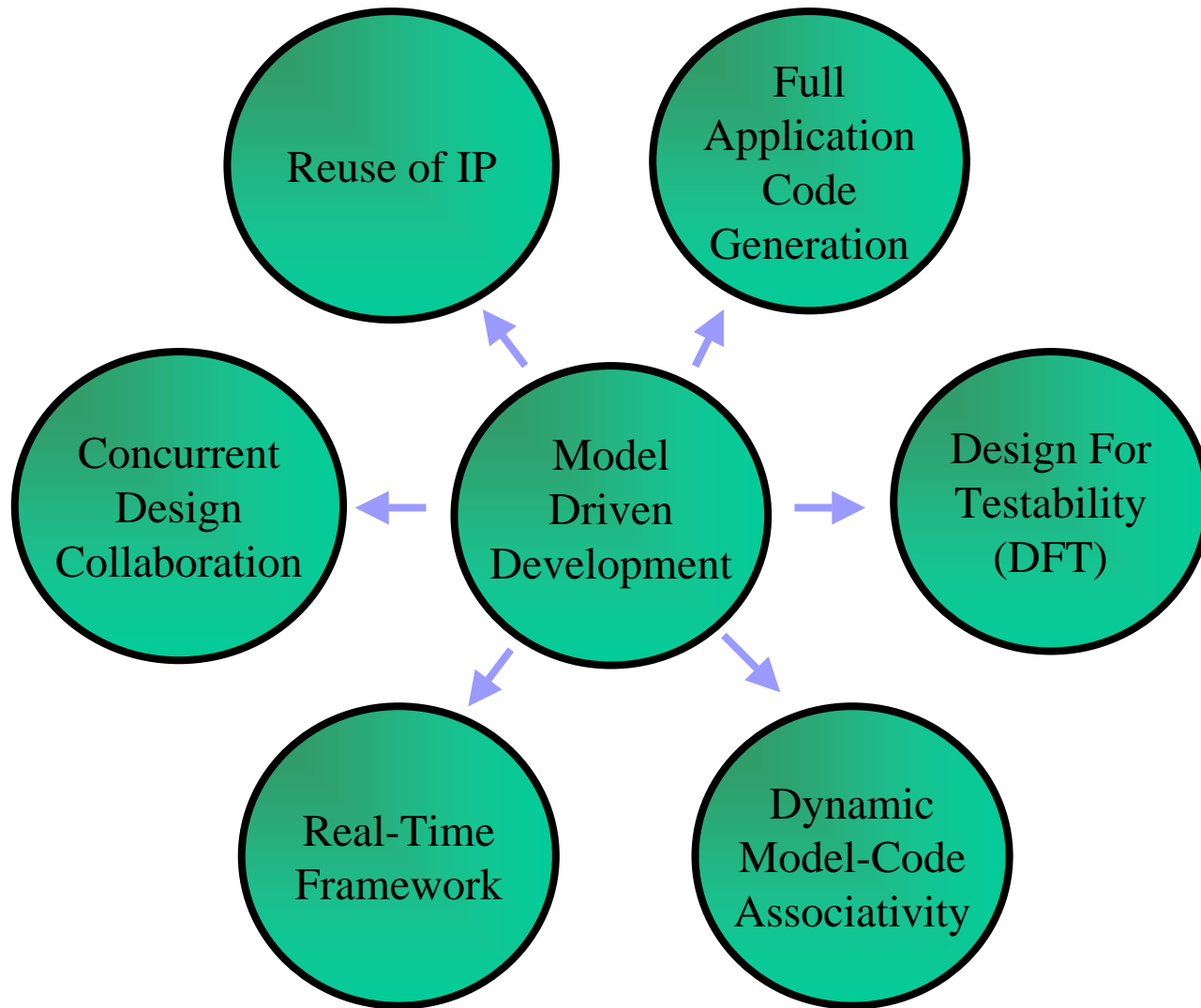


Model Driven Development



Full Application Code Generation

- ❑ Rhapsody leverages *all* structural and behavioral model views to produce an executable application
 - Structure models
 - State charts: event driven behavior
 - Activity graphs: algorithms and process flows
 - Components and artifacts

- ❑ Rhapsody generates very clean, readable code, easily debugged through any commercial IDE
 - Integrated “white-box” Code (C, C++, Java, Ada, IDL) generation
 - High productivity; low cost of maintenance

- ❑ Rhapsody generates all application construction artifacts to provide an integrated build environment

- ❑ Comprehensive code generation technologies
 - OO based and / or functional based
 - Stereotype based
 - Rules based

Rule Based Ada Code Generation

- ❑ I-Logix offers full code generation for specific needs/environments, such as
 - ❑ GNAT
 - ❑ Greenhills
 - ❑ Aonix
- ❑ End-Users can generate their own variants of code
 - ❑ Offers Full control over the generated code
 - ❑ Addresses the various code standards and needs of Mil/Aero
 - ❑ End user experts can modify the rules to address particular needs
 - e.g Code indentation, file naming conventions, case vs. if statements
 - ❑ New rules can be used by all team members
 - As simple as a selection from a pull-down menu

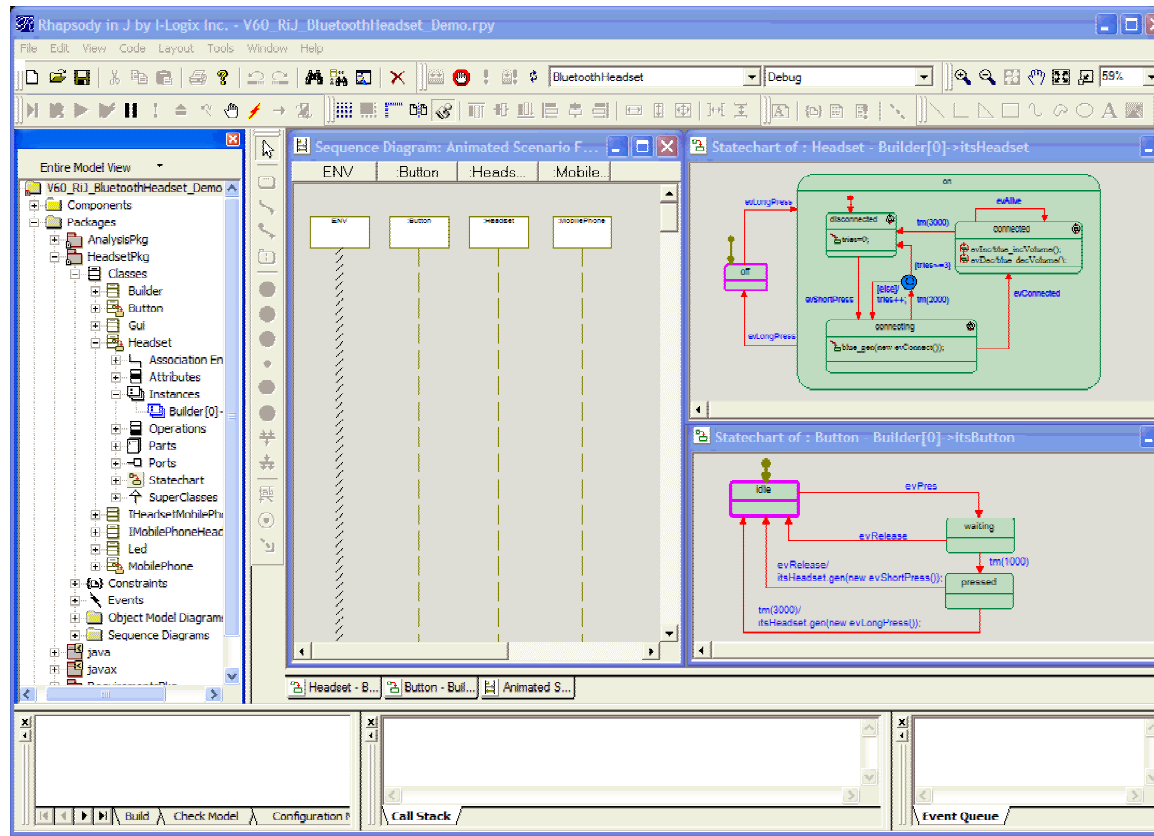
Code Generation Technologies

- ❑ State of the art code generation technology
 - ❑ “Hard-coded” code generation
 - ❑ Template based code generation
 - ❑ Stereotype based code generation
 - ❑ Rules based code generation
 - ❑ Execution environment includes:
 - ❑ On the fly code generation from a UML model
 - ❑ WYSIWYG editor to ease rules editing
 - ❑ Rules, macros and scripts to control dynamic content
 - ❑ Rules tracer to analyze and debug rules execution
 - ❑ Built-in rules diff&merge
- Low flexibility*
- High flexibility*

DFT: Executable Models

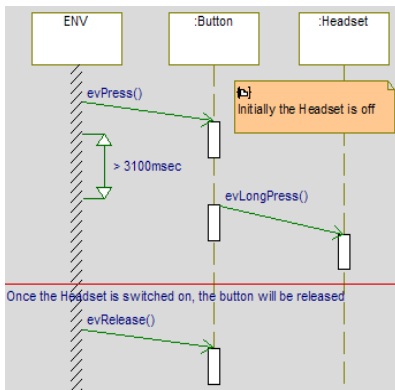
You can't test what you can't execute!

- ❑ Rapid execution *at the design level* on host or even target
- ❑ The best way to avoid having bugs is to not introduce them to the system

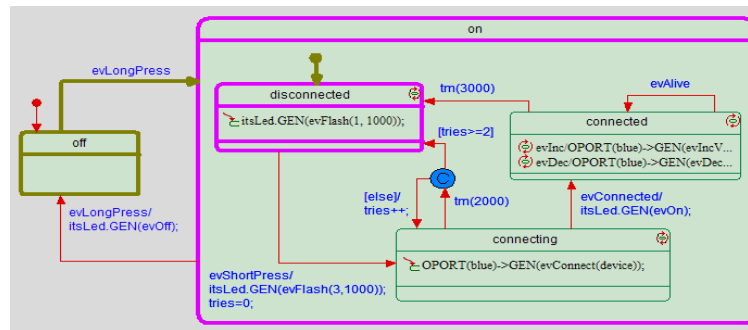


DFT: Test Conductor™

Sequence Diagrams



Stimulate & Monitor the Model



Test Results

```

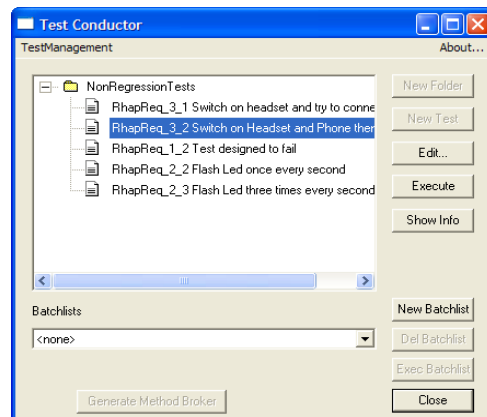
NonRegressionTests.summary - Notepad
File Edit Format View Help
Summary of TestConductor batch mode execution: 16:39:55, wednesday,
Environment info
Test executed on machine: MOURVEDRE
Tests executed by user: Mark Richardson
Used OS version: windows 2000 / windows XP
Used Rhapsody version: 6.0, build 588706.
Used TestConductor version: 1.6, build 331.

Tested Project
Project: V60_Ricpp_BluetoothHeadset_Demo
Active Component: HeadsetAndMobilePhoneWithGui
Active Configuration: VisualCpp_NET_Debug

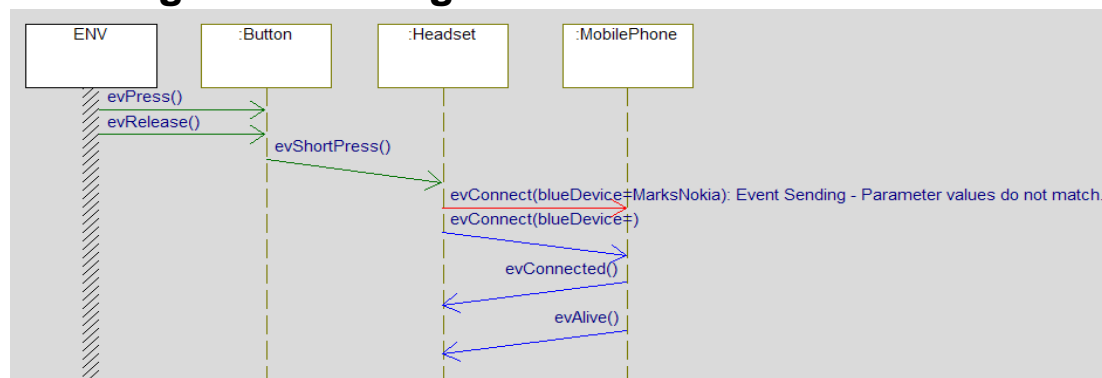
Executing all Tests in Folder NonRegressionTests:
-----
Test RhapReq_3_1 Switch on headset and try to connect: Passed
Test RhapReq_3_2 Switch on headset and Phone then connect: Passed
Test RhapReq_1_2 Test designed to fail: Failed
Test RhapReq_2_2 Flash Led once every second: Passed
Test RhapReq_2_3 Flash Led three times every second: Passed

Batch mode execution finished.
  
```

Test Configuration



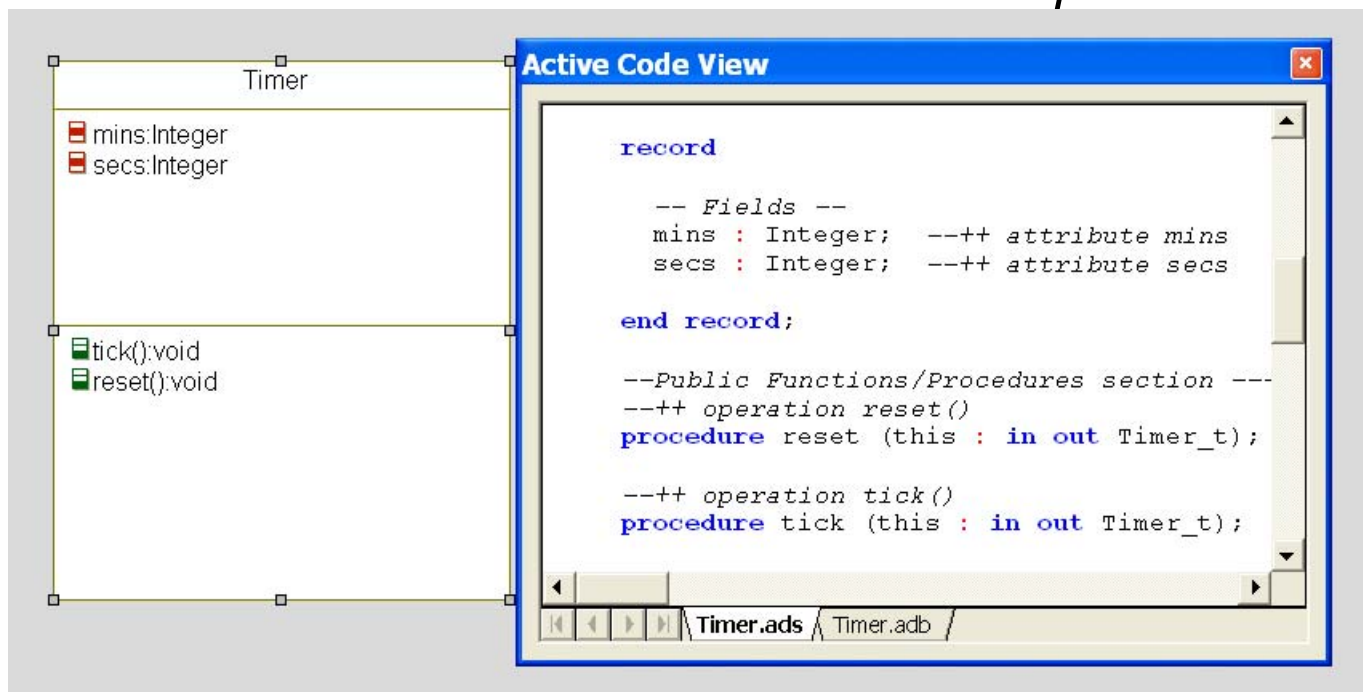
Finding & Correcting Errors



Dynamic Model Code Associativity

Rhapsody works the way you do

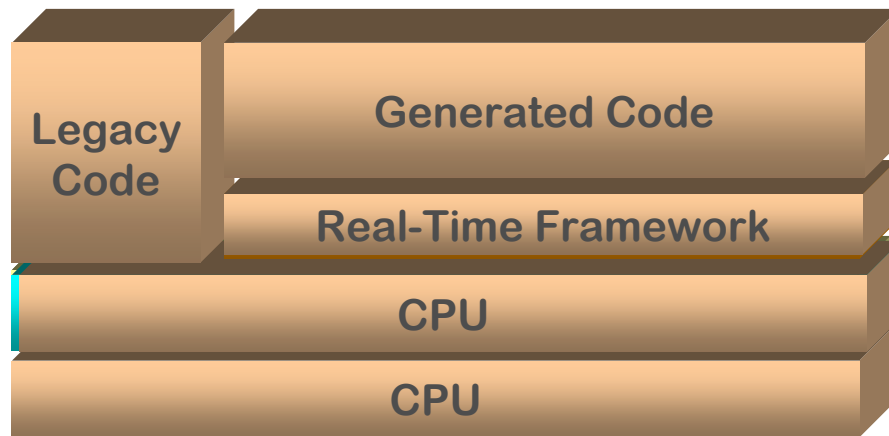
- ❑ Design, Code and Documentation are always kept in sync
- ❑ Freedom to work at code level or design level
- ❑ Change one view, the others **change automatically**
- ❑ *Critical for real-time embedded software development*



Real-Time Frameworks

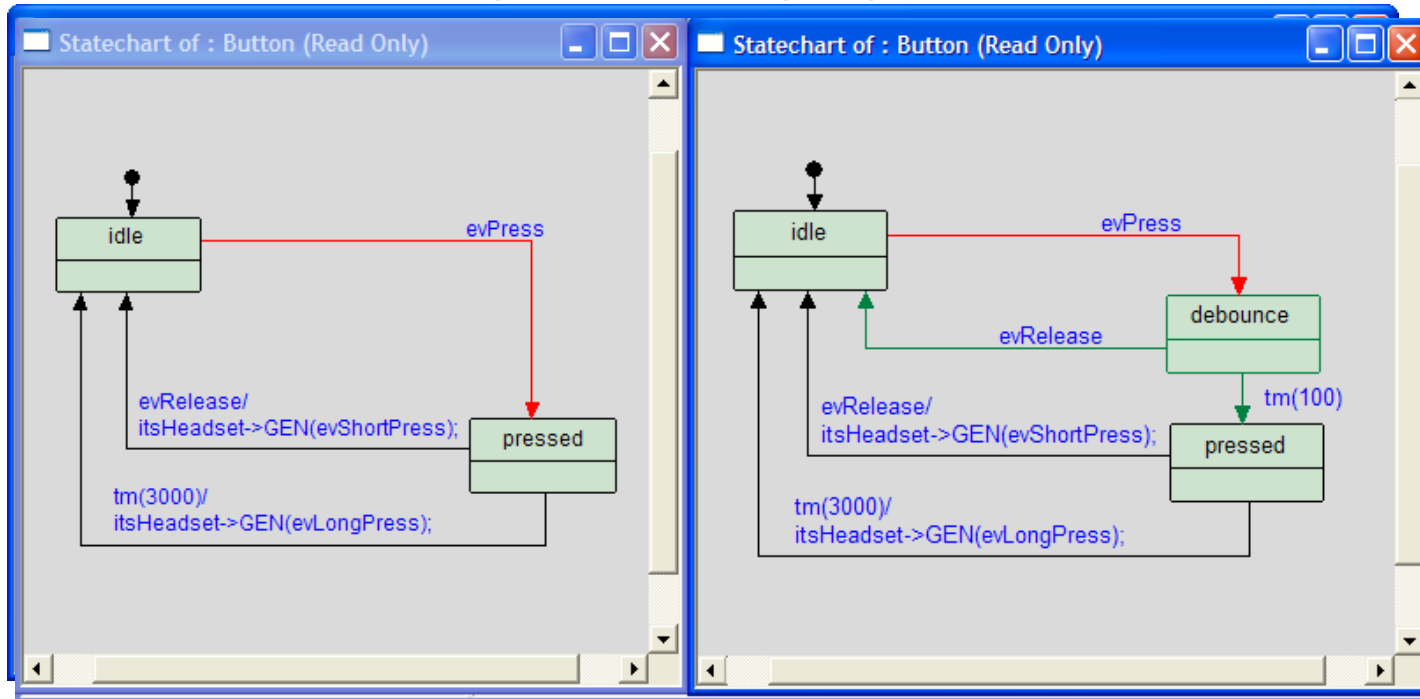
Rhapsody provides an executable real-time framework

- ❑ Most applications are over 50% “housekeeping code” which is redeveloped every time you create a system
- ❑ A *framework* is a partially completed application
 - ❑ **you** customize and specialize for **your** application
- ❑ A *real-time framework* is an
 - ❑ integrated set of design patterns
 - ❑ optimized for embedded applications



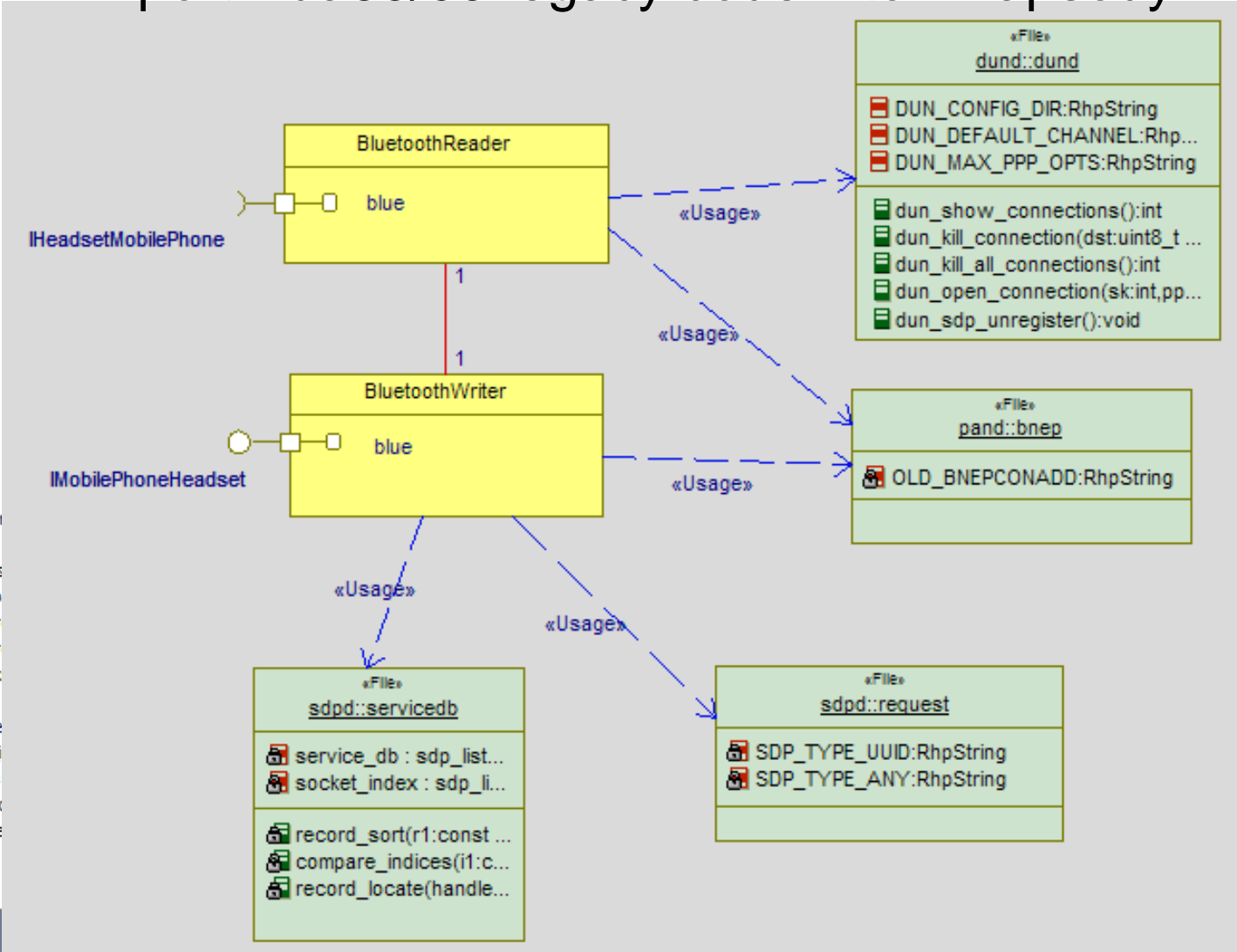
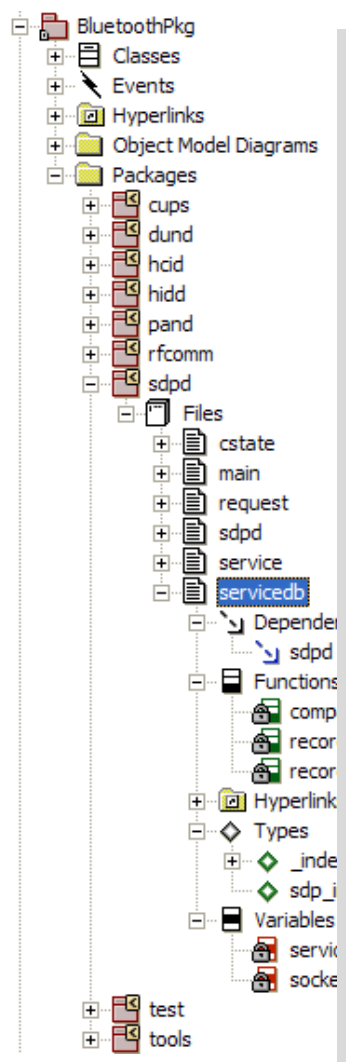
Concurrent Design Collaboration

- ❑ Small and Large Scale Development
- ❑ Tight integration with configuration management
- ❑ Partial loading
- ❑ Visual Differencing and Merging



Reuse of IP : Import Legacy Ada code

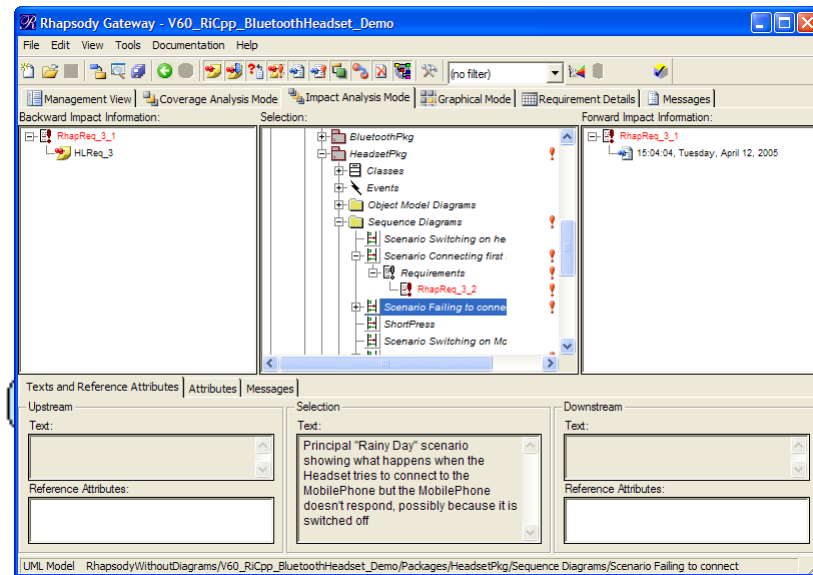
Import Ada83/95 legacy code into Rhapsody



Rhapsody® Gateway

- ❑ Extending Rhapsody's existing RM interface capabilities with the introduction of the Rhapsody® Gateway (RG) :
 - ❑ Allows Rhapsody users to seamlessly work with Rhapsody and classic 3rd Party Requirements Management tools *and other common requirements authoring tools*.
 - ❑ Enables traceability analysis and reporting from within Rhapsody.
 - ❑ Complete and constant synchronization between UML and Req Tool Views

- ❑ Telelogic DOORS
- ❑ UGS Slate
- ❑ Microsoft Word/ Excel
- ❑ XML
- ❑ Adobe PDF Acrobat Files
- ❑ Generic Text files
- ❑ Code
- ❑ IBM Requisite Pro
- ❑ UGS TeamCenter Requirements (Future)
- ❑ Serena Software RTM (Future)



Documentation: Rhapsody ReporterPLUS™

I-Logix

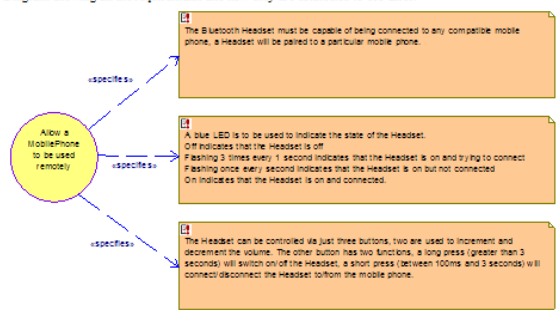
4. Package : RequirementsPkg

Package containing requirements that have been imported from DOORS via the Gateway.

Name	Description	Owner	Status	Andors
HLReq_1	The Bluetooth Headset must be capable of being connected to any compatible mobile phone, a Headset will be paired to a particular mobile phone.	Andy	10%	Requirement: RhapReq_1_2
HLReq_2	A blue LED is to be used to indicate the state of the Headset. Off indicates that the Headset is off Flashing 3 times every 1 second indicates that the Headset is on and trying to connect Flashing once every second indicates that the Headset is on but not connected On indicates that the Headset is on and connected.	Mark	Complete	Requirement: RhapReq_2_2 Requirement: RhapReq_2_3
HLReq_3	The Headset can be controlled via just three buttons, two are used to increment and decrement the volume. The other button has two functions, a long press (greater than 3 seconds) will switch on/off the Headset, a short press (between 100ms and 3 seconds) will connect/disconnect the Headset to/from the mobile phone.	Mark	Complete	Requirement: RhapReq_3_1 Requirement: RhapReq_3_2

4.1 Object Model Diagram : Requirements Taxonomy

Diagram showing all the requirements and how they are connected to use cases.



Document generated using "RhapsodyPackageOrientedProjectReport.tpl" Template

I-Logix

5. Gateway Traceability Matrix

5.1 UML Model covers RequirementsPkg

Covering ratio: 100%

	Upstream	Downstream
HLReq_1		RhapReq_1_2
HLReq_2		RhapReq_2_2
HLReq_2		RhapReq_2_3
HLReq_3		RhapReq_3_2
HLReq_3		RhapReq_3_1

5.2 TestConductor Tests covers UML Model

Covering ratio: 100%

	Upstream	Downstream
RhapReq_1_2		073650, Tuesday, April 26, 2005
RhapReq_2_2		073650, Tuesday, April 26, 2005
RhapReq_2_3		073650, Tuesday, April 26, 2005
RhapReq_3_1		073650, Tuesday, April 26, 2005
RhapReq_3_2		073650, Tuesday, April 26, 2005

Document generated using "RhapsodyPackageOrientedProjectReport.tpl" Template