IBM Software Group

Rational Rhapsody

Model-driven development for systems design and software development of technical, realtime or embedded systems

Rational. software

Go to IBM

Systems design and software development dilemmas

- Need to communicate effectively with customers, suppliers and multidiscipline project members
- Ability to effectively collaborate within large teams, often distributed around the globe
- Effectively managing ever-changing and evolving requirements
- Deal effectively with complexity of the problem domain
- Spending more than half the development time for integration and testing
- Shrinking development schedules
- Parallel development
- Increasing complexity and quality
- Reducing functionality in order to meet deadlines
- Fixed resources
- Pressure to lower costs

Key Challenges Facing the Systems Marketplace

- *Increasing complexity and accelerating changes*
- *Uncovering defects late in development cycle*
- *Reducing functionality to meet time to market pressure*
- *Compliance with key engineering and design processes*
- *Effective collaboration of geographically distributed teams*
- *Building more innovative products to grow the business*

Aerospace & Defense

Consumer Electronics

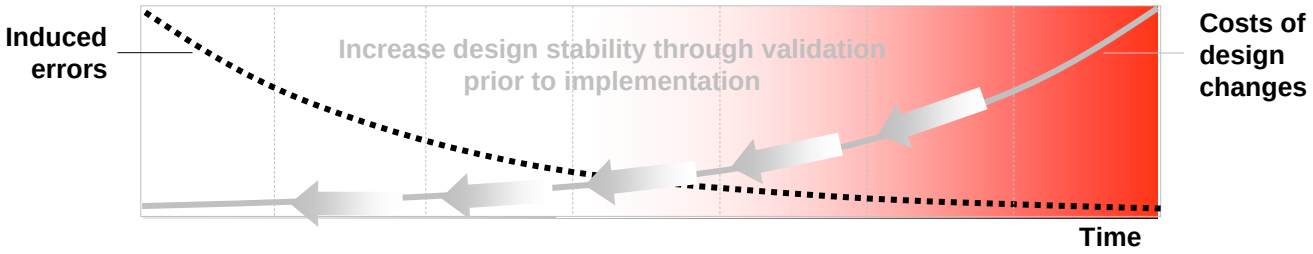
Medical Devices

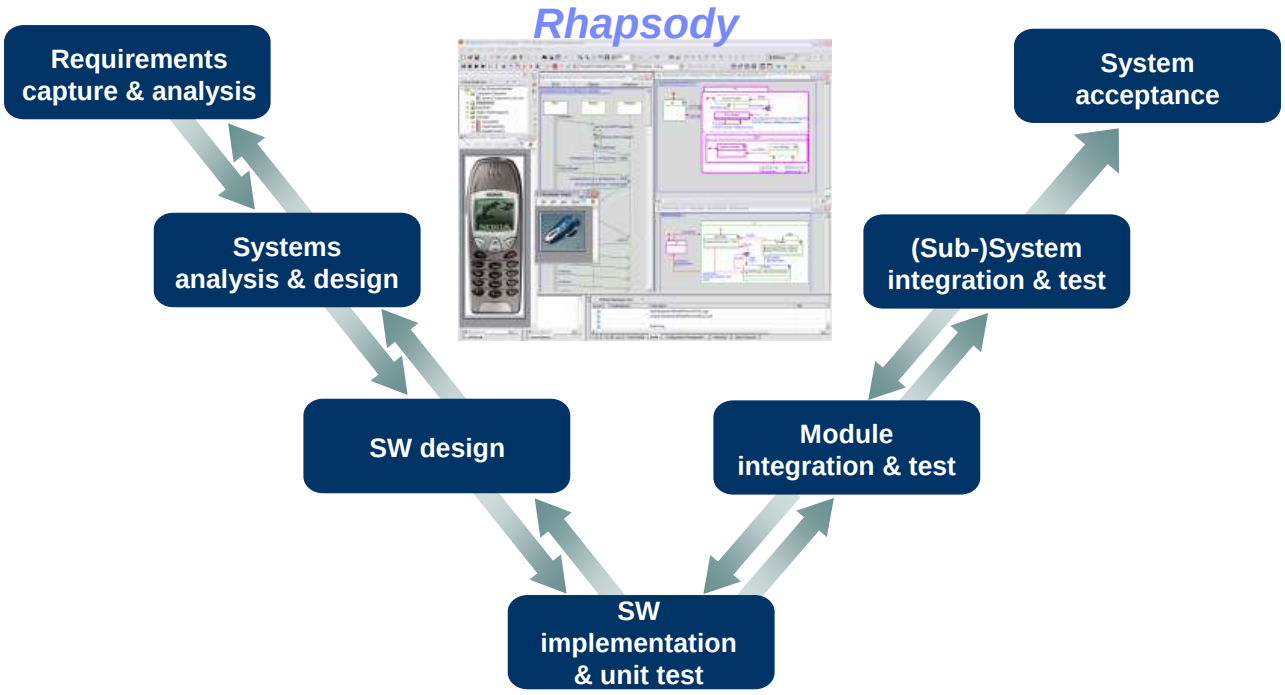
Automotive

Telecom/ Datacom

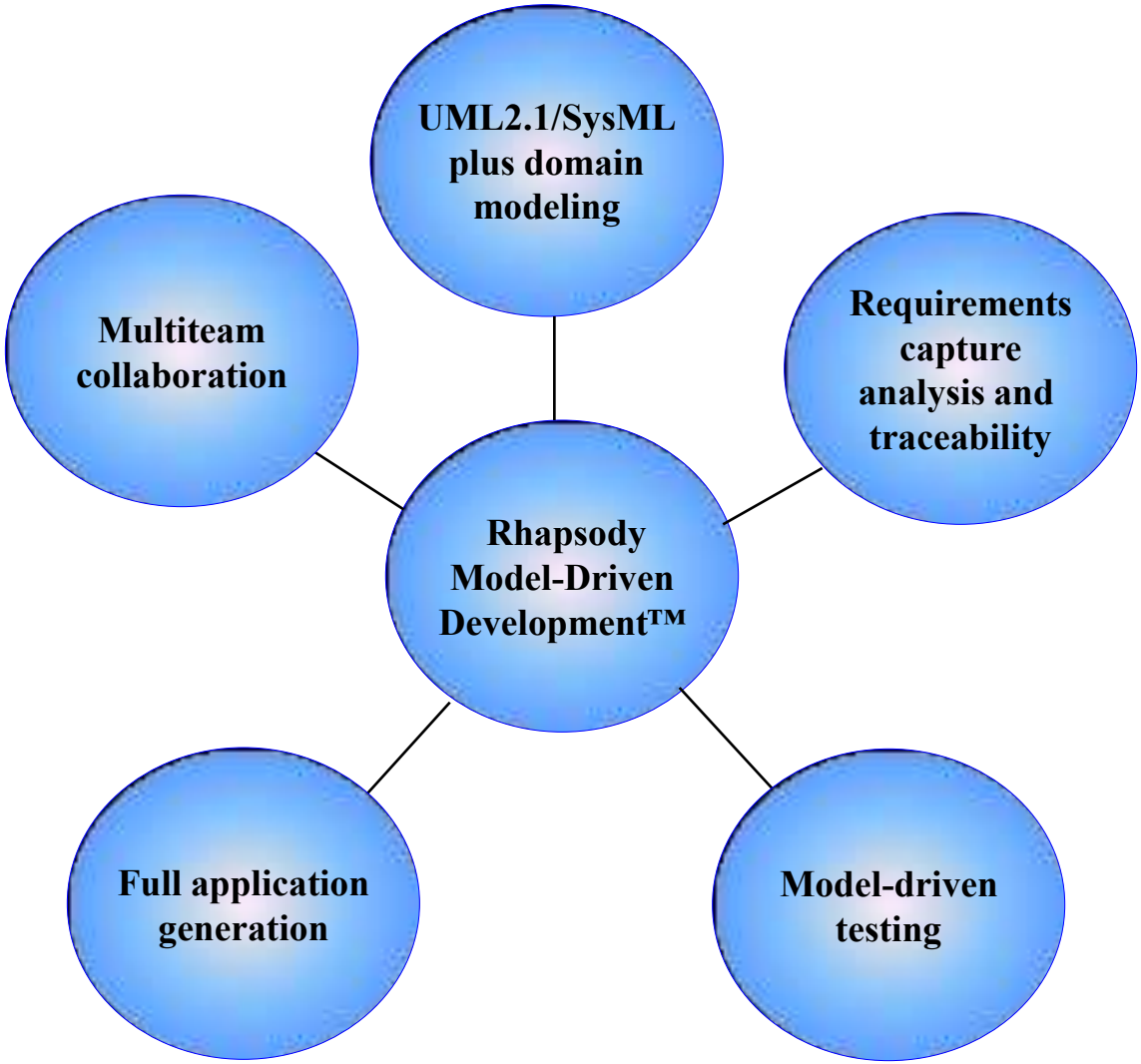
3

Simulation and execution of models support elimination of errors early in the process

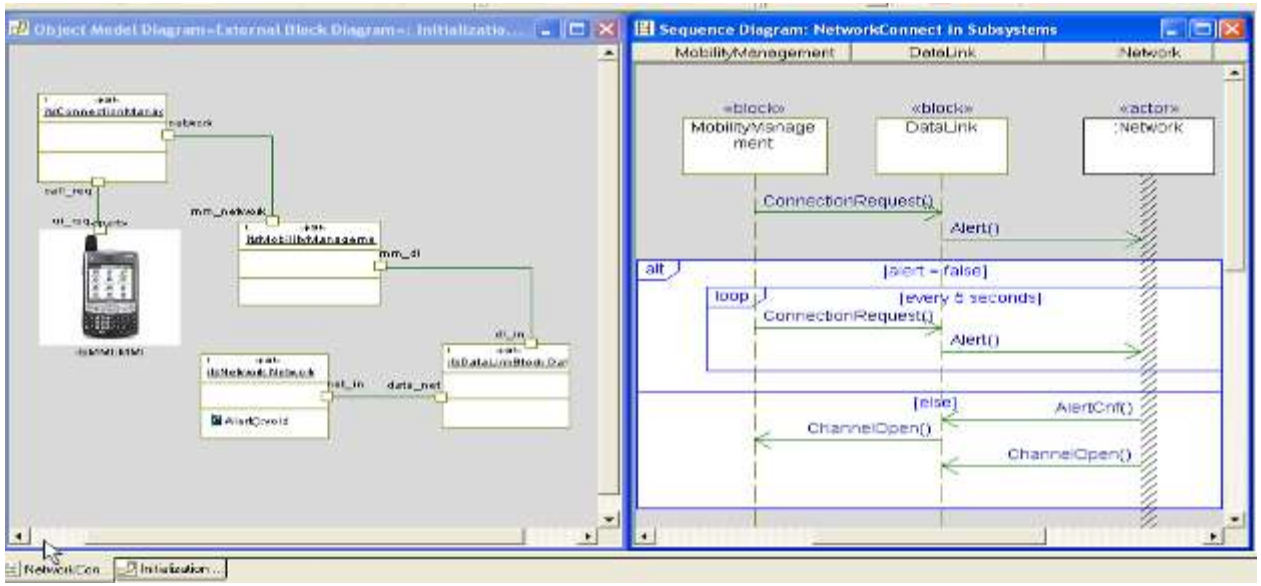




Telelogic Rhapsody Model-Driven Development



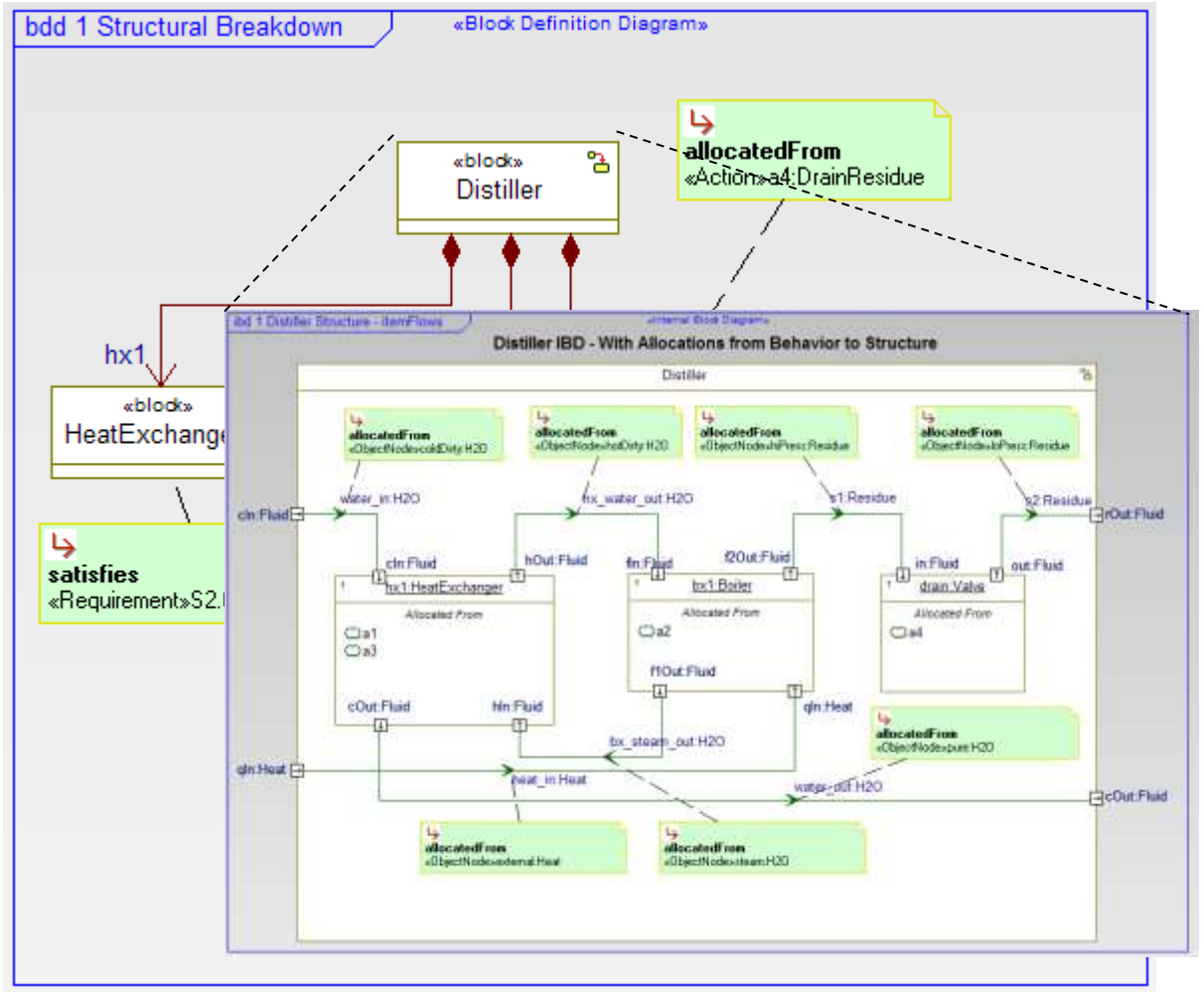
UML 2.1



UML views

- Structure
  - Structure diagram
  - Package diagram
  - Component diagram
  - Object model diagram
    - Class and object
  - Deployment diagram
- Behavior
  - Statecharts
  - Activity diagrams
  - Use case diagram
- Interaction
  - Sequence diagram

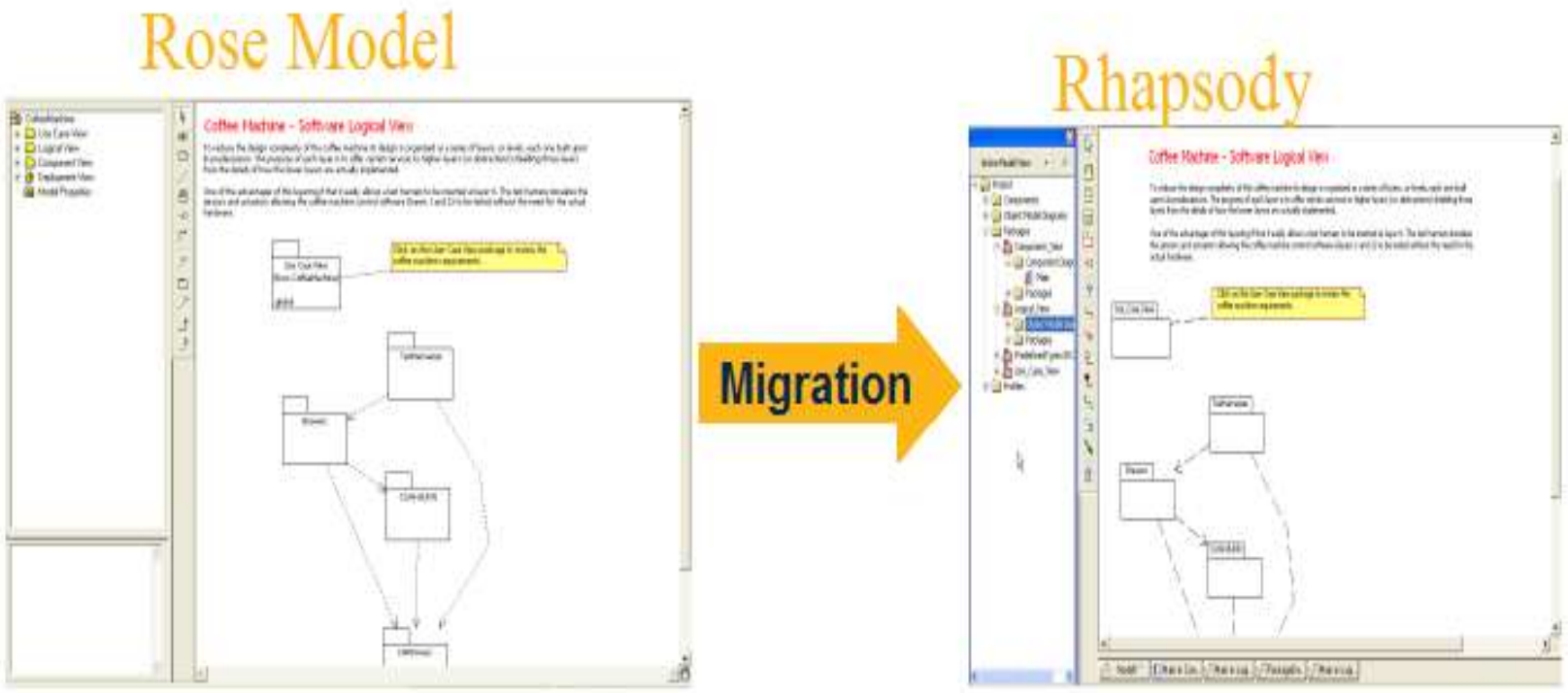
SysML



- Systems Modeling Language (SysML)
- SysML diagrams
  - Requirements
    - Requirements diagram
    - Use case diagram
  - Structure
    - External block diagram
    - Internal block diagram
  - Behavior
    - Statechart
    - Activity diagram
    - Sequence diagram
  - Constraints
    - Parametric diagram

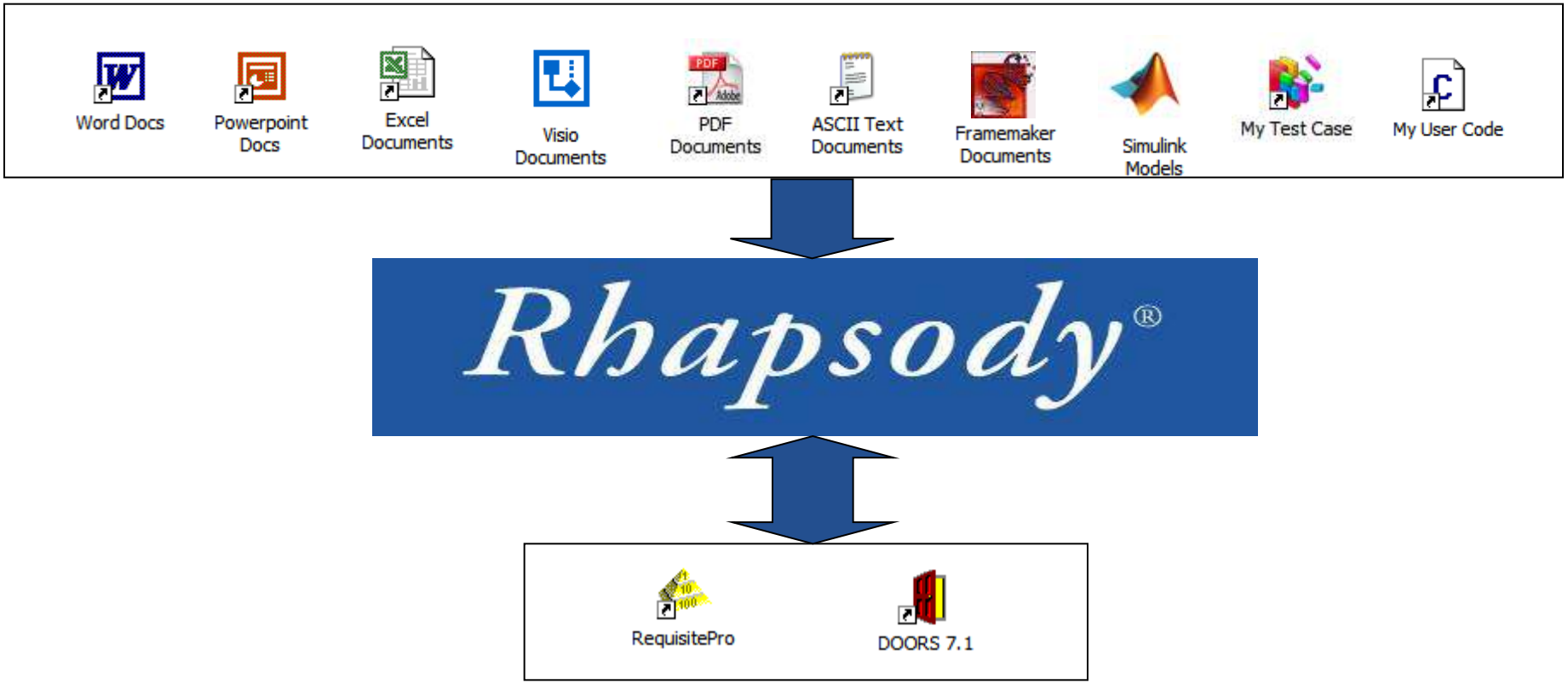
Legacy model reuse

- Import and reuse your old Rational or XMI-compliant models directly into Rhapsody



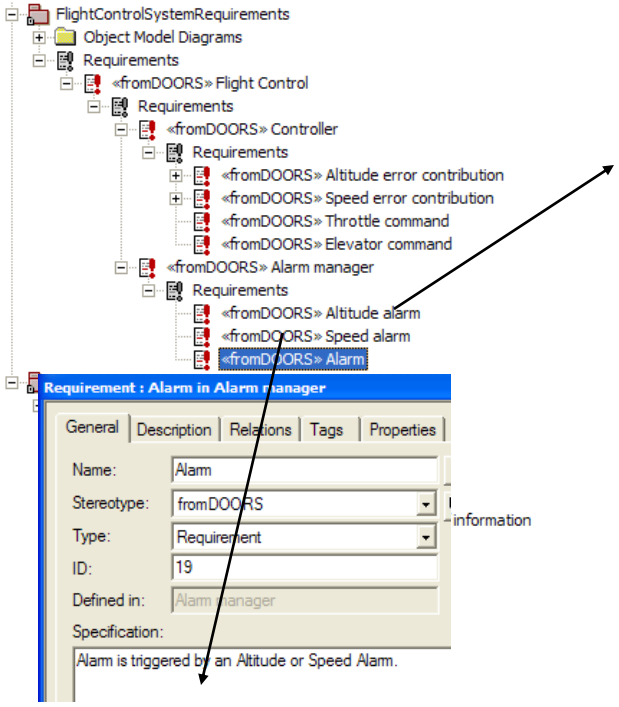
Lifecycle traceability

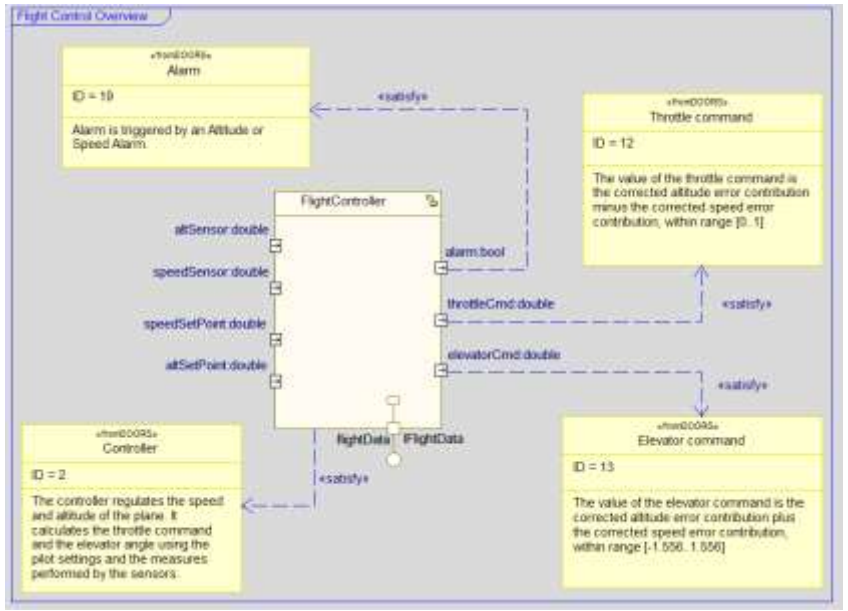
- Create traceability links from model to requirements
- Produce automatic traceability documentation
- Import requirements from multiple sources



Visual requirements capture

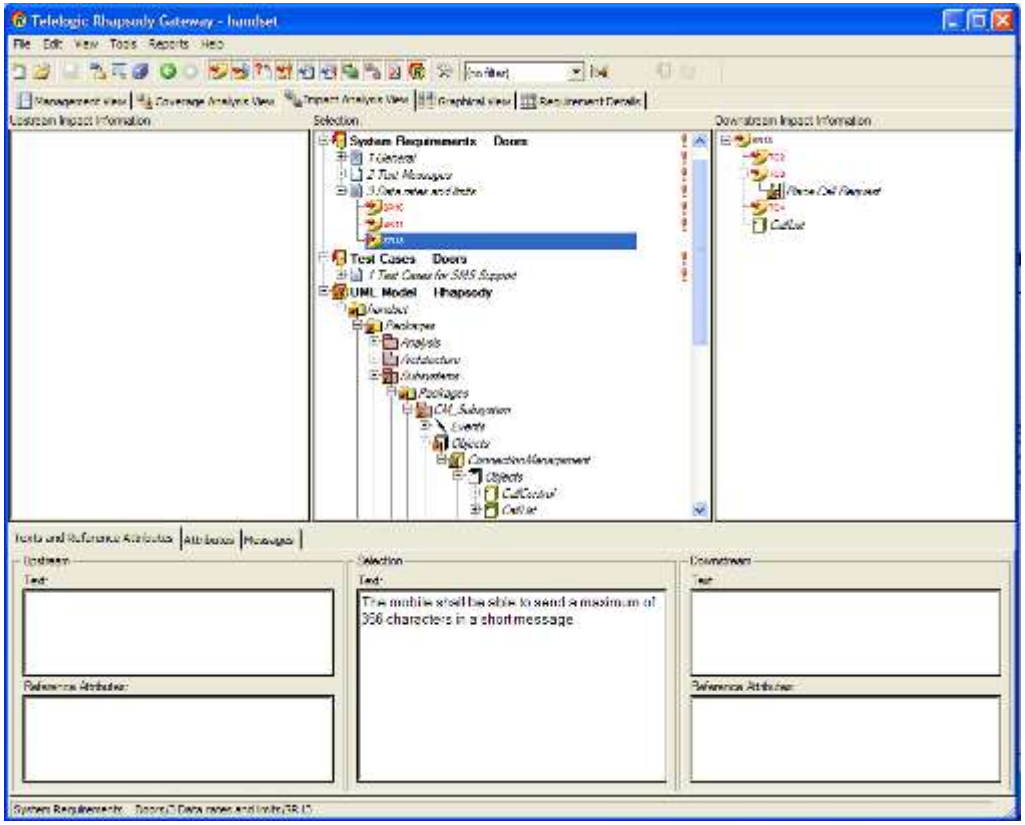
- Use requirements and use case diagrams to define requirements
- Supplement definitions and descriptions with tags and constraints
- Describe requirements behavior using sequence, activity and state diagrams
- Include advanced graphics and icons with domain-specific modeling





Change impact analysis

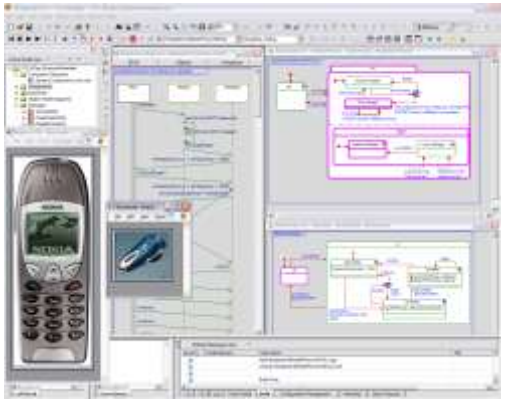
- Locate elements potentially impacted by a requirement change
- Determine requirements possibly impacted by a design change



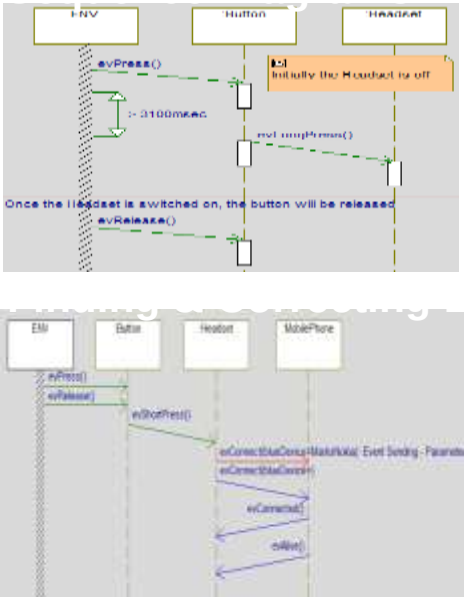
Model-driven testing

- Bring the benefits of abstraction and automation to testing
- Reduce defects early in the process when they are less costly to fix
- Deliver products meeting customer expectations

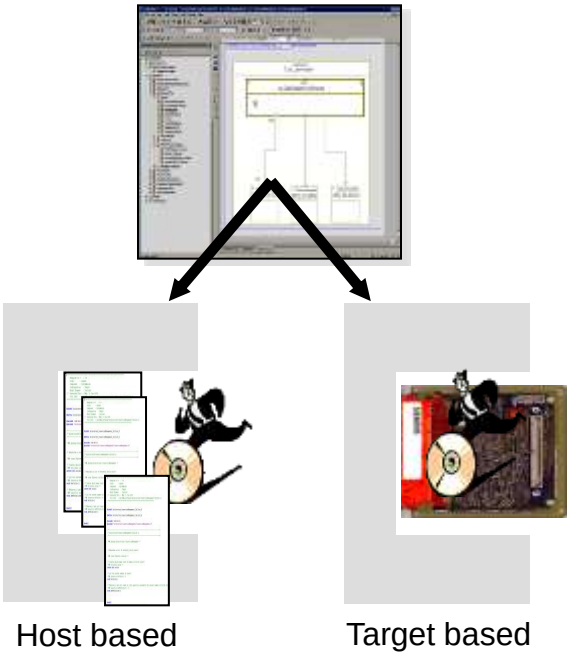
Simulation



Requirements-based testing



Automated unit testing

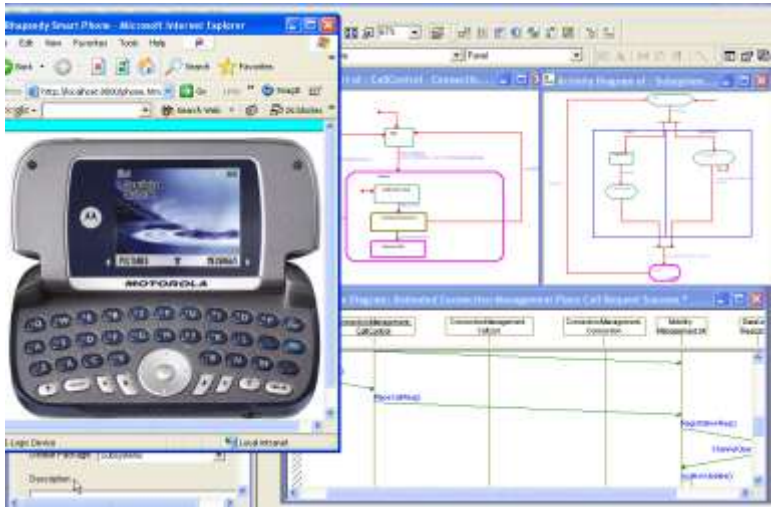


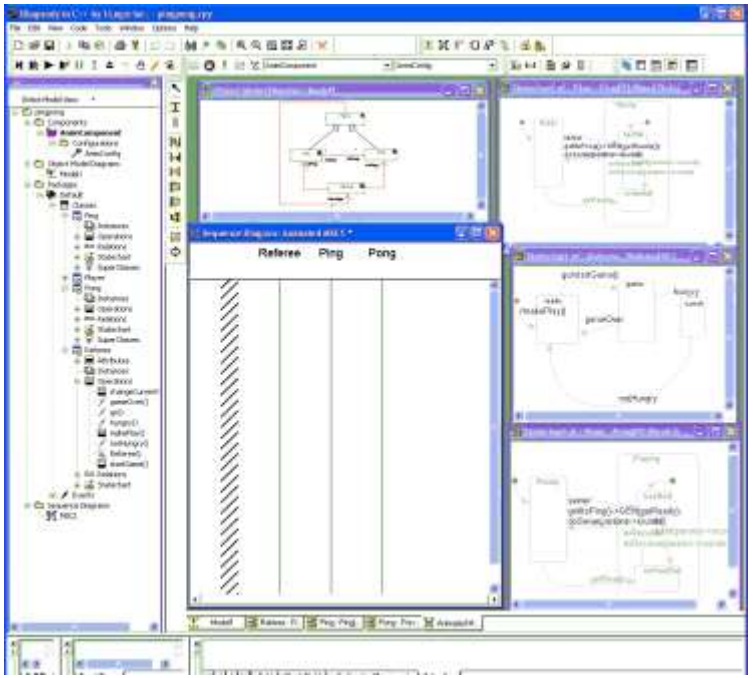
Host based

Target based

Simulation, execution and animation

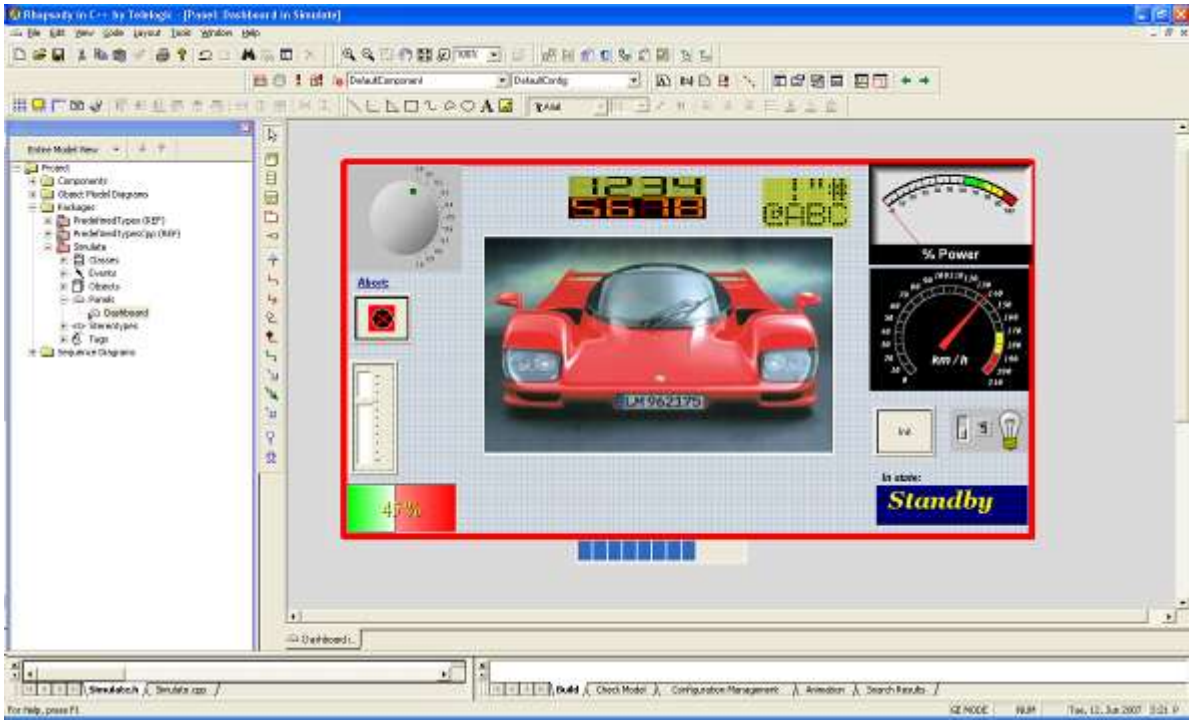
- Simulate to verify that model is correct
  - Best practice for avoiding errors and thereby helping to reduce development cost
  - Rapid simulation at the design level or even target level debugging
- Virtual prototype/panel graphics support
  - Ideal communications aid for design reviews and to share information





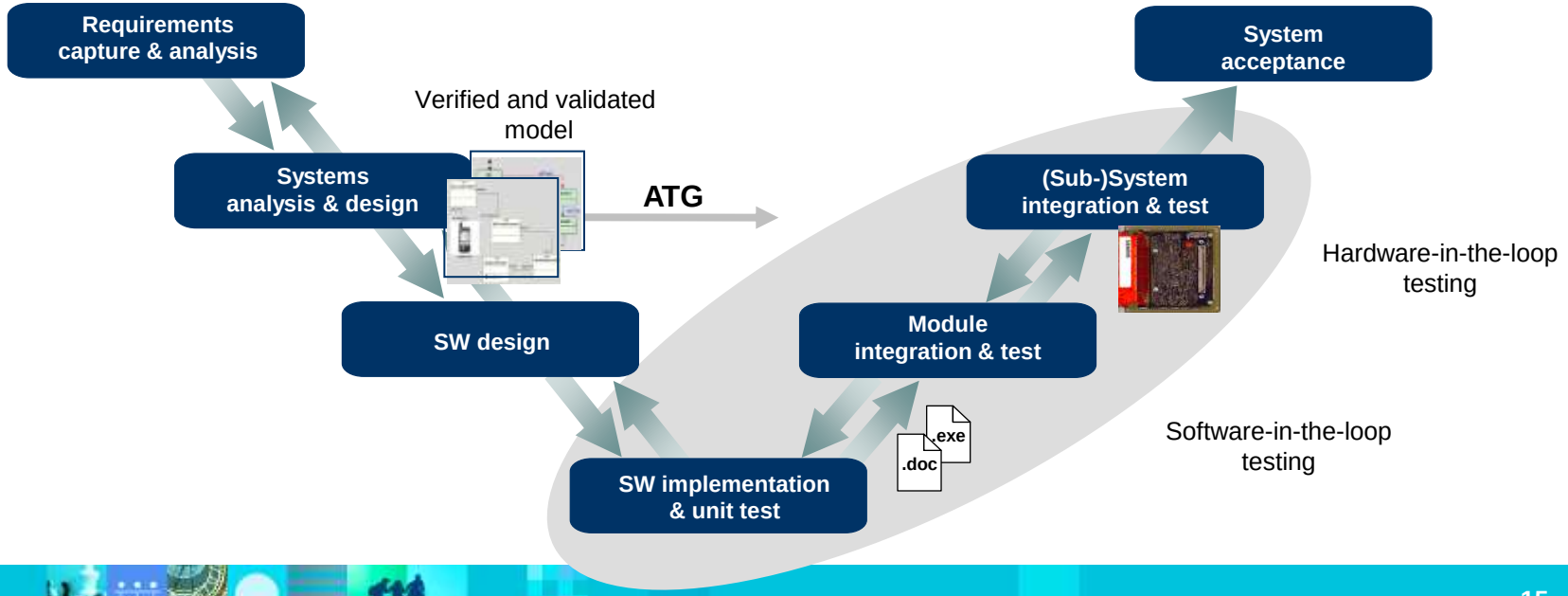
Graphical panels

- Create mock-ups of interface to effectively communicate intended design behavior to customers
- Modify, monitor and analyze data values during simulation to help ensure that the design is correct early in the process



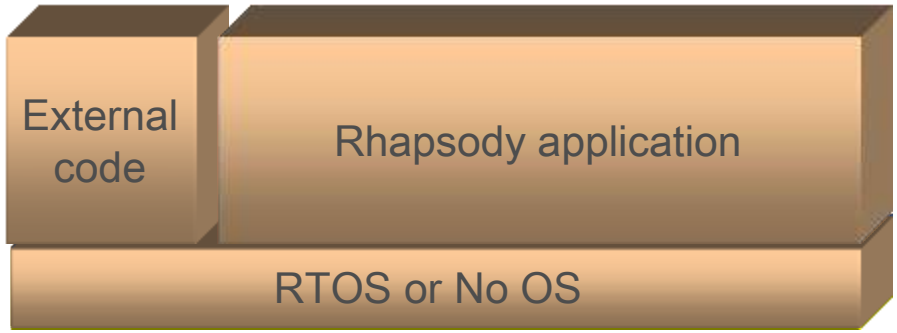
Automatic test generation

- Automatic test generation (ATG) provides model-driven test generation
  - Generates test cases with high coverage of the model
  - Consistent with the UML testing profile
- Automatically generates tests from the model
  - Coverage of states, transitions, operations, events
  - Produce all relevant combinations of inputs for MC/DC analysis
- Perform regression testing on the model
- Export to third-party tools for testing the implementation



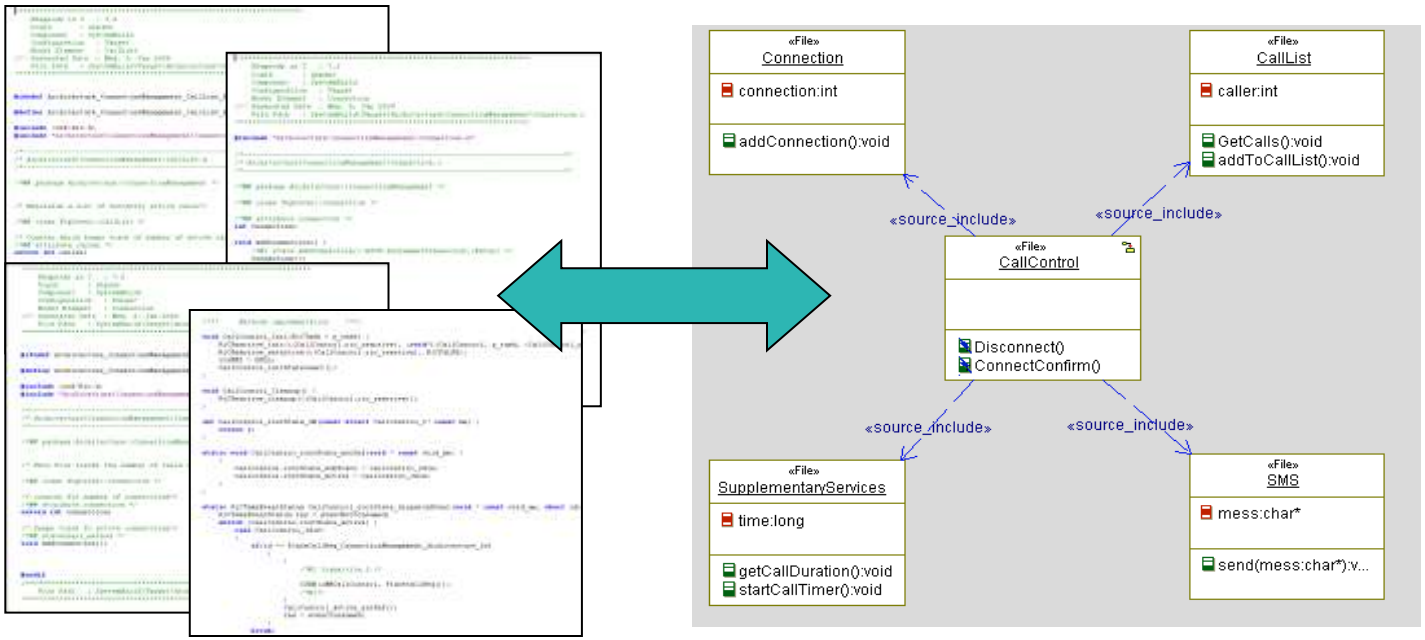
Full application generation

- Meet-time-to market pressures with complete applications, not frames
  - Generate C, C++, Java™ and Ada applications
  - Rhapsody generates very clean, readable code, readily debugged through any commercial IDE, including Eclipse
- Rapidly deploy your design onto any target platform
  - Provides platform-independent models (PIMs)
- Flexible development environment, work at code or model level

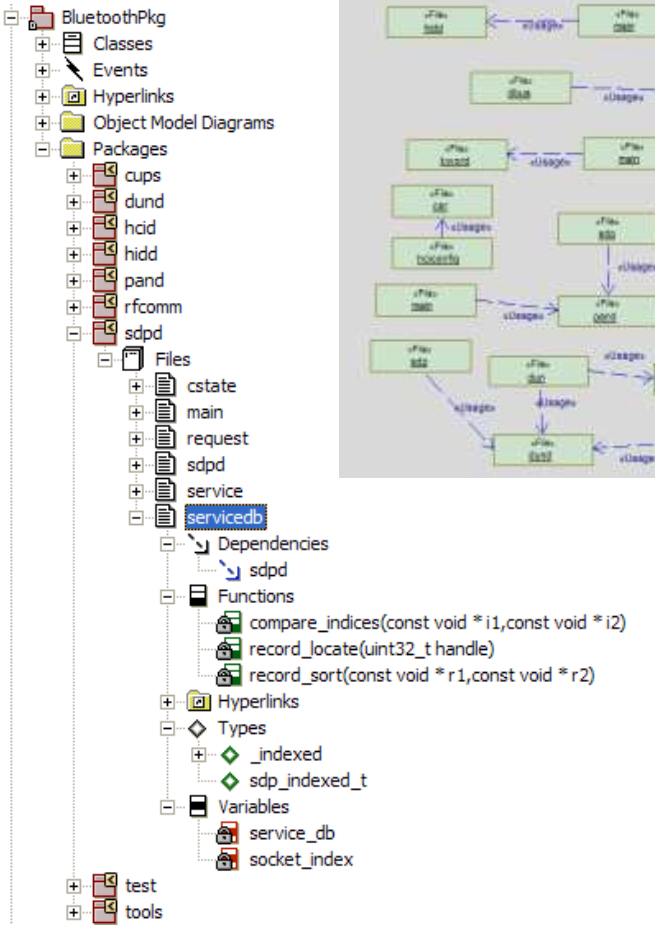


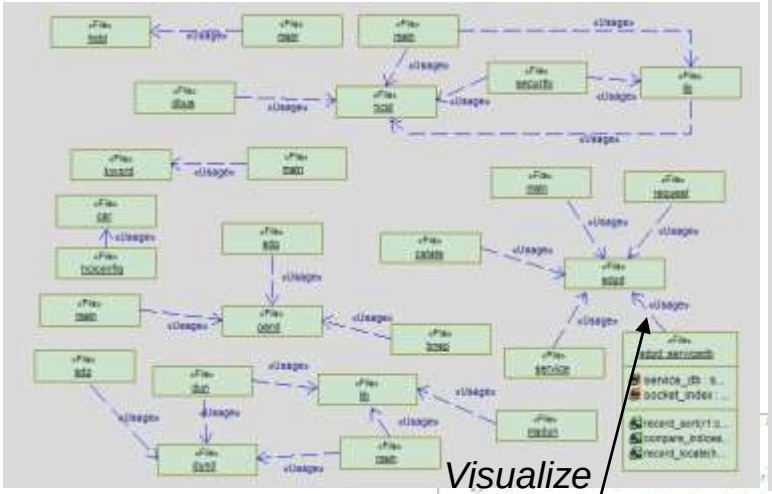
Reuse existing code (IP)

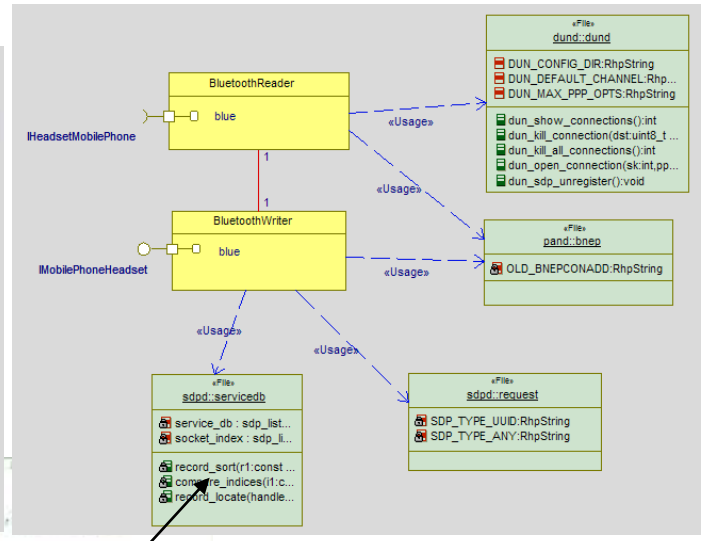
- Reuse code from other projects
- Integrate code developed by a third party
- Visualize in the model for better understanding
- Reference third-party code within Rhapsody



Code visualization example

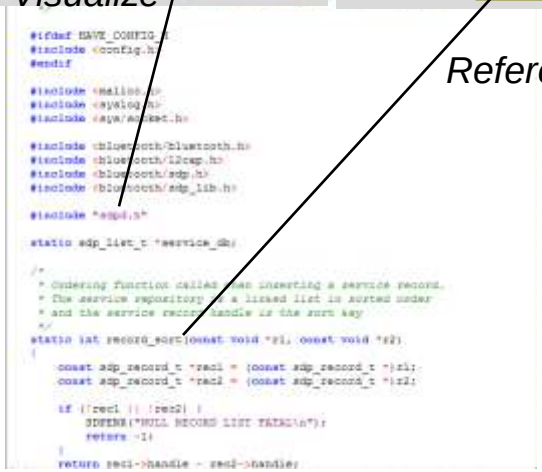






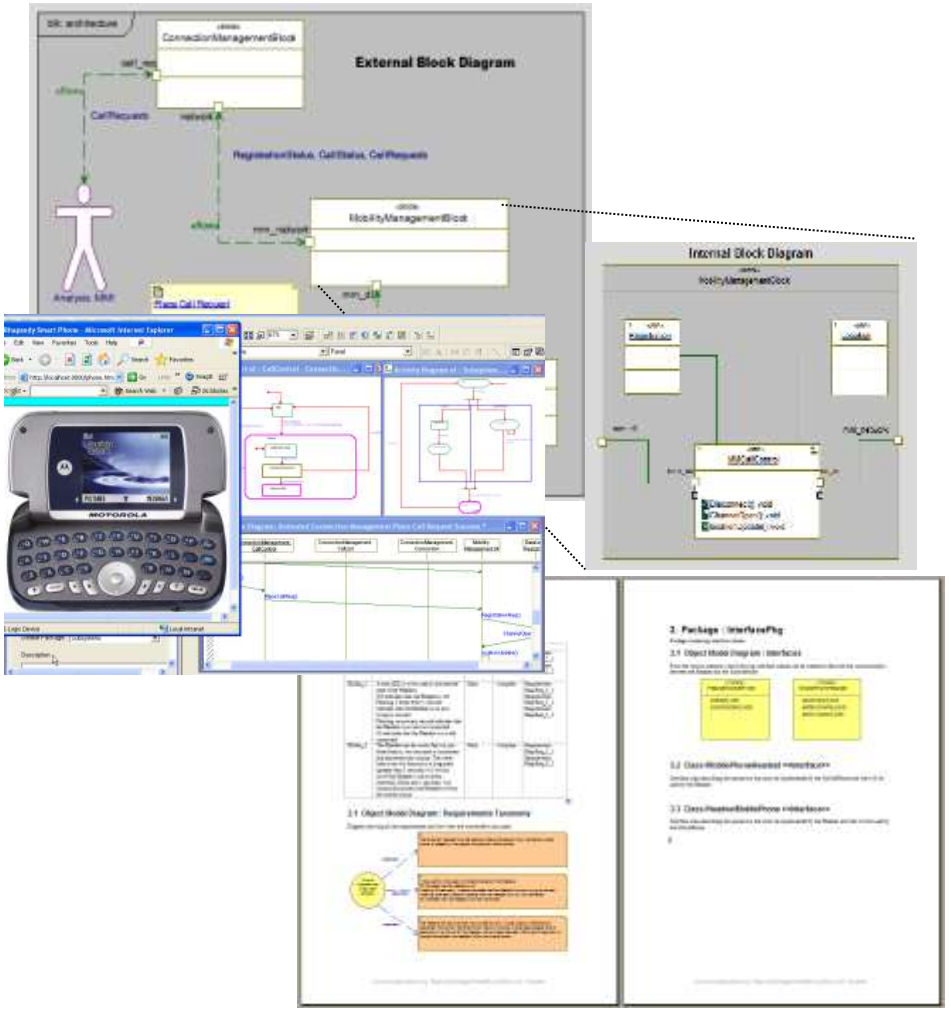
Visualize

Reference



Common visual modeling

- Common graphical modeling language (SysML and UML)
  - Nonambiguous
  - Readily understood by all stakeholders
- Example product deliverables
  - System models
  - Software models
  - Reusable asset models
  - Requirements traceability documentation
  - Specification documentation
  - Design documentation
  - Executable reference model
    - Simulation
    - Prototypes
  - Test cases and scenarios



Collaboration

- Tight integration with configuration management tools
  - Telelogic Synergy, Rational ClearCase, Rational Team Concert or any SCC-compliant tool
- Graphically identify differences between versions and evaluate design alternatives
- Quickly accept or automatically and merge changes between multiple versions