



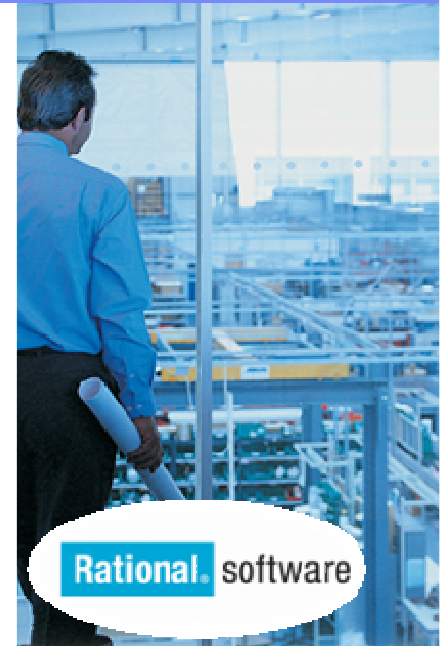
IBM Software Group

*Turning Product Development Into  
Competitive Advantage:*

# IBM Rational Solutions for Complex Systems and Software Engineering

**Eugen PASLARU**

*Rational Technical Specialist CEEMAS*



**Rational** software

[Goto IBM](#)

## Products are Getting Smarter Every Time We Look

- One billion camera phones were sold in 2007, double that of 2006
- One customizable device: phone, e-mail, music, Web, camera, GPS, apps, video recorder, e-reader, ...
- User productivity and enjoyment have skyrocketed
- In 2000 this would have been science fiction
- In 2012 it's **yesterday's news!**

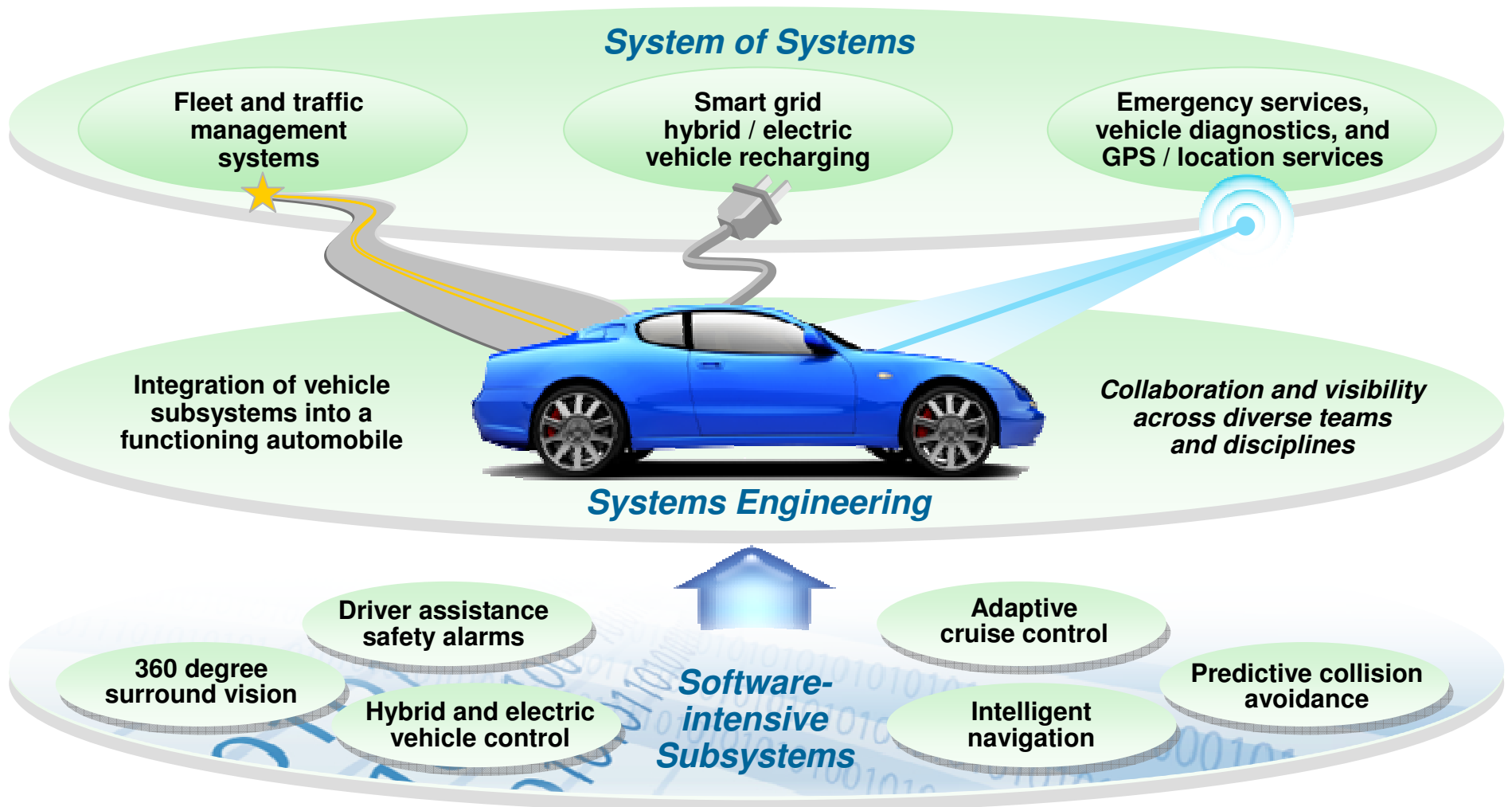


***What's possible by 2020?***

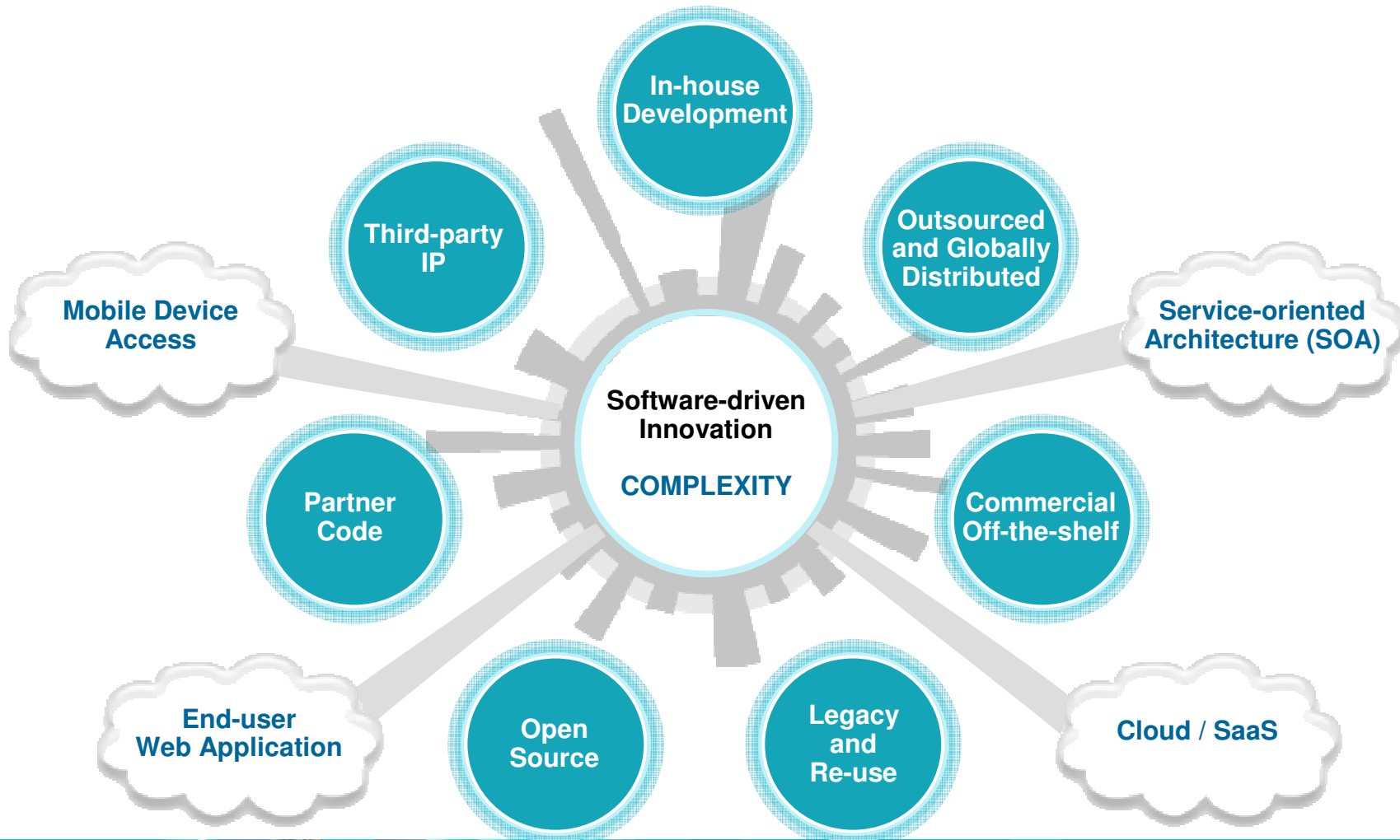


# Smart Products Require Innovative Systems and new Development Methodologies

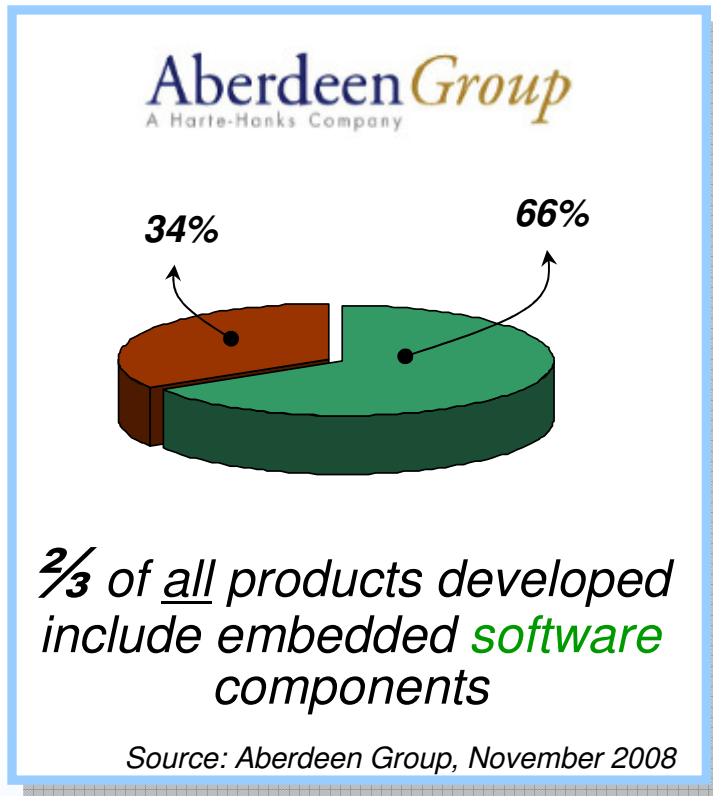
*Incremental value is created by global interconnection across products, systems, applications and networks*



# The defining challenge: Managing “systems of systems”



## Software is the Heart of Today's Systems Innovation



*“The medical field is highly dependent on **software**, which significantly enhances delivery of patient care”*

*“Like many of the components that make up today's vehicles, the hydraulic hybrid systems are intelligent **software**-intensive systems.”*

*“**Software** has evolved from a hidden component driving functionality to the keystone of product differentiation and end-user experience.”*

-- VDC Research



# Complexity Creates Development Challenges

*Leading to cost overruns, schedule slips and quality issues*

***Poor requirements engineering = failed projects***

***Paper-based and manual processes hinder efficiency***

***Complex architecture is difficult to textually explain***

***Functionality is poorly distributed across components***

***Hardware/software integration is often late***

***Many organizations lack formalized practices***

***Silos of people,  
process, and projects***

## **Geographic Barriers**

- Poor communication
- Language, culture, time
- Process gaps resulting in rework

## **Organizational Barriers**

- Weak collaboration
- Poor project governance and LOB oversight
- Security of IP

## **Infrastructure Barriers**

- Incompatible tools
- Unreliable access
- Lengthy on-boarding
- Inflexible integration



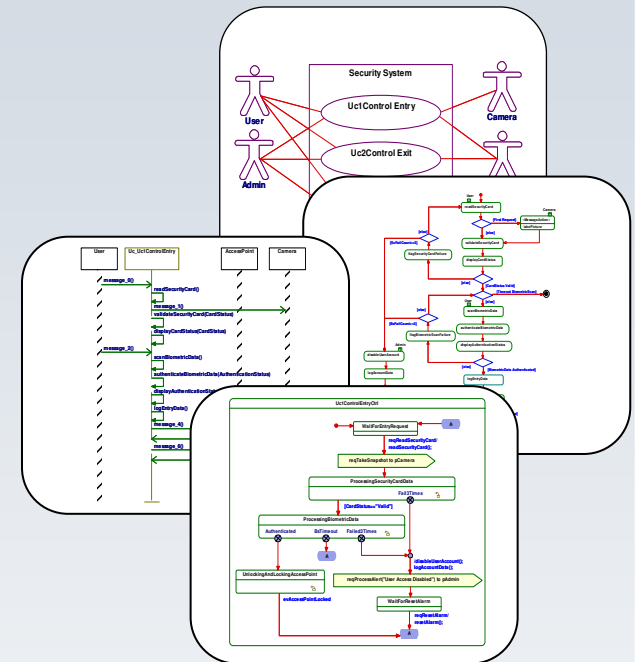
# Modern Approaches for Describing Systems Are Evolving *To Better Manage Complexity and Reduce Time-to-market*

*Past*



Specifications  
Interface requirements  
System design  
Analysis & trade-off  
Test plans

*Future*



*Moving from manual methods to an automated, visual approach*





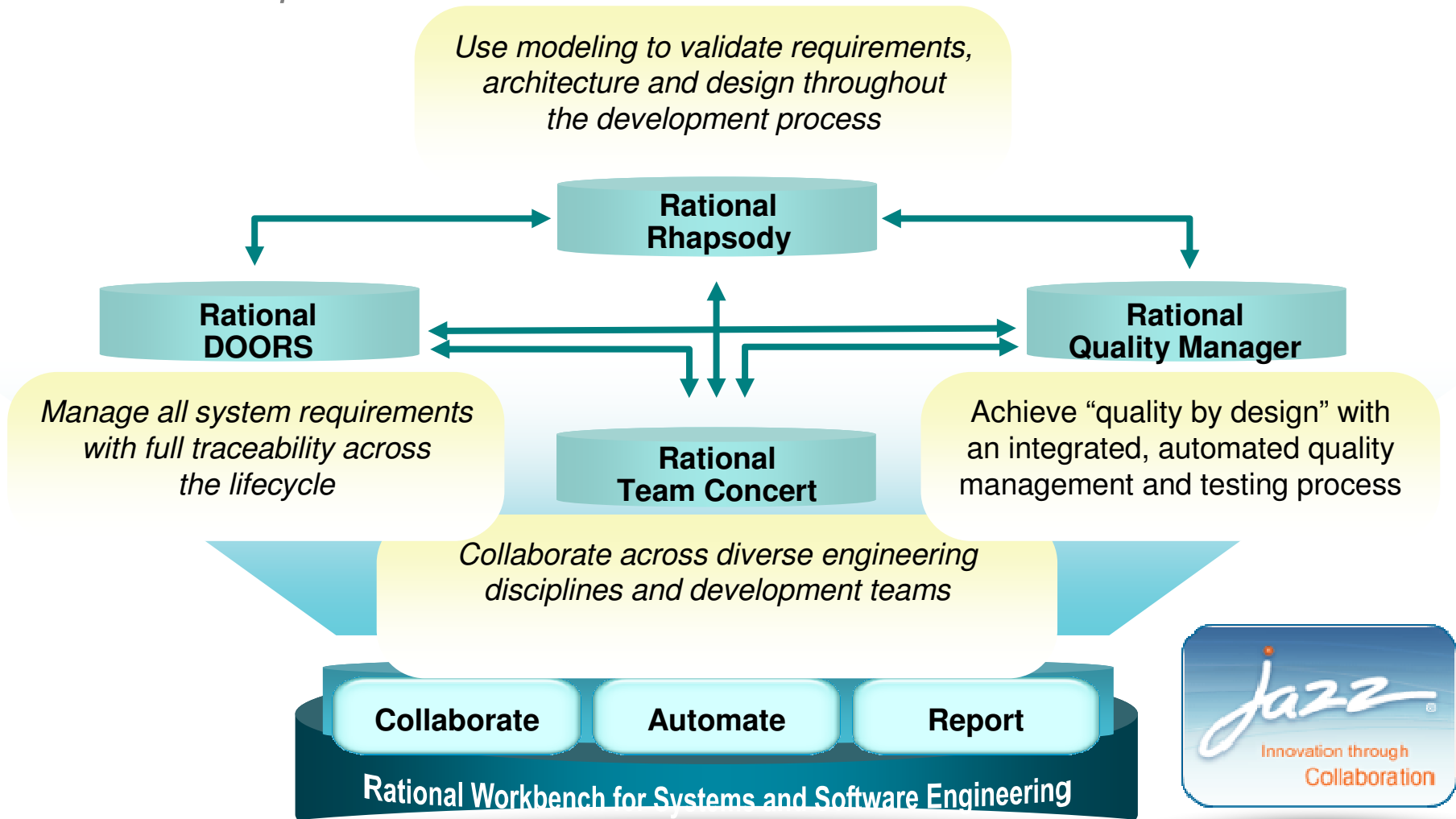


# Rational Core Components for Systems and Software Engineering



# Rational Solutions for Systems and Software Engineering

*Built on a core product set*



## ALM is about connecting the disciplines

### Project/Planning

- Business Drivers
- Iterations
- Sign-off
- Contract
- Risk Assess
- User Involvement

### Requirements

- Use Cases
- Nonfunctional
- Sign-off
- Contract
- Risk Assess
- Threat Model
- Test Requirements

### Development

- Test Driven Development
- Build Management
- Static Analysis
- Source Management
- Pair Programming/Code Review

### Testing

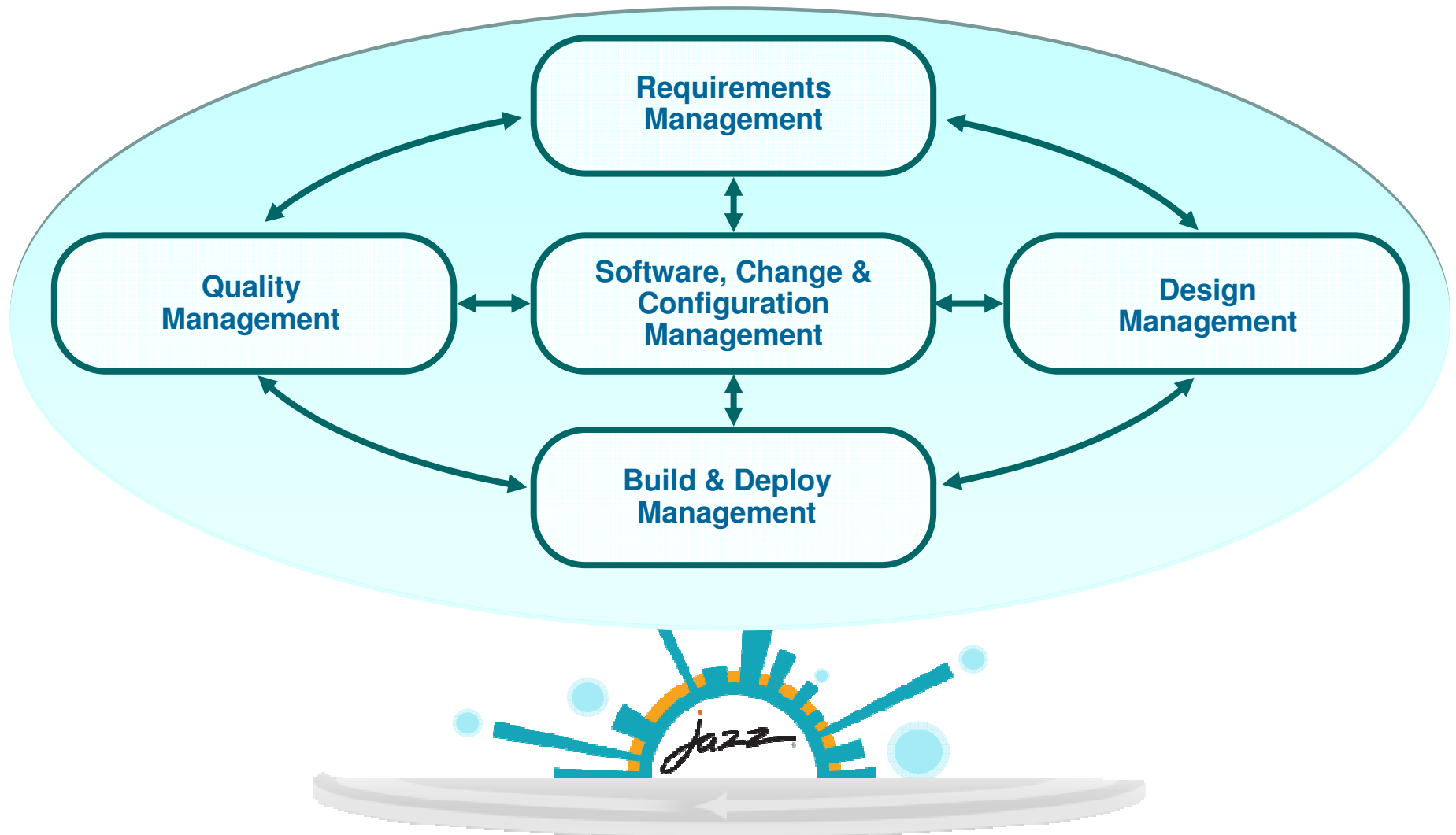
- Scenario-Driven Automation
- Exploratory Test
- User Involvement
- Contract Validation

Continuous Learning and Feedback

Source: Gartner Application Architecture, Development & Integration Summit Presentation, The Future and Present of AD, Thomas E. Murphy, December 2008

# Rational Application Lifecycle Management (ALM)

*Modular, open and extensible*



# Domain Focused Development

*Apply industry and domain standards*

Standards-  
based  
Development

- **Interconnected diagrams** form multi-dimensional models
  - ▶ Can describe even the most complex systems

- Unified Modeling Language – UML 2.x
  - ▶ Industry-standard notation for specifying, visualizing, and documenting systems and software designs
- Systems Modeling Language - SysML
  - ▶ Extends/specializes UML to address needs of the Systems Engineer
  - ▶ Open standard published by the OMG and INCOSE

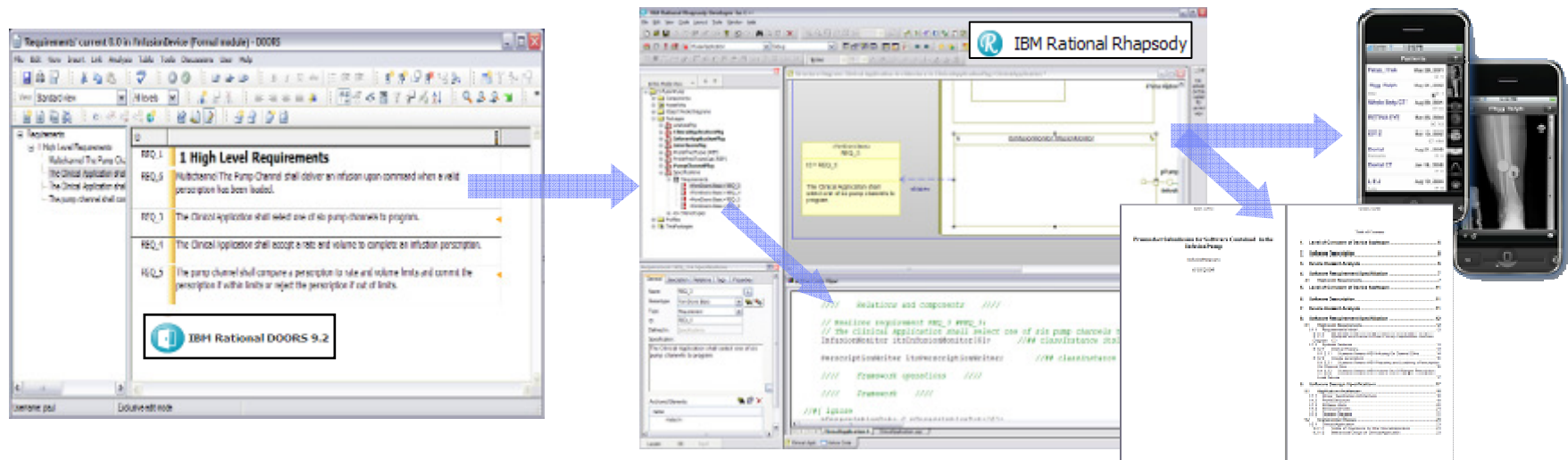


- Industry notations and frameworks: DoDAF, MODAF, UPDM, AUTOSAR...



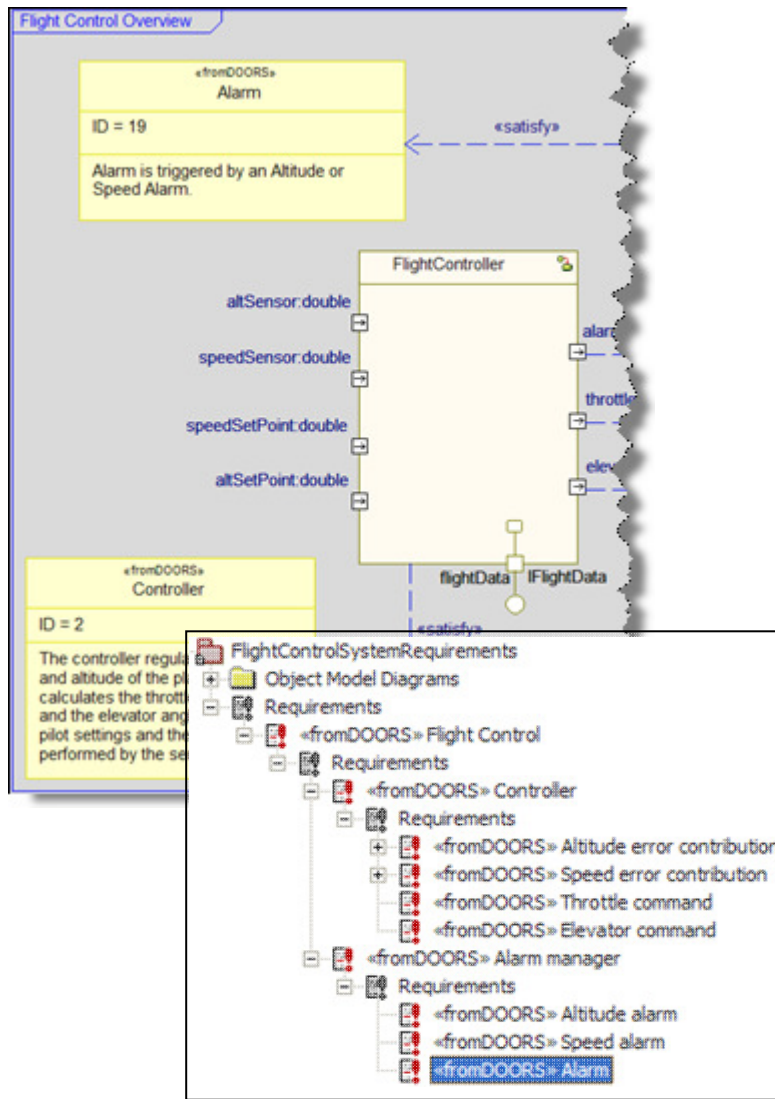
## Manage Requirements across Lifecycle and Disciplines

- *Build the right product* because the requirements are visible at all times
  - ▶ Prove that all requirements (user, safety, regulatory, etc.) were fully satisfied
- Understand the requirements
  - ▶ Analyze stakeholder needs
  - ▶ Evaluate coverage and impact analysis
- Validate the requirements
  - ▶ Analyze for correctness and to determine next steps





# Translate Requirements into a System Design



- *Build the product right* with structural and behavioral analysis and design
- Visualize the system
  - ▶ Reduce confusion over requirements
  - ▶ Specify system functionality
  - ▶ Simulate to confirm functionality
- Analyze impact of changes
  - ▶ Whether in requirements or design
- Trace requirements in either direction
  - ▶ Provide full accountability and understanding
- Specify and develop software
  - ▶ Monitor and control the system



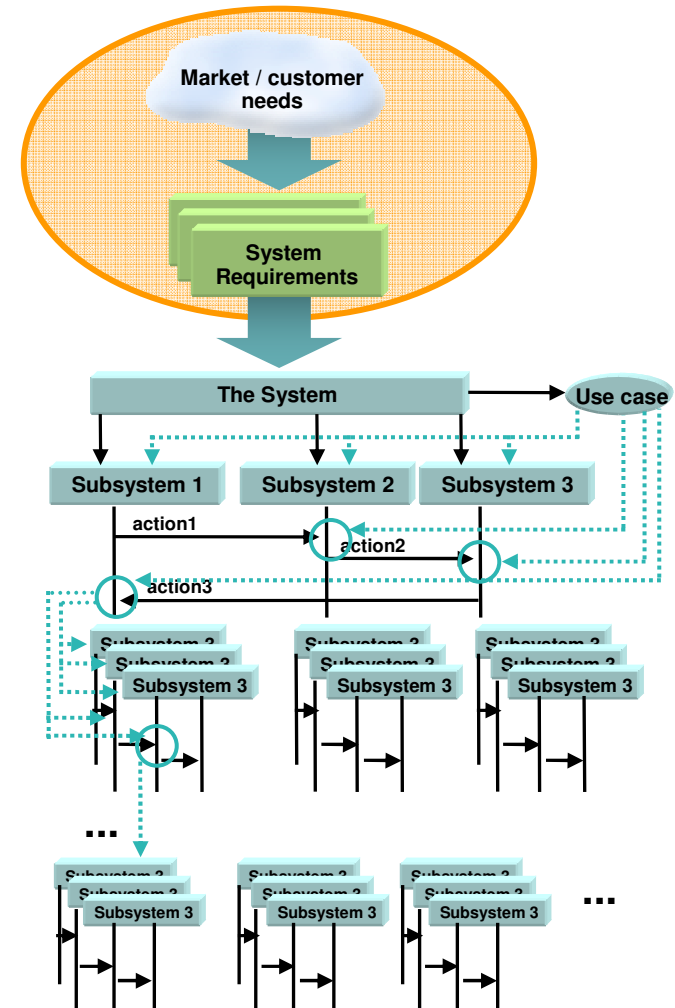
# Create and Manage Your Architecture

*Transform requirements into a working system*



- **Derive the system** in the context of its environment
  - ▶ Reduce confusion over requirements
  - ▶ Establish system functionality and its constraints
- **Eliminate errors** as they are introduced
  - ▶ Before they are too expensive to find and repair

- ▶ **Simulate often**
  - ❖ Animate and execute the design model
  - ❖ Validate functionality and verify correctness
- ▶ **Automatically create and execute tests**
  - ❖ Derive from the design model or target platform
  - ❖ Create test harnesses for unit testing
- ▶ **Manage test cases**
  - ❖ Prioritize the features and functions to be tested



○ Derived requirement  
 .....▶ Traceability link

## Build in Quality from Concept to Launch

- Simulate often to validate functionality and verify correctness
- Automatically create and execute tests from the design model or target platform
- Manage test cases, while prioritizing the features and functions to be tested

The screenshot displays the Rational Quality Manager interface. The main window shows the 'Execution Result' for a test case named 'SD\_tc\_0', which has passed. The test was executed on 10/20/31, Monday, April 27, 2009. The test case is associated with the test plan 'TestPlan\_CashRegister...' and the test script 'SD\_tc\_0'.

The 'Actual Result' section shows a green checkmark and the word 'Passed'. The 'Host Name' is 'jekyllslave', the 'Owner' is 'Mary, Test Manager', and the 'Test Milestone' is 'TestCase\_01\_SD\_InitCashRe'. The 'Test Case' is 'SD\_tc\_0', the 'Test Script' is 'SD\_tc\_0', and the 'Test Data' is 'Unassigned'. The 'Weight' is '100'.

The 'Result Details' section lists the following files: 'TCon\_CashRegister\_\_SD\_tc\_0.html', 'TestConductorAdapter20844.out', 'TestConductorAdapter20845.err', and 'TestLog20843.log'.

The 'Environment Info' section provides details about the test environment:

Environment Info	
Test executed on machine:	JECKYLSLAVE
Test executed by user:	Administrator
Used OS version:	Windows 2000 / Windows XP
Used Rhapsody version:	7.5, build 1195117
Used TestConductor version:	2.4, build 1406

The 'Tested Project' section provides details about the project being tested:

Tested Project	
Project:	CppCashRegister
Active Component:	TRng_CashRegister_Comp
Active Configuration:	DefaultConfig

The 'SDs used in test' section lists the test scenarios used:

SDs used in test	
TRng_CashRegister::SDTestScenario_0	

The 'Summary Info' section provides a summary of the test results:

Summary Info	
Summary:	passed
Total number of SDs used:	1
Total number of SD instances in test:	1
Total number of executed SD instances:	1
Total number of PASSED SD instances:	1 (100%)
Total number of FAILED SD instances:	0 (0%)
Total number of ACTIVE SD instances:	0 (0%)
Total number of NOT ACTIVE SD instances:	0 (0%)

The 'Sequence Diagram' section shows a sequence diagram for the test scenario 'SDTestScenario\_0'. The diagram includes two lifelines: '«SUT» TCon\_CashRegister.itsCashRegister' and 'TCon\_CashRegister.itsTC\_at\_hw\_of\_CashRegister:TC\_at\_hw\_of\_CashRegister'. The sequence of events is as follows:

- The SUT lifeline starts with a vertical dashed line.
- The SUT lifeline sends a message 'evStart()' to the TC lifeline.
- The TC lifeline sends a message 'show(aMsg = Ready)' back to the SUT lifeline.
- The SUT lifeline sends a message 'evEnd()' to the TC lifeline.
- The SUT lifeline ends with a vertical dashed line.



## Recapture Intellectual Property

- Preserve intellectual property
  - ▶ Visualize and reverse-engineering existing software
  - ▶ Create a library of design assets
  - ▶ Analyze to best meet requirements
- Work with product lines
  - ▶ Expand product offerings
  - ▶ Exploit commonality across products
  - ▶ Focus efforts on unique product variants



Utility



Speed

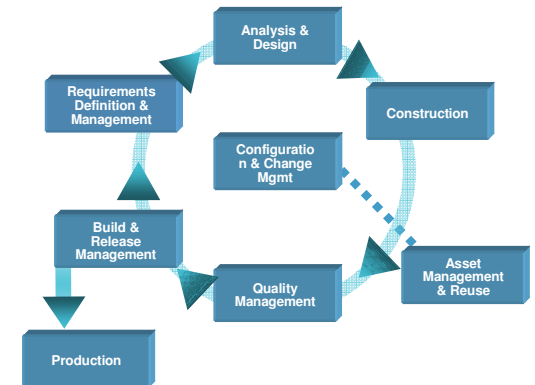


Fuel economy



## Extend the Solution to Meet Your Needs

The Rational solution can be tailored to meet virtually any systems development workflow :



- ▶ *Automated reporting and documentation* with **Rational Publishing Engine**
- ▶ *Enterprise systems delivery* with **Rational System Architect**
- ▶ *Embedded software testing* with **Rational Test RealTime**
- ▶ *Team-based configuration management* with **Rational ClearCase** or **Synergy**
- ▶ *Domain specialization with industry-specific profiles such as AUTOSAR, Android, functional safety, and defense architecture frameworks*
- ▶ *Embedded platform development* with **Wind River Workbench/VxWorks**
  - Support also exists for **Green Hills Integrity, QNX Momentics/Neutrino** and many other embedded platform operating system environments
- ▶ and many others...





### Learn more at:

- [IBM Rational software](#)
- [IBM Rational Software Delivery Platform](#)
- [Process and portfolio management](#)
- [Change and release management](#)
- [Quality management](#)
- [Architecture management](#)
- [Rational trial downloads](#)
- [Leading Innovation Web site](#)
- [developerWorks Rational](#)
- [IBM Rational TV](#)
- [IBM Business Partners](#)
- [IBM Rational Case Studies](#)

© Copyright IBM Corporation 2010. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. IBM, the IBM logo, Rational, the Rational logo, Telelogic, the Telelogic logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

