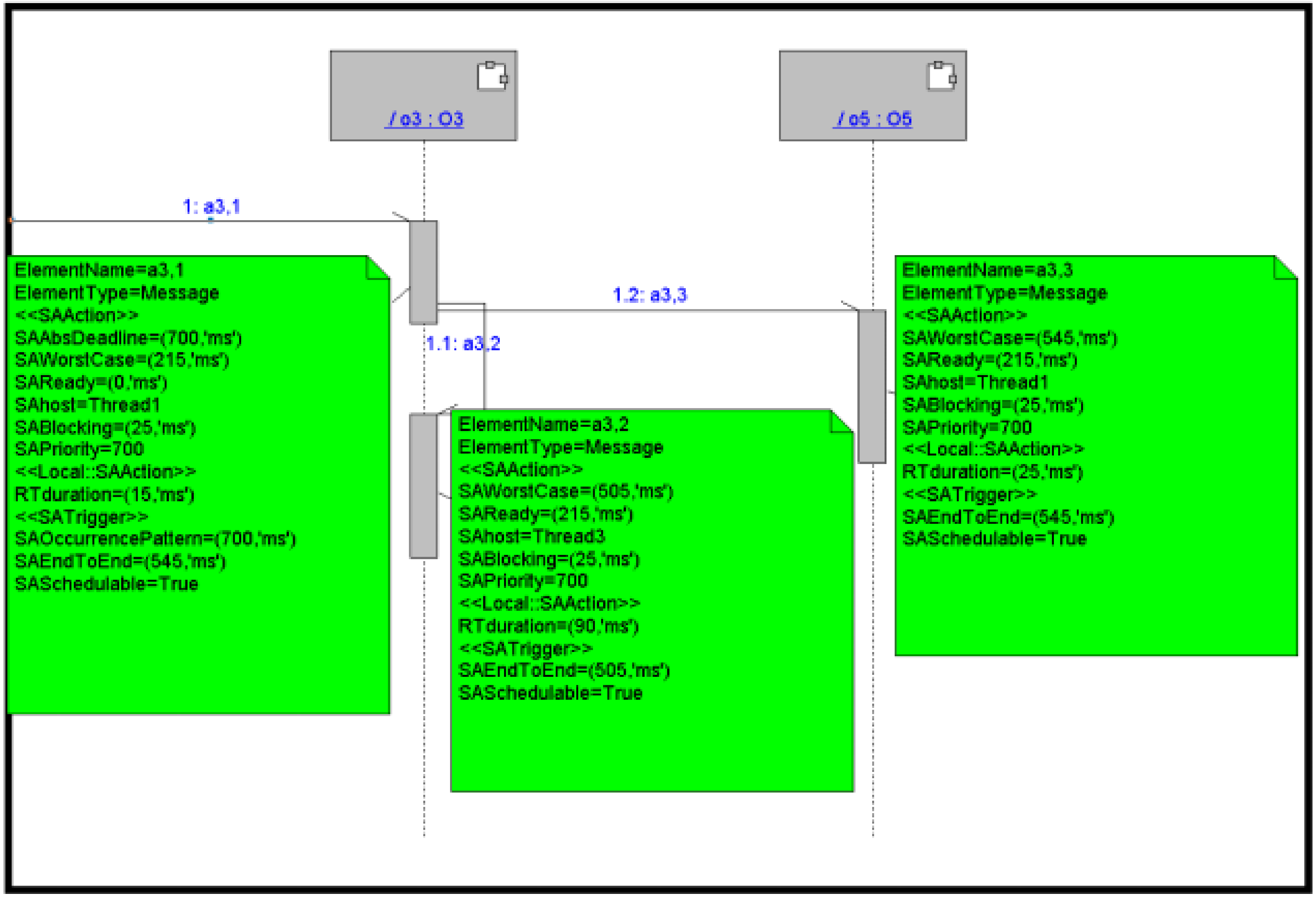
ElementName=a3,3
ElementType=Message
<<SAAction>>
SAhost=Thread1
SAPriority=3
<<Local::SAAction>>
RTduration=(25,'ms')

Timing Model

Tasks



Results Example



ElementName=a3,1
ElementType=Message
<<SAAction>>
SAAbsDeadline=(700,'ms')
SAWorstCase=(215,'ms')
SAReady=(0,'ms')
SAhost=Thread1
SABlocking=(25,'ms')
SAPriority=700
<<Local::SAAction>>
RTduration=(15,'ms')
<<SATrigger>>
SAOccurrencePattern=(700,'ms')
SAEndToEnd=(545,'ms')
SASchedulable=True

ElementName=a3,2
ElementType=Message
<<SAAction>>
SAWorstCase=(505,'ms')
SAReady=(215,'ms')
SAhost=Thread3
SABlocking=(25,'ms')
SAPriority=700
<<Local::SAAction>>
RTduration=(90,'ms')
<<SATrigger>>
SAEndToEnd=(505,'ms')
SASchedulable=True

ElementName=a3,3
ElementType=Message
<<SAAction>>
SAWorstCase=(545,'ms')
SAReady=(215,'ms')
SAhost=Thread1
SABlocking=(25,'ms')
SAPriority=700
<<Local::SAAction>>
RTduration=(25,'ms')
<<SATrigger>>
SAEndToEnd=(545,'ms')
SASchedulable=True

Conclusion

- We have implemented a model processor for the RT UML profile
  - Conforms to the standard
  - Meets our “seamless” goals
- Future work
  - Implement the entire standard
    - Layered models
    - Parameterized tagged values
  - Extensions to the standard
    - Stochastic analysis
    - Scripting interface

Questions?