Non-Functional Analysis for UML Models

Model Processing for Analysis

Ben Watson
Tri-Pacific Software, Inc.
watson@tripac.com
The 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
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
### Defined Stereotypes (1 of 3)

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Applies To</th>
<th>Tags</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>«SAAction» (subclass of «RTaction» and «CRAction»)</td>
<td>Action, ActionExecution, Stimulus, Action, Message, Method…</td>
<td>SAPriority [0..1] \nSAActualPty [0..1] \nSABlocking [0..1] \nSAReady [0..1] \nSADelay [0..1] \nSARelase [0..1] \nSAPreempted [0..1] \nSAWorstCase [0..1] \nSALaxity [0..1] \nSAPriority [0..1] \nSAAbsDeadline [0..1] \nSARelDeadline [0..1] \nSAUsedResource [0..1] \nSAhost [0..1]</td>
<td>An action</td>
</tr>
<tr>
<td>«SAEngine»</td>
<td>Node, Instance, Object, Classifier, ClassifierRole</td>
<td>SASchedulingPolicy [0..1] \nSAAccessPolicy [0..1] \nSARate [0..1] \nSAContextSwitch [0..1] \nSAPriorityRange [0..1] \nSAPreemptible [0..1] \nSAUtilization [0..1] \nSASchedulable [0..1] \nSaresources [0..1]</td>
<td>An execution engine</td>
</tr>
</tbody>
</table>
### Defined Stereotypes (2 of 3)

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Applies To</th>
<th>Tags</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>«SAOwns» (subclass of «GRMrealize»)</td>
<td>Abstraction</td>
<td></td>
<td>Identifies ownership of resources</td>
</tr>
<tr>
<td>«SAPrecedes»</td>
<td>Usage</td>
<td></td>
<td>A precedence relationship between actions and triggers</td>
</tr>
<tr>
<td>«SAResource»</td>
<td>Classifier, ClassifierRole, Instance, Object, Node</td>
<td>SAAccessControl [0..1] SAConsumable [0..1] SACapacity [0..1] SAAcquisition [0..1] SADeacquisition [0..1] SAPtyCeiling [0..1] SAPreemptible [0..1]</td>
<td>A resource of some kind</td>
</tr>
<tr>
<td>«SAResponse» (subclass of «SAAAction»)</td>
<td>Action, ActionExecution, Stimulus, Action, Message, Method...</td>
<td>SAUtilization [0..1] SASpare [0..1] SASlack [0..1] SAOverlaps [0..1]</td>
<td>A response to a stimulus or action</td>
</tr>
<tr>
<td>«SASchedulable» (subclass of «SAResource»)</td>
<td>Classifier, ClassifierRole, Instance, Object, Node</td>
<td></td>
<td>A schedulable resource</td>
</tr>
</tbody>
</table>
## Defined Stereotypes (3 of 3)

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Applies To</th>
<th>Tags</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>«SAScheduler»</td>
<td>Classifier, ClassifierRole, Instance, Object</td>
<td>SASchedulingPolicy [0..1] SASchedulingPolicy [0..1]</td>
<td>A scheduler</td>
</tr>
<tr>
<td>«SAPrecedes»</td>
<td>Usage</td>
<td></td>
<td>A precedence relationship between actions and triggers</td>
</tr>
<tr>
<td>«SASituation»</td>
<td>Collaboration, CollaborationInstance, ActivityGraph</td>
<td></td>
<td>A schedulability analysis context</td>
</tr>
<tr>
<td>«SATrigger» (subclass of «SAAction»)</td>
<td>Message, Stimulus</td>
<td>SAScheduled [0..1] SASAprecedes [0..1]</td>
<td>A trigger</td>
</tr>
<tr>
<td>«SAusedHost»</td>
<td>Usage</td>
<td></td>
<td>Identifies schedulable resources used for execution of actions</td>
</tr>
<tr>
<td>«SAUses»</td>
<td>Usage</td>
<td></td>
<td>Identifies sharable resources</td>
</tr>
</tbody>
</table>
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>>
    - Inherits the trigger occurrence pattern
    - End-to-end deadline
- Determine tasks and resources
  - Synchronous vs asynchronous messages
Example Sequence Diagram
Timing Model

Tasks

Resources

telemetryGatherer

sensorData

sensors
Another Example
Timing Model

Tasks

/o3:O3::a3,1 → /o5:O5::a3,3
/o3:O3::a3,2
Results Example
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?