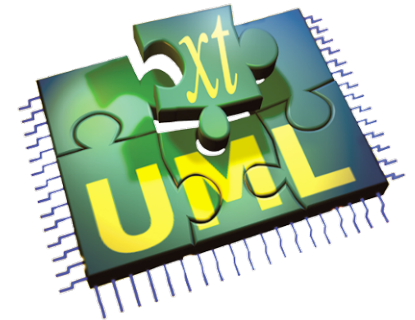


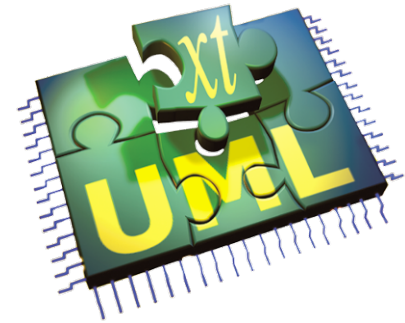
The xtUML method - Verification

- ◆ **Analysis** – questioning, thinking, sketching...
 - Descriptive UML diagrams
 - use case, sequence, ...
- ◆ **Executable Modeling** – formalizing the analysis:
 - Component Diagrams (partitioning/interfaces)
 - Class Diagrams (data)
 - State Machines (control)
 - Activities (processing)
- ◆ **Verification**
 - **Interpretive Model Execution**
- ◆ **Code generation**
 - Template and Rule-Based Translation



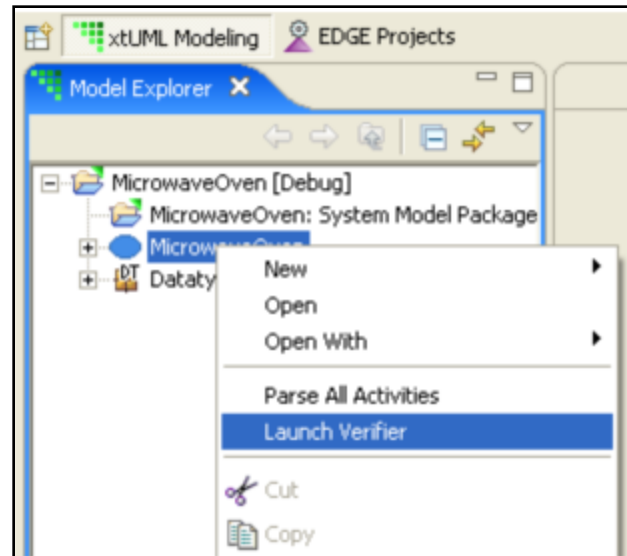
Verifier Overview

- ◆ **Execute your models using the BridgePoint Verifier**
- ◆ **Allows interactive debugging of the model without translation into platform specific code**
- ◆ **Functions like a normal debugger**
 - Breakpoints
 - Event queue for pending events
 - Variable inspection
 - Stepping (into/over)



xtUML Verification Perspective

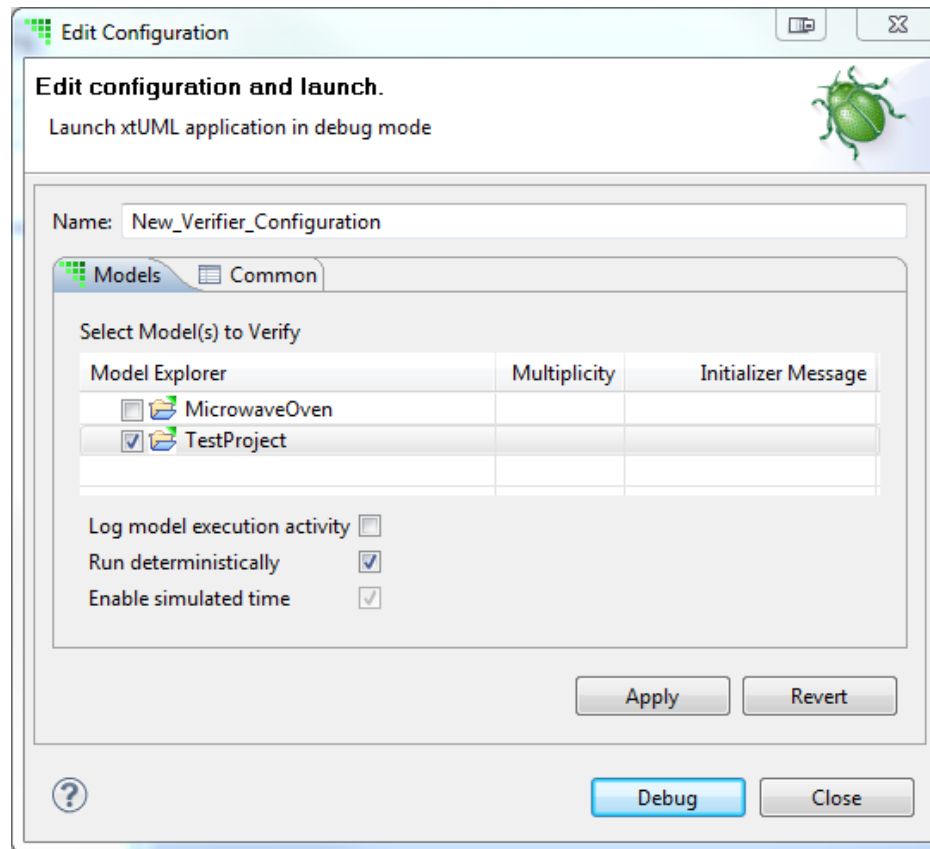
- ◆ Can be launched via right-click on the model in the Model Explorer perspective



- ◆ xtUML Debugging perspective will open and a Model Verifier Application configuration will appear. Select the model to launch in Verifier

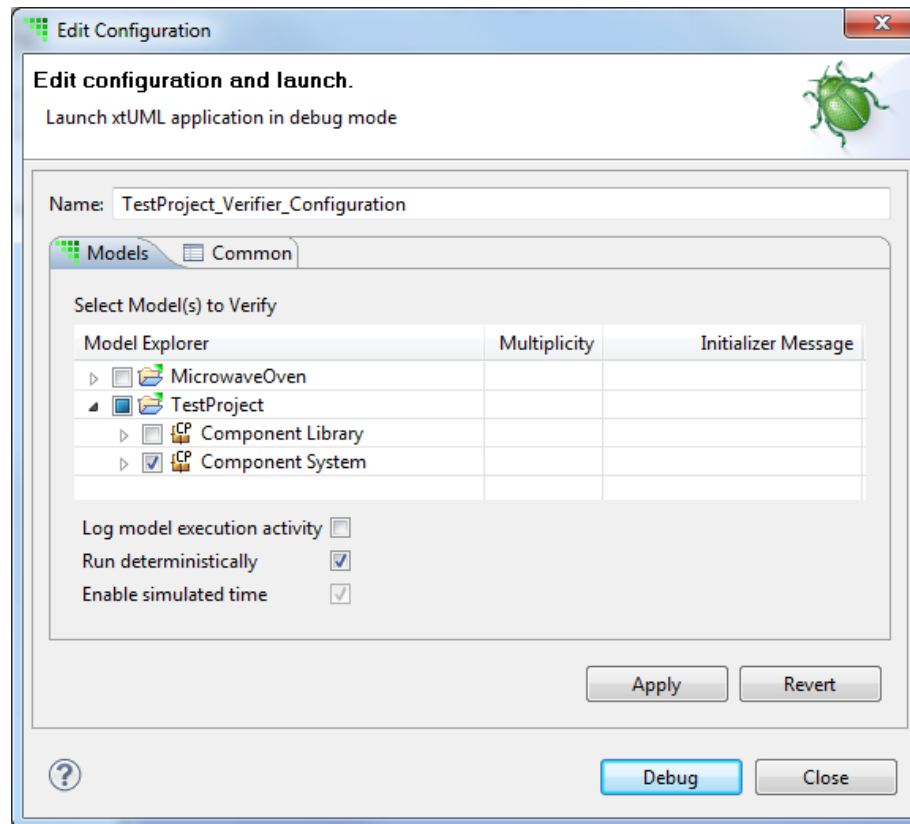
Verifier Debug Configurations

- ◆ Verifier requires that at least one debug configuration be created for each model
- ◆ Select a model to verify, click 'Debug' to start



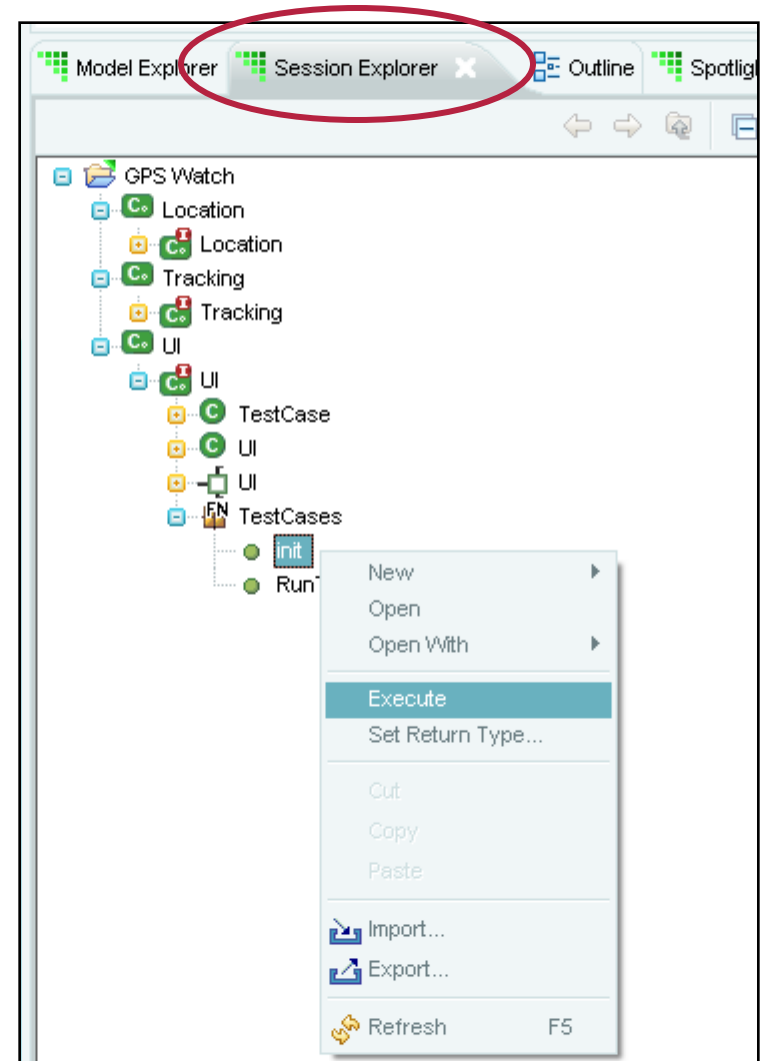
Managing Verifier Debug Configurations

- ◆ Useful for selecting which components to debug in a session
- ◆ At least one component must be selected to run Verifier



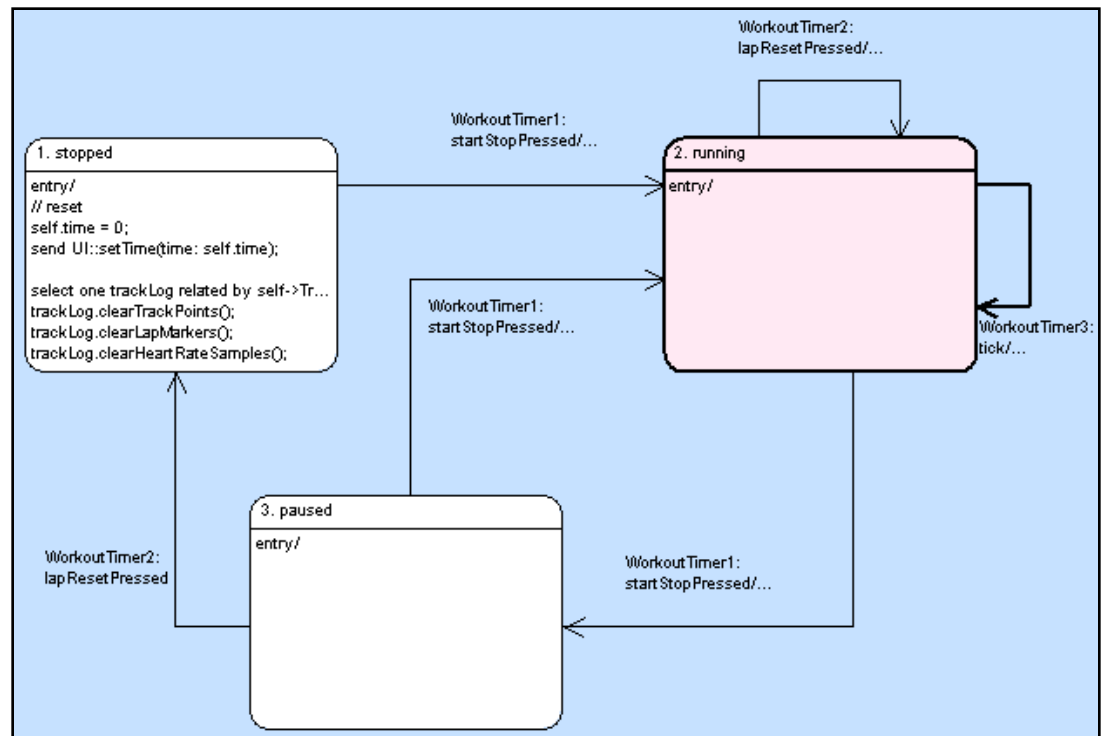
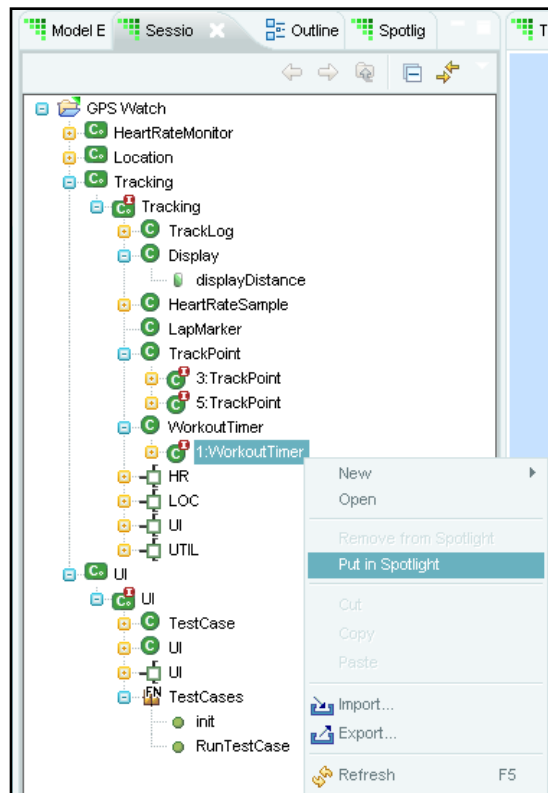
Executing Models in Verifier

- ◆ **Models do not run automatically, functions must be manually invoked**
- ◆ **Functions contained in Function Packages will create instances of classes involved if state machines do not have creation transitions**
- ◆ **Port operations and signals can also be executed via the context menu**
- ◆ **Context menu “execute” option not available in Model Explorer window, must use Session Explorer**



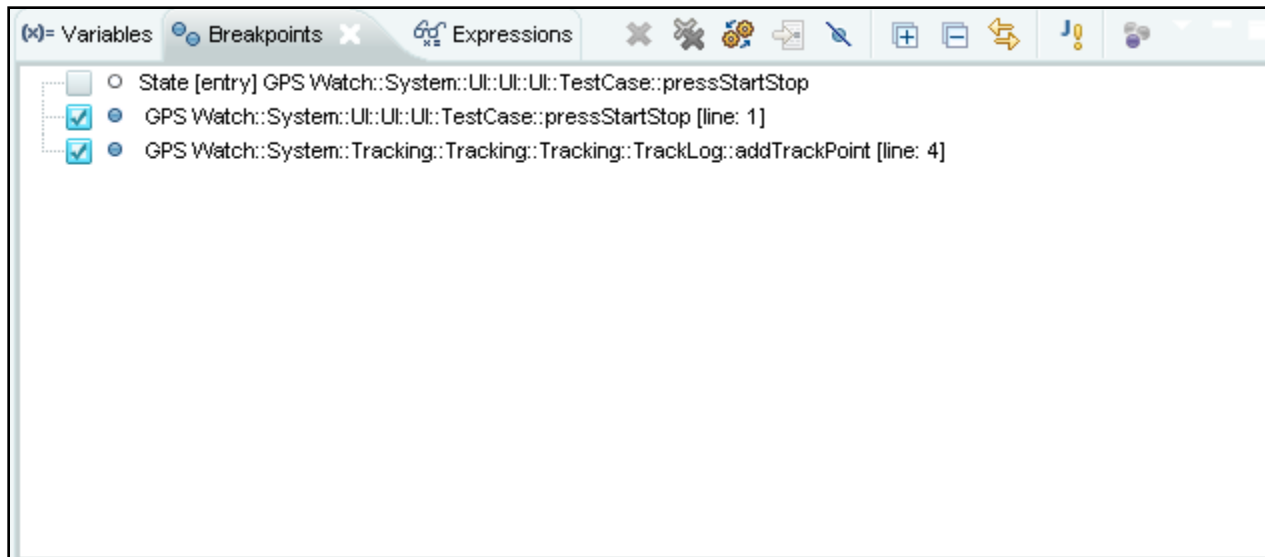
Spotlight in Verifier

- ◆ **Spotlight is a method of highlighting current states in state machines**
- ◆ **Transitions that caused the state to be entered are also highlighted**



Breakpoints in Verifier

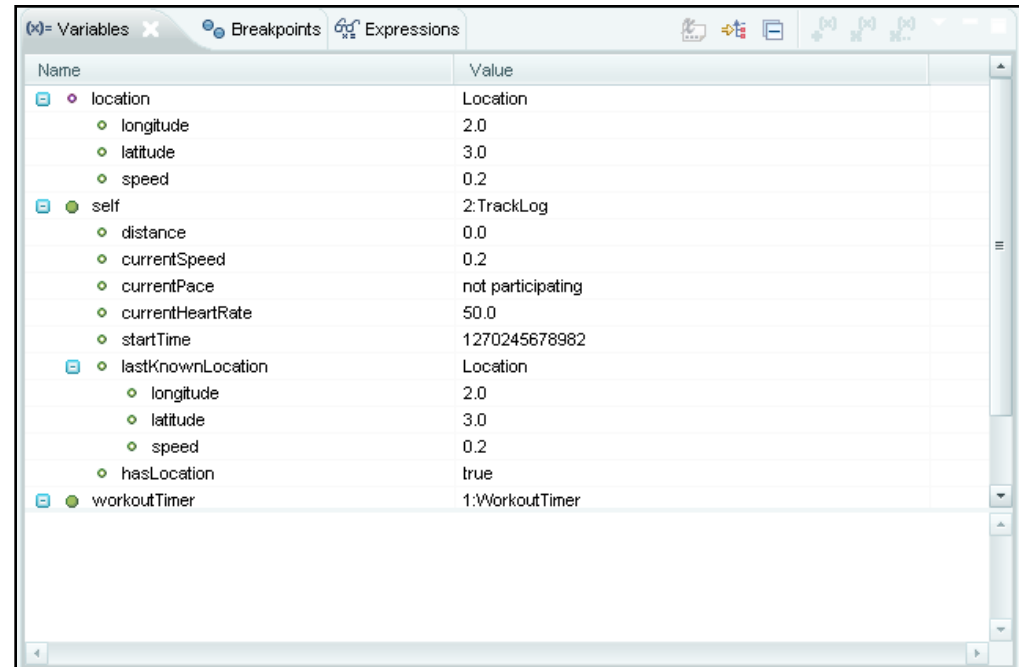
- ◆ Breakpoints can be set on
 - State entry and exit
 - Action language statements
- ◆ Properties are contextual to type of breakpoint; states or action language lines
- ◆ Debug window shows that a breakpoint has been hit, and events that are queued



Variables window

- ◆ Variables window displays:
 - Transient variables
 - Operations
 - Parameters
 - Class instances, attributes also displayed hierarchically

- ◆ List is dynamic and expandable



Name	Value
location	Location
longitude	2.0
latitude	3.0
speed	0.2
self	2:TrackLog
distance	0.0
currentSpeed	0.2
currentPace	not participating
currentHeartRate	50.0
startTime	1270245678982
lastKnownLocation	Location
longitude	2.0
latitude	3.0
speed	0.2
hasLocation	true
workoutTimer	1:WorkoutTimer