



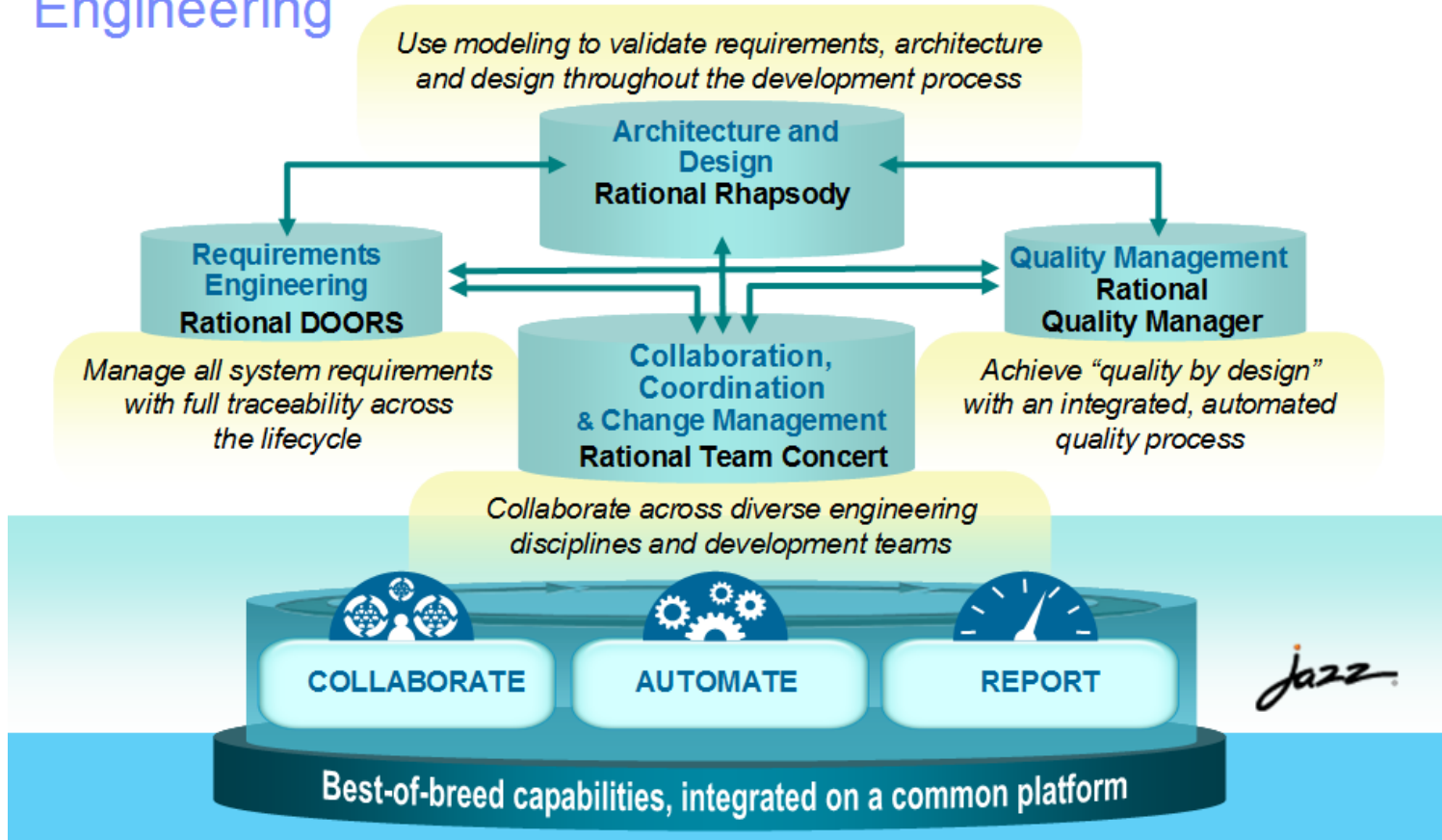
MODEL DRIVEN DEVELOPMENT IN RHAPSODY

Rational software

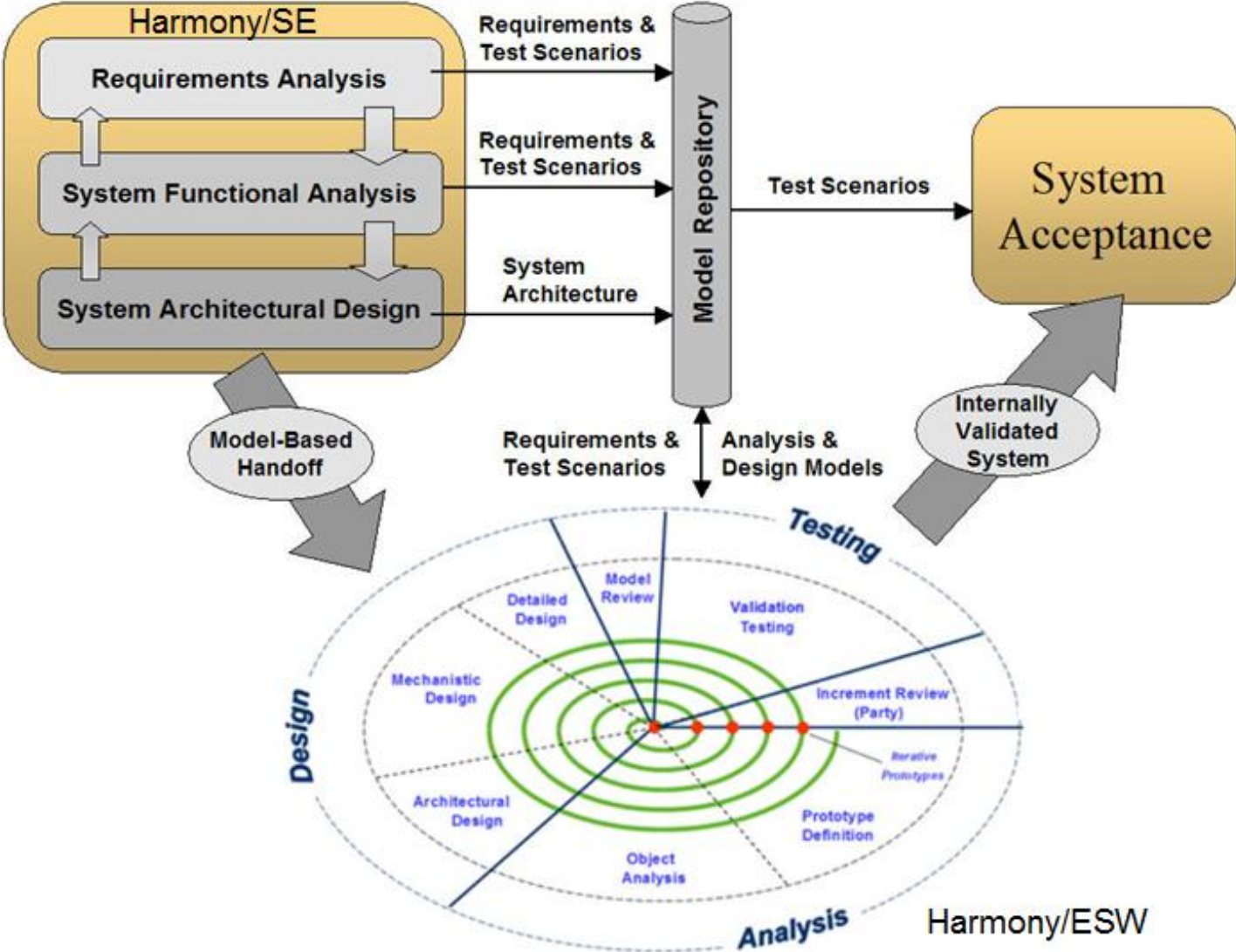
→ **Go to IBM**

Rhapsody & CLM

Rational Solutions for Systems and Software Engineering

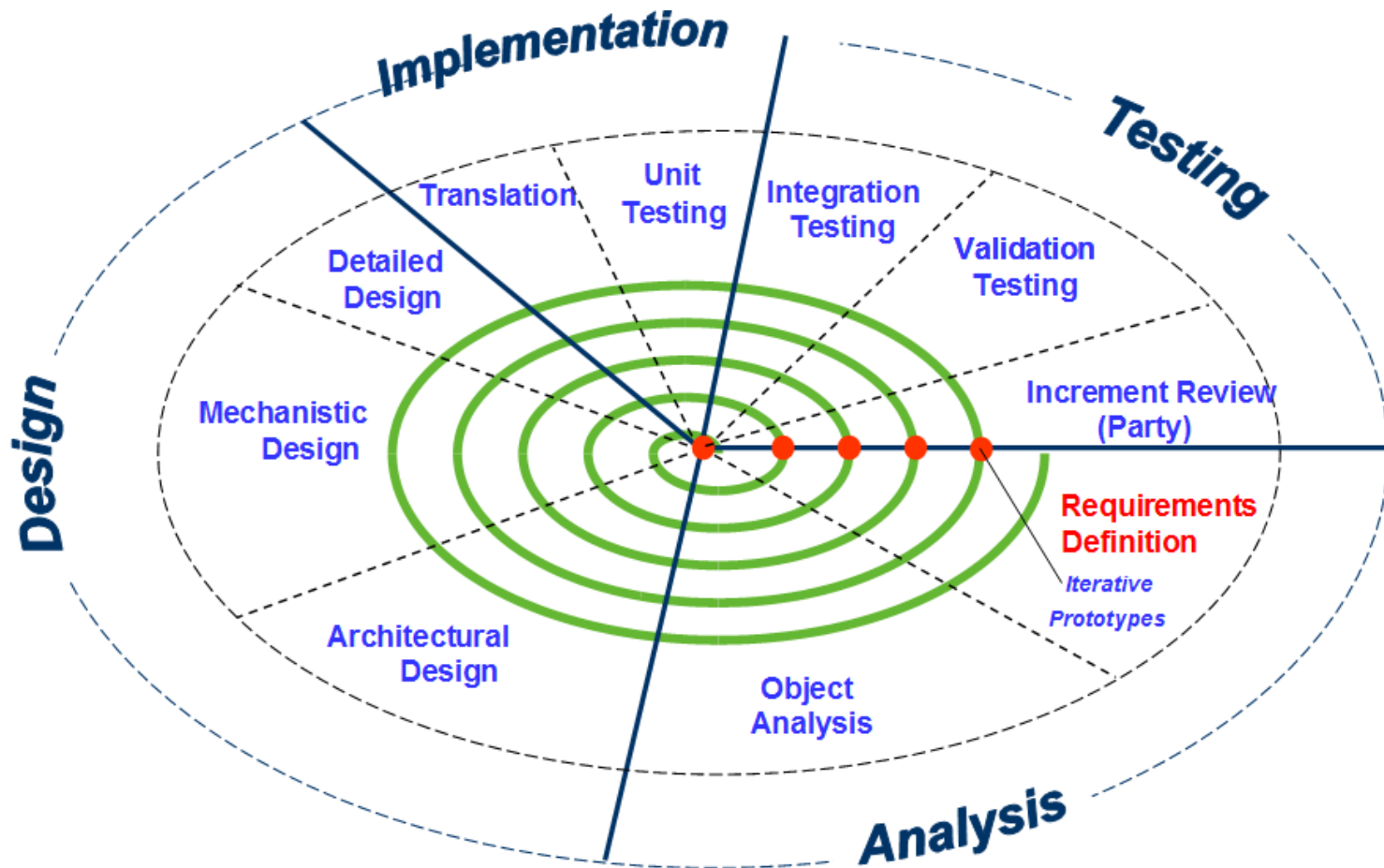


Harmony Process

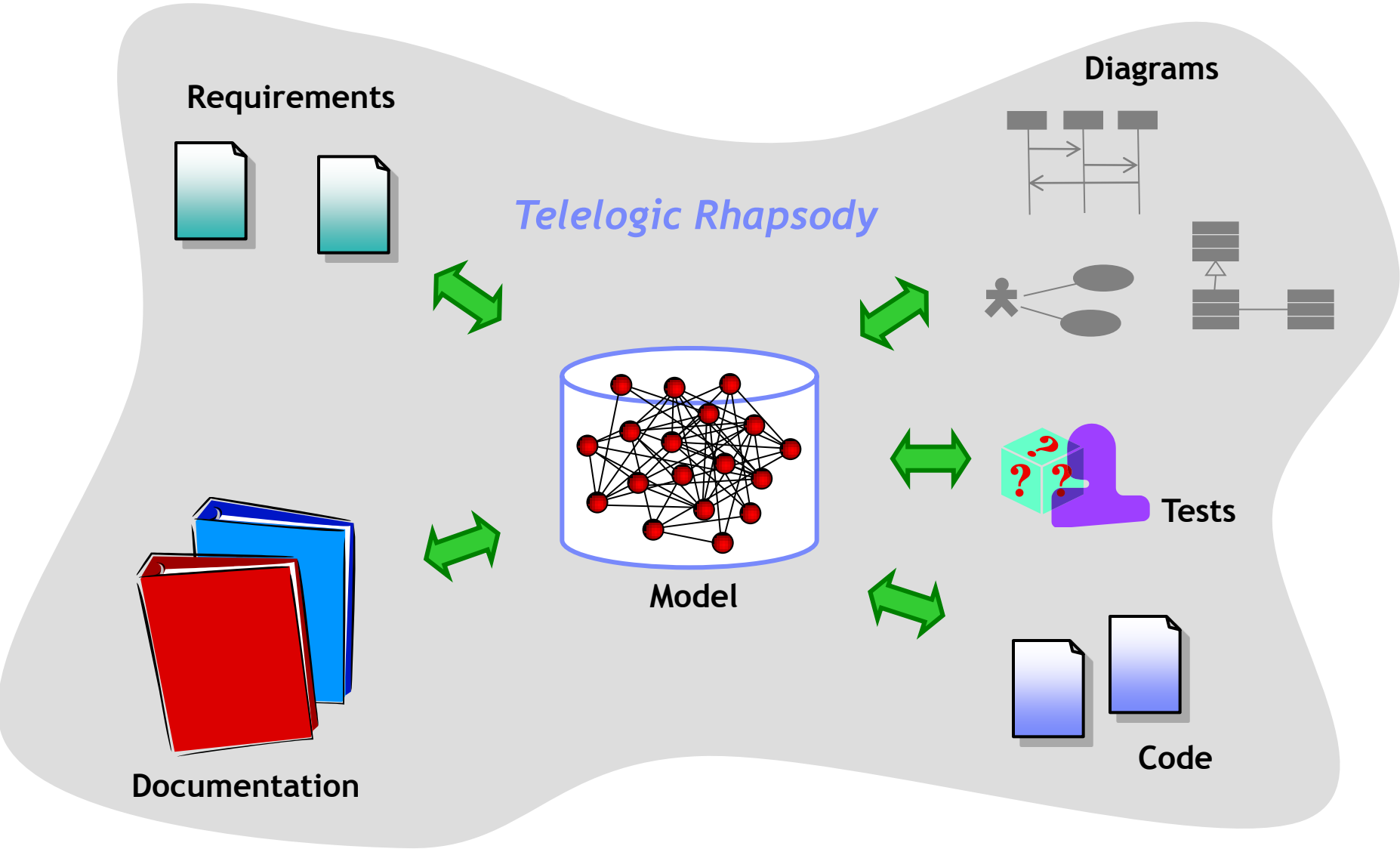


Harmony ESW for MDD

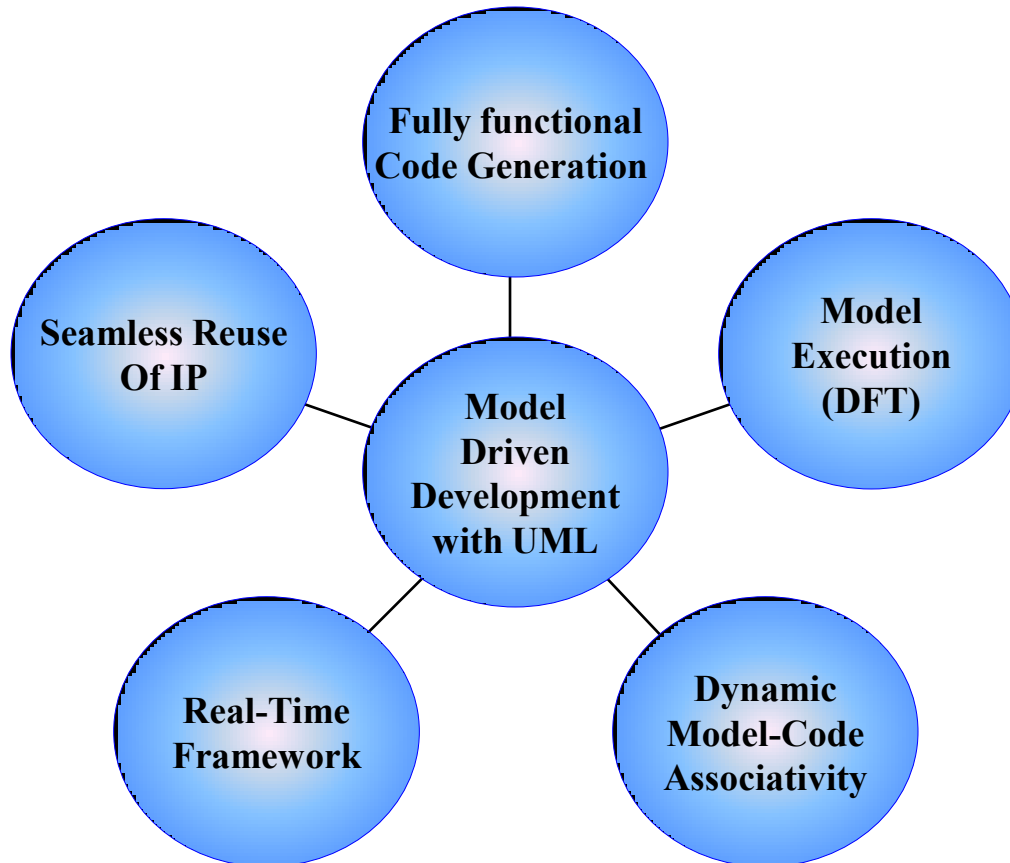
Harmony MDD is an agile based iterative, incremental development process which has a running / deployable prototype at the end of each iteration.



Model-Driven Development

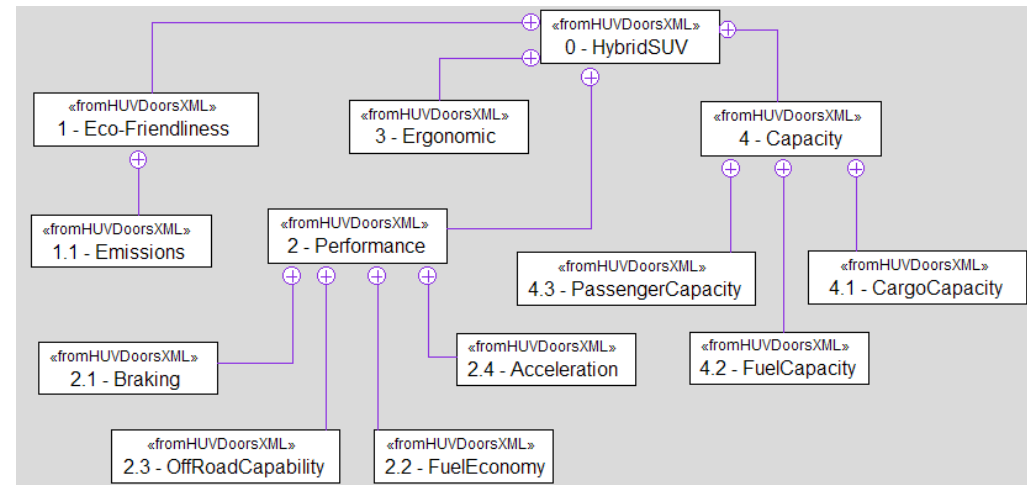


Rhapsody's key enabler



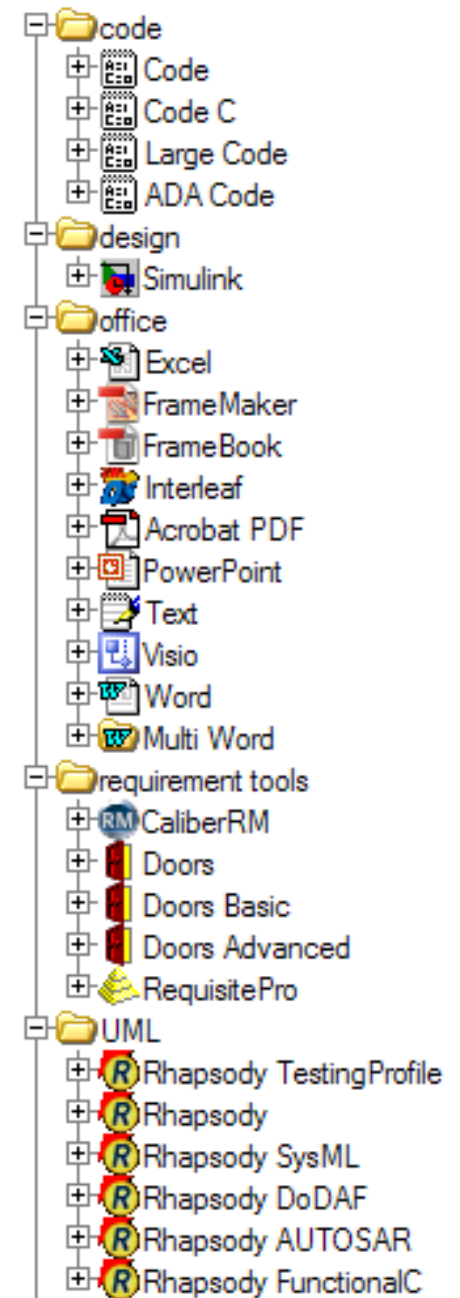
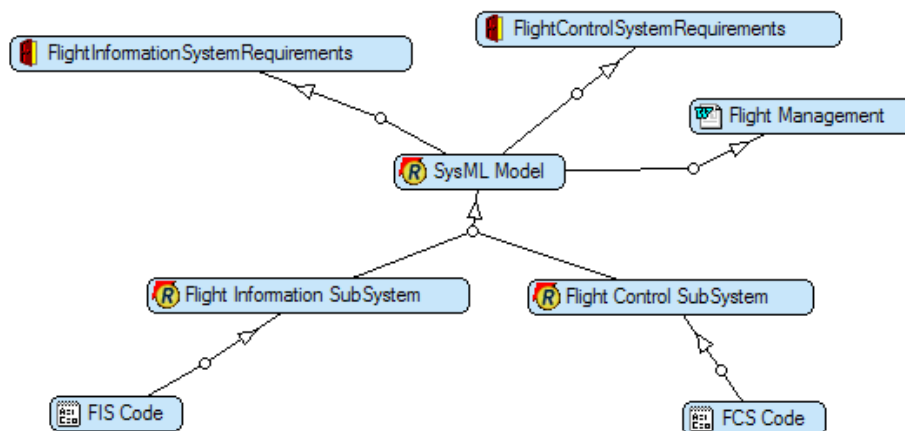
SysML

- SysML is a domain customization of UML 2 for systems engineers
 - ▶ Supports the standard proposal in its latest form (V1.0)
- Support for SysML views
 - ▶ Requirements: [Requirements diagram](#); Use case diagram
 - ▶ Structure: [Block Definition diagram](#); [Internal Block diagram](#)
 - ▶ Behavior: Statechart; Activity diagram; Sequence diagram
 - ▶ Constraints: [Parametric diagram](#)
- *Uniquely Integrated Requirements and Design modeling environment*
- More than just modeling...
 - ▶ Simulation of SysML models
 - ▶ System testing for SysML

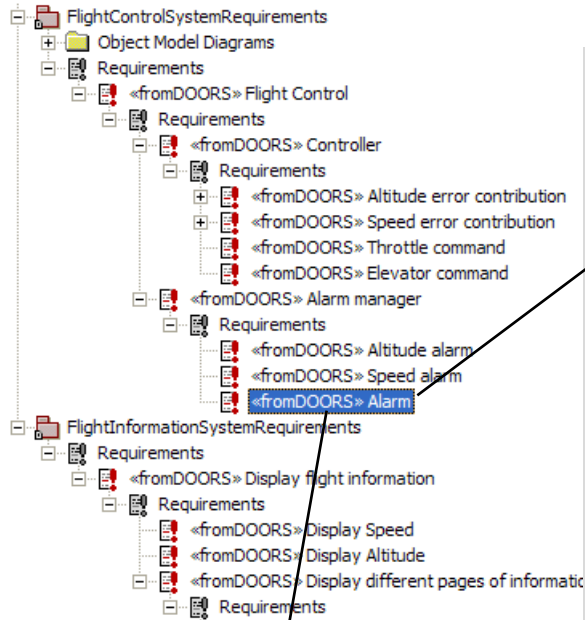


Requirements Modelling

- Requirements Capture
- Requirements Traceability
 - ▶ Create traceability links from model to requirements
 - ▶ Automatic traceability documentation
- Requirements Analysis
 - ▶ Requirement Coverage Analysis
 - ▶ Change Impact analysis
 - ▶ Automatic report generation

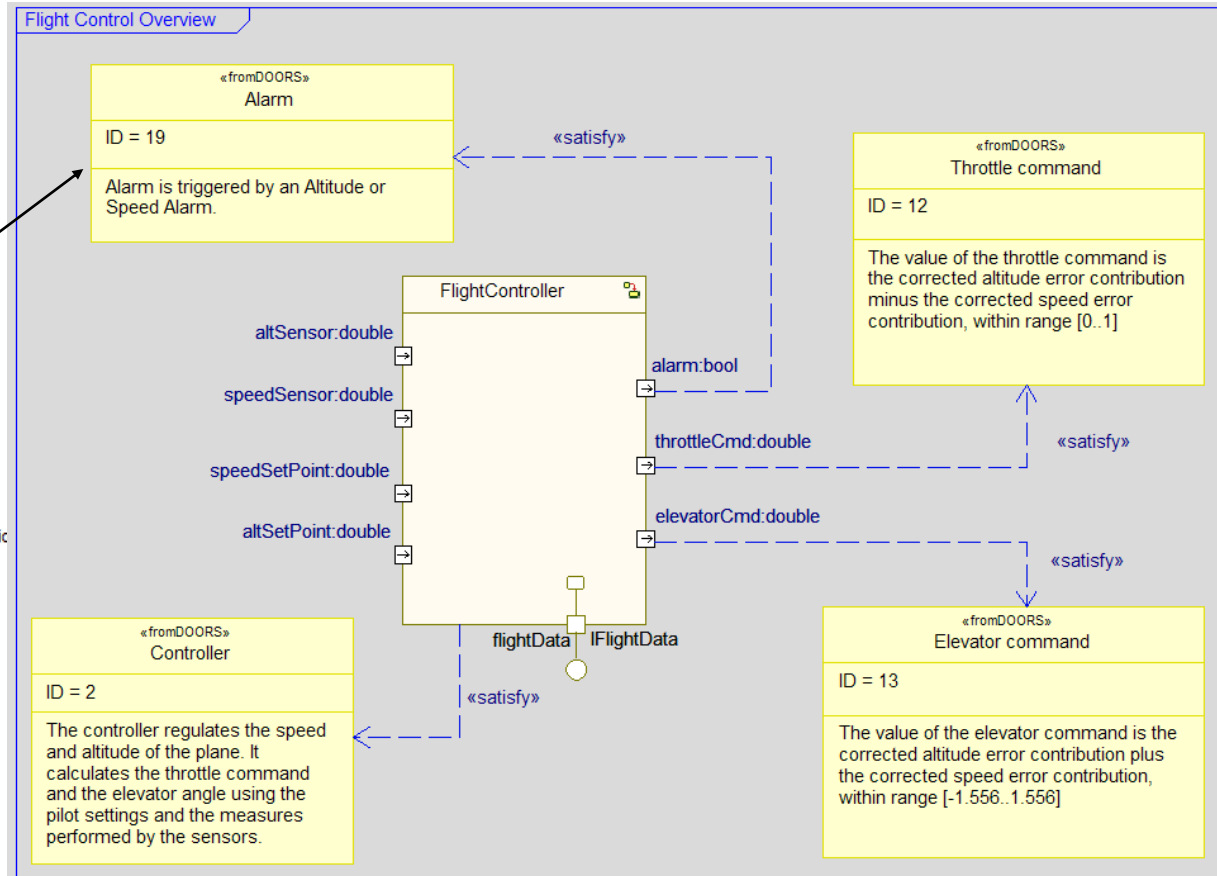


Requirements Capture and Trace



Requirement : Alarm in Alarm manager

General	Description	Relations	Tags	Properties
Name:	Alarm			
Stereotype:	fromDOORS			
Type:	Requirement			
ID:	19			
Defined in:	Alarm manager			
Specification:	Alarm is triggered by an Altitude or Speed Alarm.			



Requirements Coverage Analysis

The screenshot shows the Telelogic Rhapsody Gateway interface for a project named "V71_RiCpp_FlightControl_And_Information_System". The main window is divided into several sections:

- Upstream Coverage Information:** A tree view showing the hierarchy of requirements. The "Alarm" requirement under "FlightControlSystemRequirements" is selected and highlighted in blue.
- Downstream Coverage Information:** A tree view showing the UML Model structure. The "UML Model Rhapsody" is highlighted with a 76% coverage percentage. The structure includes Packages, Classes, and FlowPorts.
- Selection View:** A detailed view of the selected requirement. It shows the requirement text: "Alarm is triggered by an Altitude or Speed Alarm." and its reference attributes.

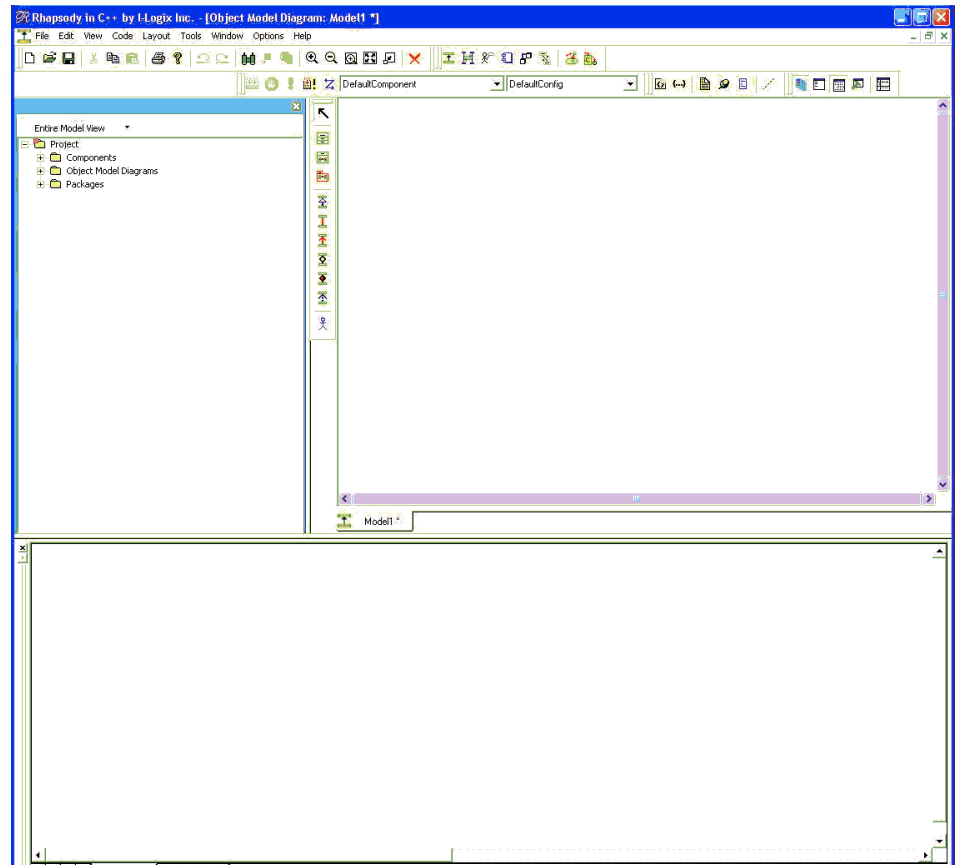
The Selection view is further divided into three panels:

- Upstream:** Text: [Empty], Reference Attributes: [Empty]
- Selection:** Text: "Alarm is triggered by an Altitude or Speed Alarm.", Reference Attributes: [Empty]
- Downstream:** Text: [Empty], Reference Attributes: [Empty]

The status bar at the bottom indicates the current selection path: "FlightControlSystemRequirements DOORS/Flight Control/Alarm manager/Alarm".

Dynamic Model-Code Associativity (DMCA)

- Code is another view of the model
- Dynamic bi-directional synchronization of model & code
- Model & code always in sync
 - Direct reflection of model domain into the code
 - Flexibility without separation of model & code
- Increases productivity



DFT : Executable Models on Host & Target

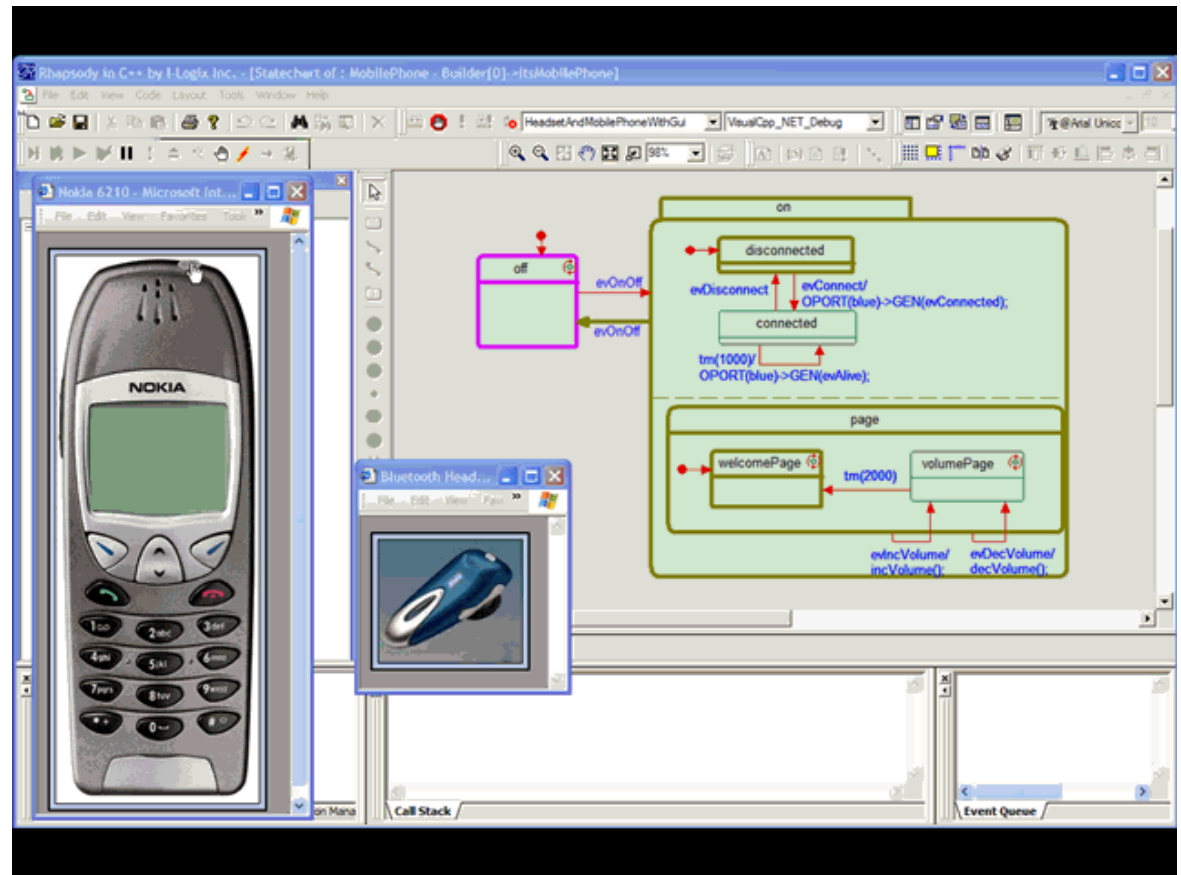
The screenshot displays the Rhapsody in C++ IDE interface. The main workspace is divided into several panes:

- Entire Model View (Left):** A tree view showing the model structure, including packages like CommandDown, CommandLeft, CommandRight, and CommandUp, and their sub-elements like Hyperlinks, Instances, Operations, and SuperClasses.
- Sequence Diagram: Animated Normal Operation (Center):** A UML sequence diagram with participants: Builder, Reader, Command, CommandList, CommandListGo, and Writer. The diagram shows a sequence of messages between these objects.
- Statechart of : Builder - Builder[0] (Right):** A statechart for the Builder object. It features a state named "running" with an internal transition "count++;". A timer "tm(500)" is associated with the state. Transitions from the running state include: a guard "[rand()%2]==0" leading to an event "evWrite to itsWriter", and an "else" guard leading to an event "evRead to itsReader".
- Features of Builder[0] (Bottom Right):** A dialog box showing the instance name "Builder[0]". It lists attributes: "count" with value "0" and type "int". It also shows relations: "itsCommandList", "itsReader", and "itsWriter".

At the bottom of the IDE, there are panels for "Call Stack", "Event Queue", and "Executable is Idle". The status bar at the very bottom indicates "GE MODE" and the date/time "Thu, 1, Mar 2007 6:21 PM".

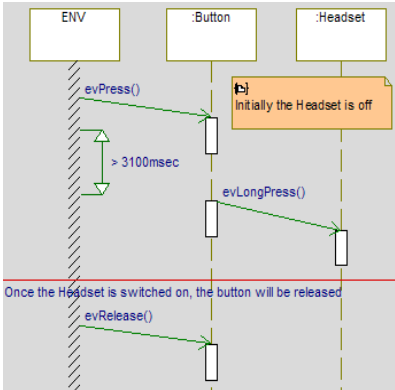
DFT: Collaborative Debugging

- Web based collaborative debugging
- Rapid prototyping

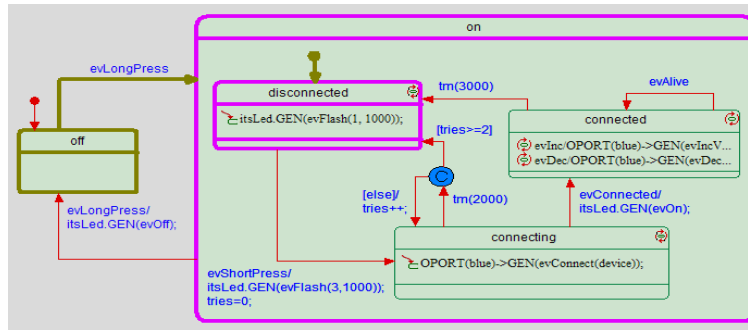


DFT: Test Conductor™

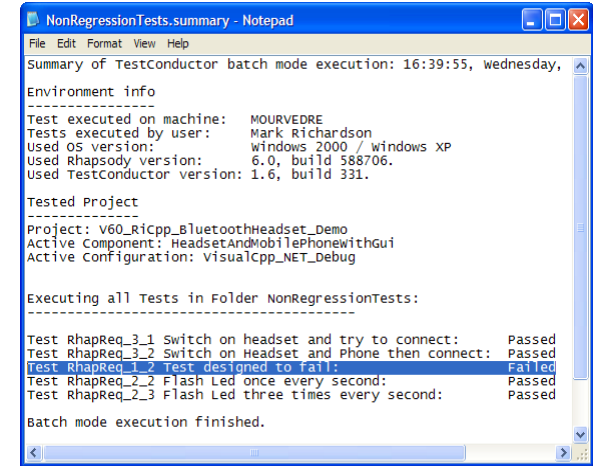
Sequence Diagrams



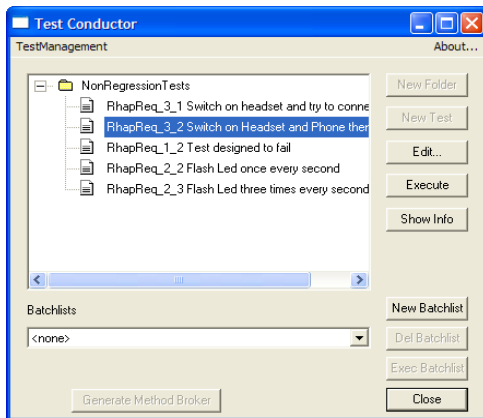
Stimulate & Monitor the Model



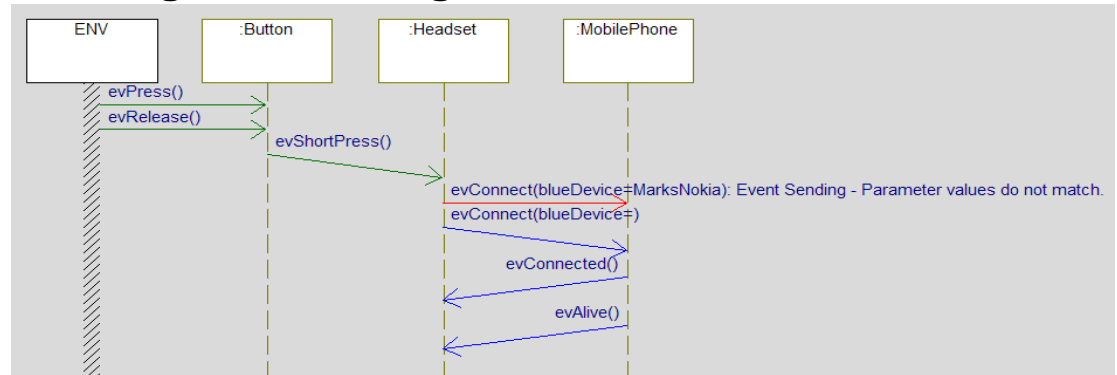
Test Results

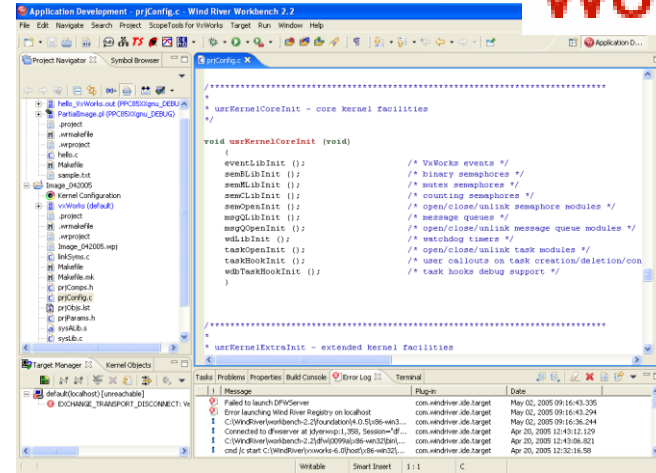
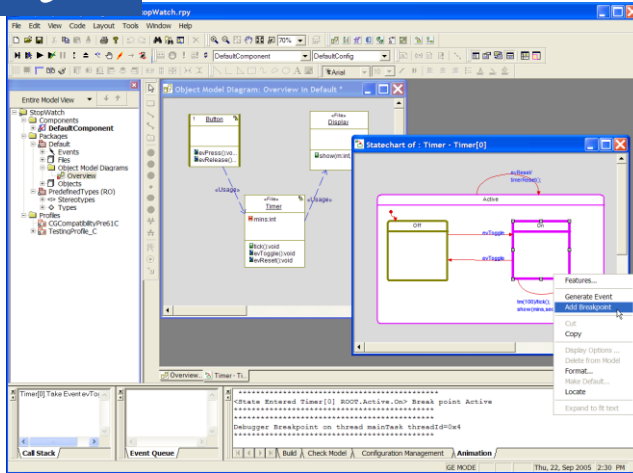


Test Configuration



Finding & Correcting Errors





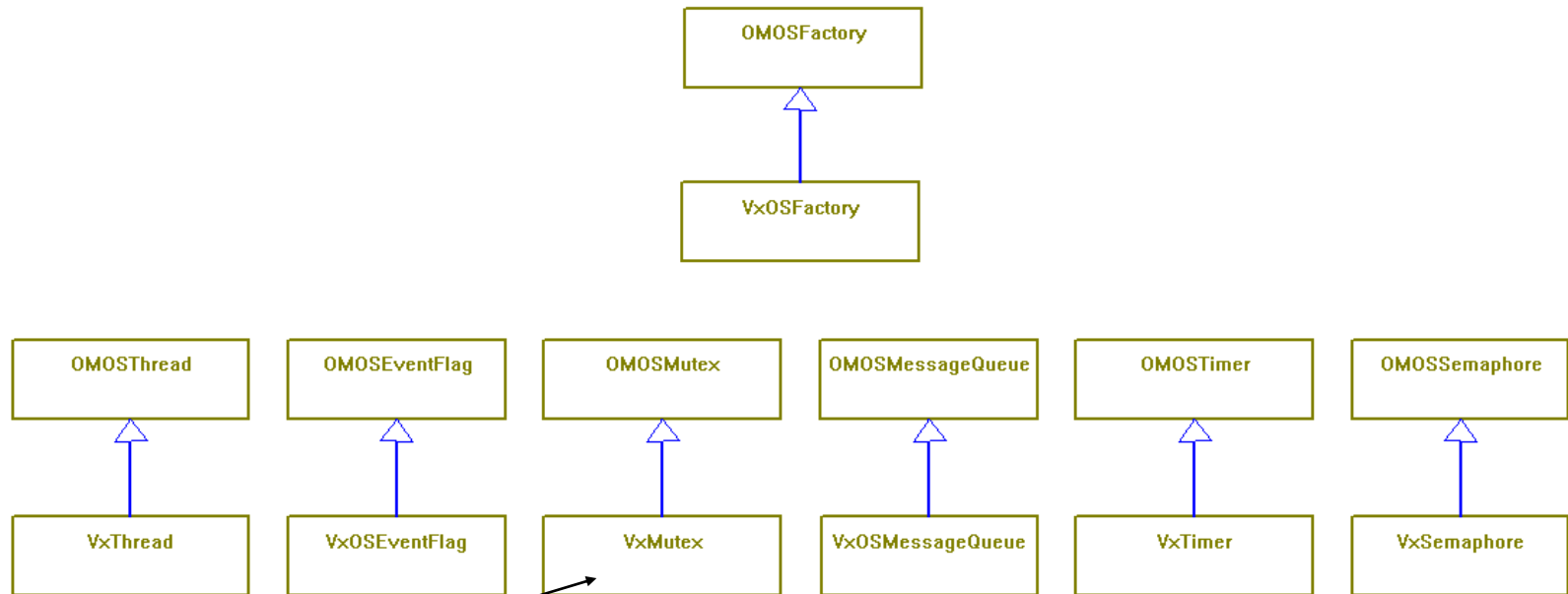
MDD
Code Generation C, C++ Ada
Combined source and Design-
level debugging

Automatic Download
Synchronized Breakpoints
Unit Testing

Targets



VxWorks Adapter

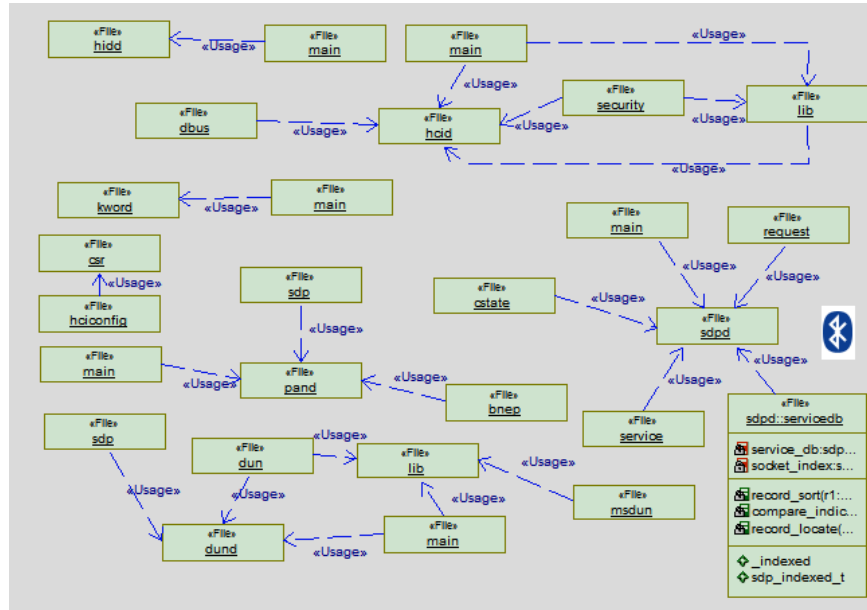
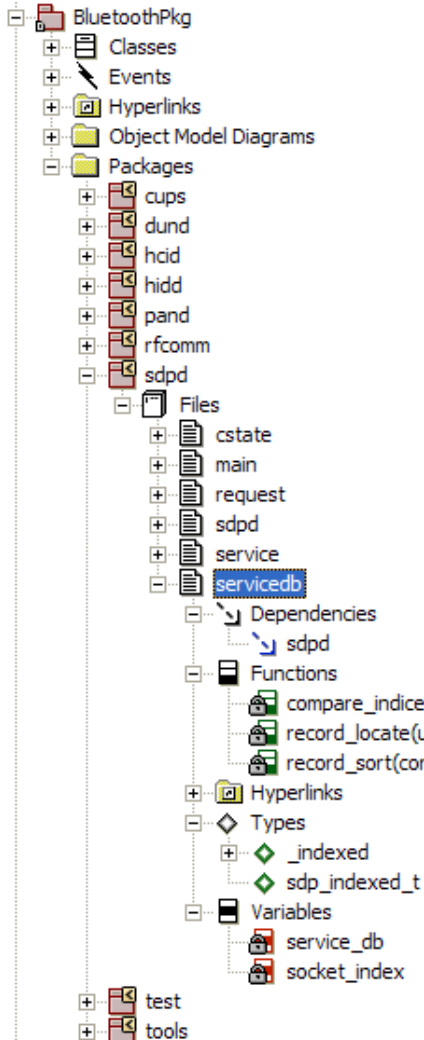


```
class VxMutex : public OMOSMutex {
private:
    SEM_ID hMutex;
public:
    void lock() { semTake(hMutex, WAIT_FOREVER); }
    void unlock() { semGive(hMutex); }
    VxMutex() {
        hMutex = semMCreate(SEM_Q_FIFO);
    }

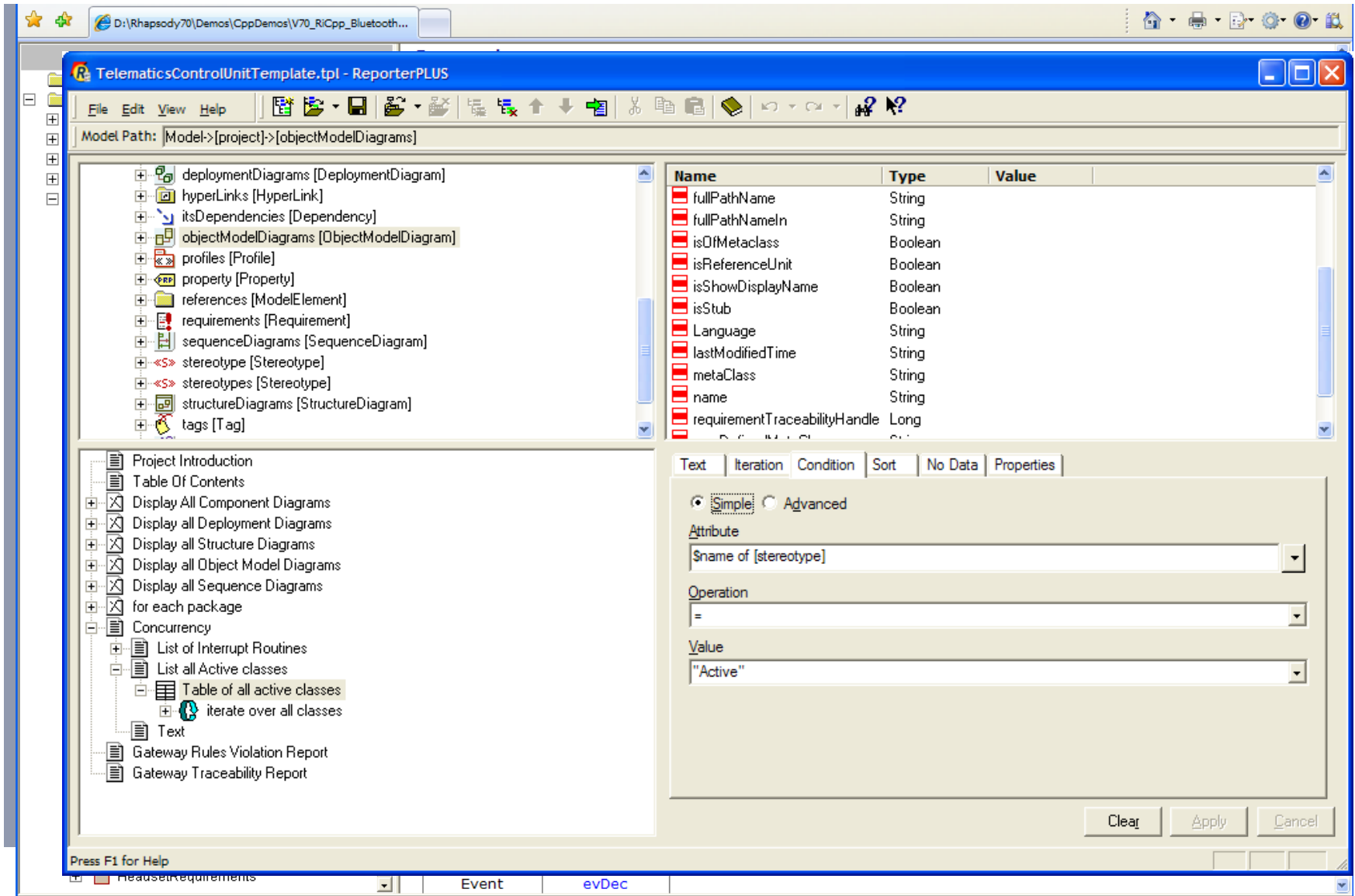
    ~VxMutex() {semDelete(hMutex);}

    void* getHandle() { return (void *)hMutex; }
    virtual void* getOsHandle() const { return (void*) hMutex; }
};
```


Reuse of IP : Import Legacy "C" code



Documentation: Rhapsody ReporterPLUS™



MENU