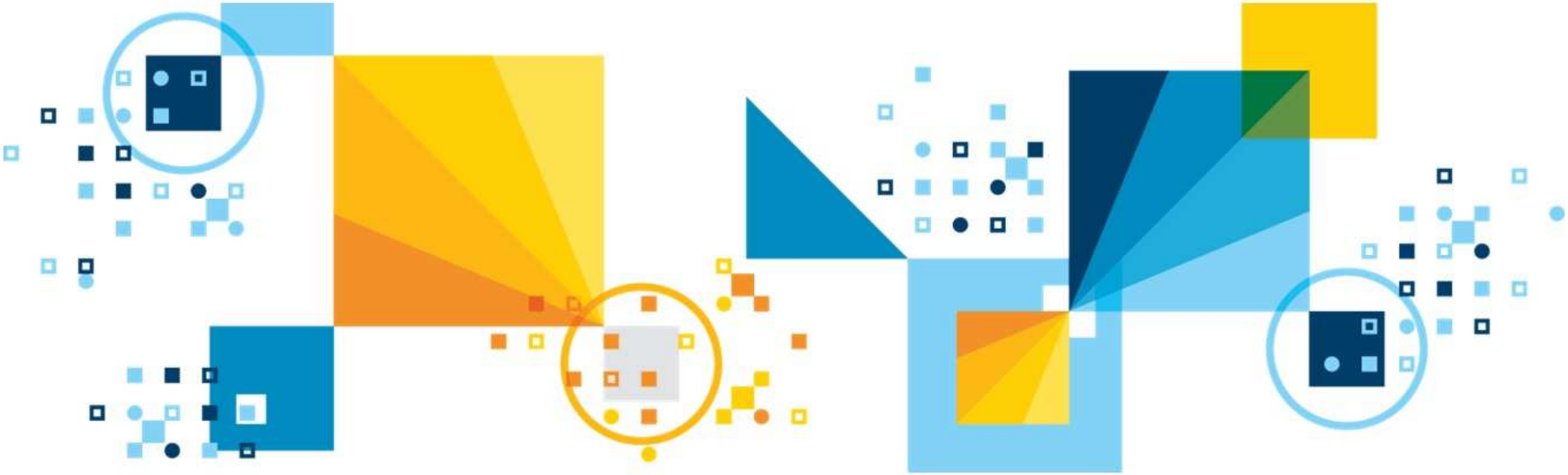
Using Continuous Engineering to Accelerate Innovation in the Automotive Industry



Watson IoT

Continuous engineering is about game-changing capabilities

Continuous engineering is an enterprise capability that speeds delivery of increasingly sophisticated and connected products by helping businesses to evolve their engineering practices to adapt to the accelerating pace of business change.

“Turn Insight into Outcomes”

Unlocking Engineering Knowledge

“Measure twice, cut once”

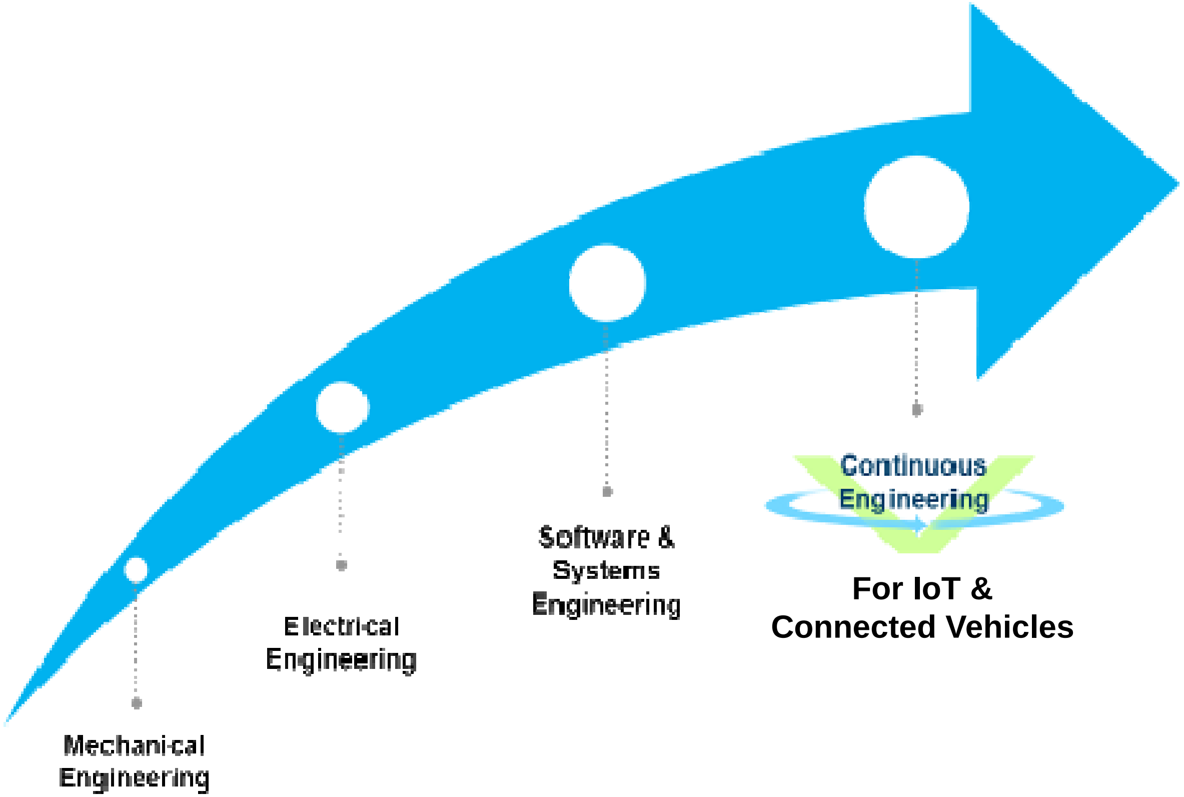
Continuous Verification

“Don’t reinvent the wheel”

Strategic Reuse

Continuous Engineering

Evolve your engineering practices to turn today's market trends into tomorrow's competitive advantage



For IoT & Connected Vehicles

Unlock Engineering Knowledge

Access, unlock and understand all engineering information, regardless of source – to enable the right decisions at the right times

Continuous Verification

Verify requirements and design at all stages of the product lifecycle – to prevent rework and achieve faster time to quality

Strategic Reuse

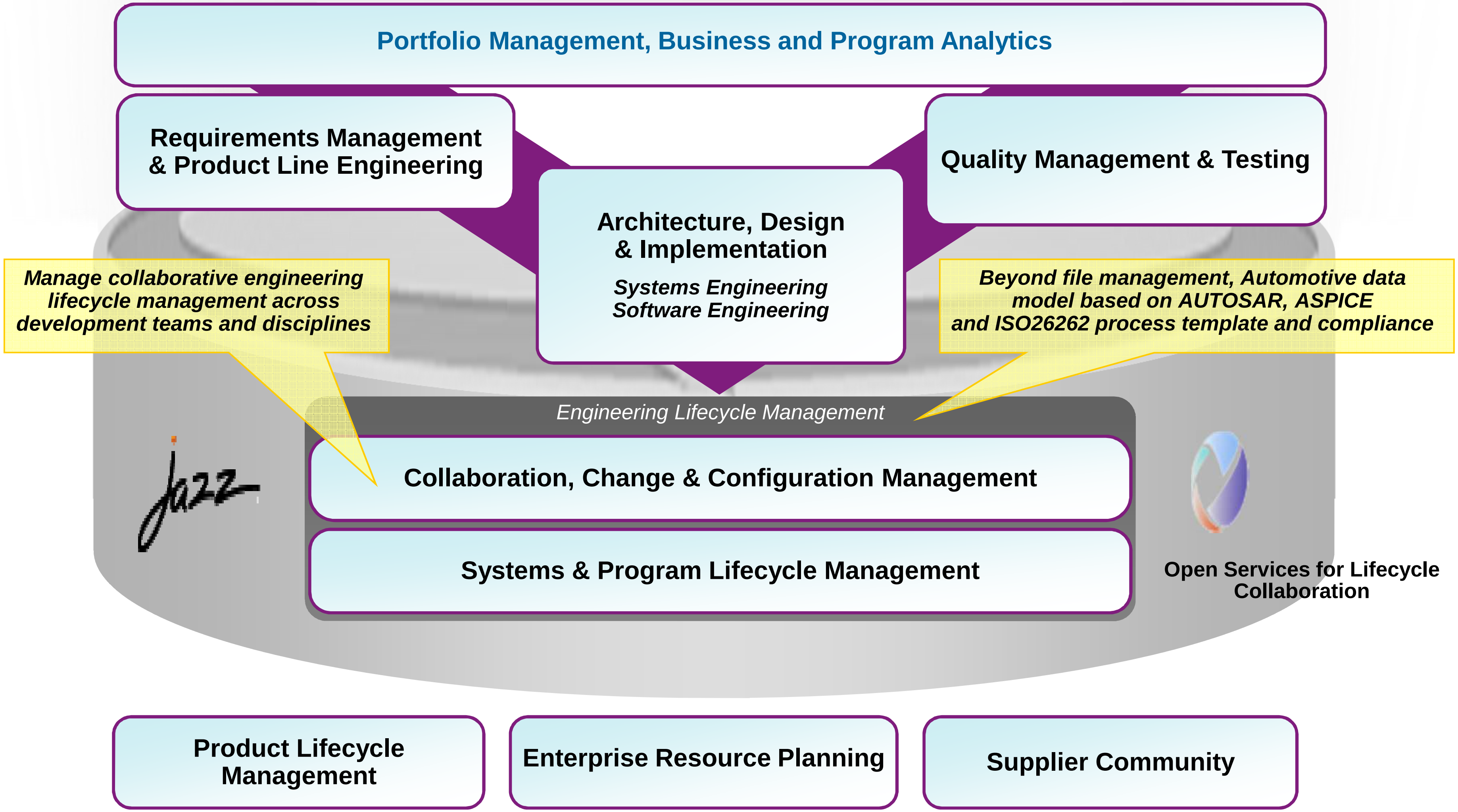
Increase design efficiencies, engineer product lines, and tame complexity

Watson IoT

IBM

IBM Continuous Engineering for Automotive

Integrations based on standardized and open technologies



Portfolio Management, Business and Program Analytics

Requirements Management & Product Line Engineering

Architecture, Design & Implementation

Systems Engineering
Software Engineering

Quality Management & Testing

Manage collaborative engineering lifecycle management across development teams and disciplines

Beyond file management, Automotive data model based on AUTOSAR, ASPICE and ISO26262 process template and compliance

Engineering Lifecycle Management

Collaboration, Change & Configuration Management

Systems & Program Lifecycle Management

Open Services for Lifecycle Collaboration

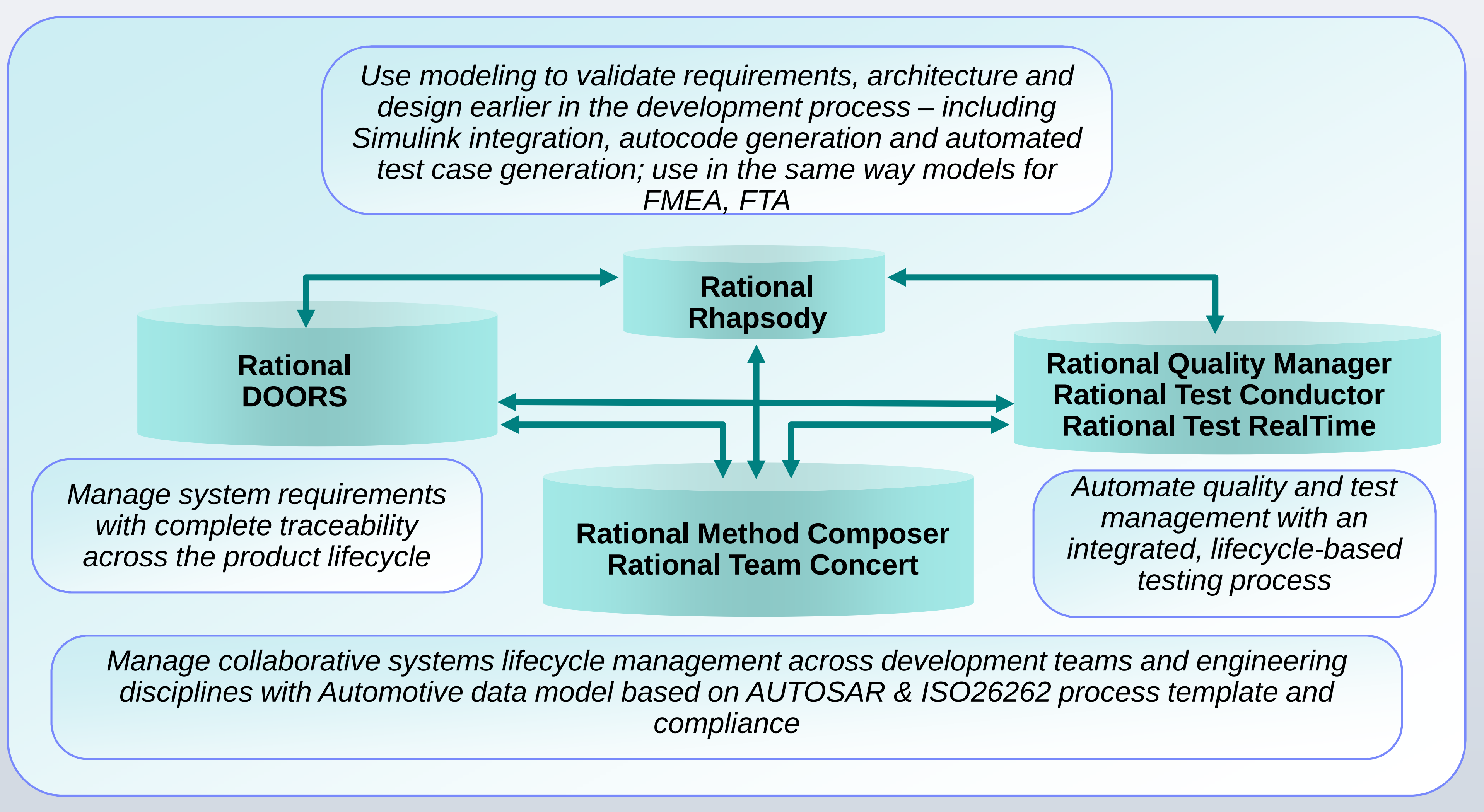
Product Lifecycle Management

Enterprise Resource Planning

Supplier Community

Watson IoT

A look to the inside: How IBM Software Platform for Automotive Systems supports ISO 26262



Use modeling to validate requirements, architecture and design earlier in the development process – including Simulink integration, autocode generation and automated test case generation; use in the same way models for FMEA, FTA

Rational Rhapsody

Rational DOORS

Rational Quality Manager
Rational Test Conductor
Rational Test RealTime

Manage system requirements with complete traceability across the product lifecycle

Rational Method Composer
Rational Team Concert

Automate quality and test management with an integrated, lifecycle-based testing process

Manage collaborative systems lifecycle management across development teams and engineering disciplines with Automotive data model based on AUTOSAR & ISO26262 process template and compliance

Rational software Process and Collaboration support

- Rational Team Concert is the enabler for controlling process and managing change
  - Process template for ISO 26262
    - Helps with project management
    - Team management
    - Task allocation
  - Integrates with practices that give can guidance on the application of ISO 26262
  - Configuration management and Collaboration platform
- Integrates with multiple Rational Tools
  - Rational Method Composer (RMC) for process management
  - Rational DOORS for requirements management
  - Rational Rhapsody for Model Based Systems Engineering
    - Removes system design errors early in the development process
    - Has a safety profile to aid in FMEA, FTA and hazard analysis and integrates with IBM business partner IKV
    - Integrations with IBM business partners INCHRON for timing analysis, and Mathworks for Closed Loop/Plant modeling/Control Law analysis
  - Rational Quality Manager (RQM) for plan tests
  - Rational Test Conductor to automate tests

Enhancements to Process and Collaboration support

- Improved coverage of phases of development
  - Covers all phases of development from ISO 26262 parts 2 to 8 inclusive
  - Detailed workflows taking into account ASIL levels
  - Enhanced detail of task descriptions
  - Enhanced guidance and practices for all phases of development
- Detailed guidance and practices in the areas of
  - Setting up various Rational tools
  - Requirements management using DOORS
  - Implementing change processes and configuration management in Rational Team Concert
  - Implementing model based development for Systems definition and Software development in Rhapsody
  - Implementing model based testing software using TestConductor, Rhapsody and Rational Quality Manager
  - Implementing model based timing analysis using INCHRON's ChronSim
  - Implementing safety analysis with IKV++ Medini Analyse

IBM Solution for Automotive to support ISO-26262 Compliance

Rational Team Concert

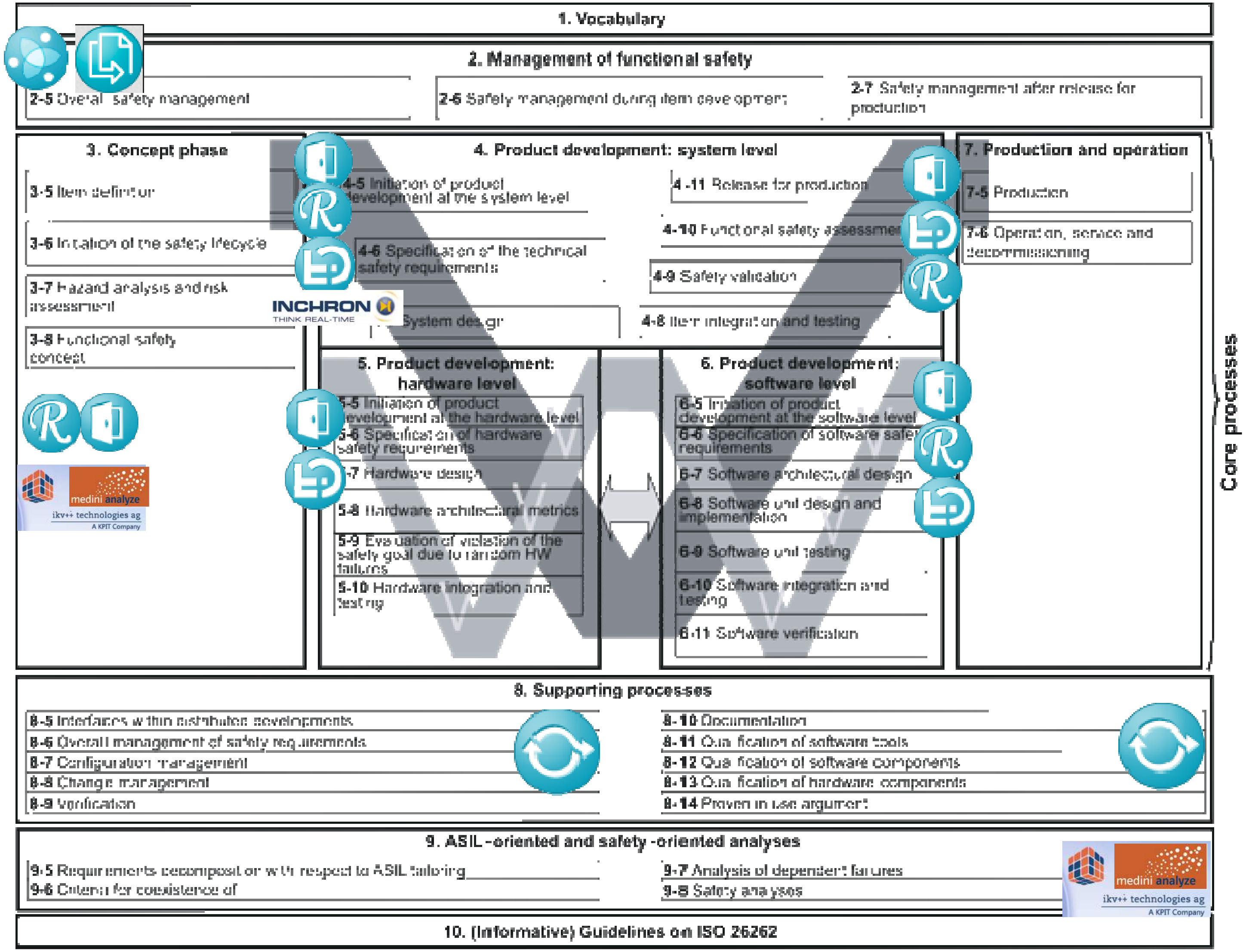
Rational DOORS

Rational Rhapsody and Test Conductor

Rational Quality Manager

Rational Method Composer

Rational Publishing Engine



1. Vocabulary

2. Management of functional safety

2-5 Overall safety management

2-6 Safety management during item development

2-7 Safety management after release for production

3. Concept phase

3-5 Item definition

3-6 Initiation of the safety lifecycle

3-7 Hazard analysis and risk assessment

3-8 Functional safety concept

4. Product development: system level

4-5 Initiation of product development at the system level

4-6 Specification of the technical safety requirements

System design

4-8 Item integration and testing

4-9 Safety validation

4-10 Functional safety assessment

4-11 Release for production

5. Product development: hardware level

5-5 Initiation of product development at the hardware level

5-6 Specification of hardware safety requirements

5-7 Hardware design

5-8 Hardware architectural metrics

5-9 Evaluation of violation of the safety goal due to random HW failures

5-10 Hardware integration and testing

6. Product development: software level

6-5 Initiation of product development at the software level

6-6 Specification of software safety requirements

6-7 Software architectural design

6-8 Software unit design and implementation

6-9 Software unit testing

6-10 Software integration and testing

6-11 Software verification

7. Production and operation

7-5 Production

7-6 Operation, service and decommissioning

Core processes

8. Supporting processes

8-5 Interfaces within distributed developments

8-6 Overall management of safety requirements

8-7 Configuration management

8-8 Change management

8-9 Verification

8-10 Documentation

8-11 Qualification of software tools

8-12 Qualification of software components

8-13 Qualification of hardware components

8-14 Proven in use argument

9. ASIL-oriented and safety-oriented analyses

9-5 Requirements decomposition with respect to ASIL tailoring

9-6 Criteria for coexistence of

9-7 Analysis of dependent failures

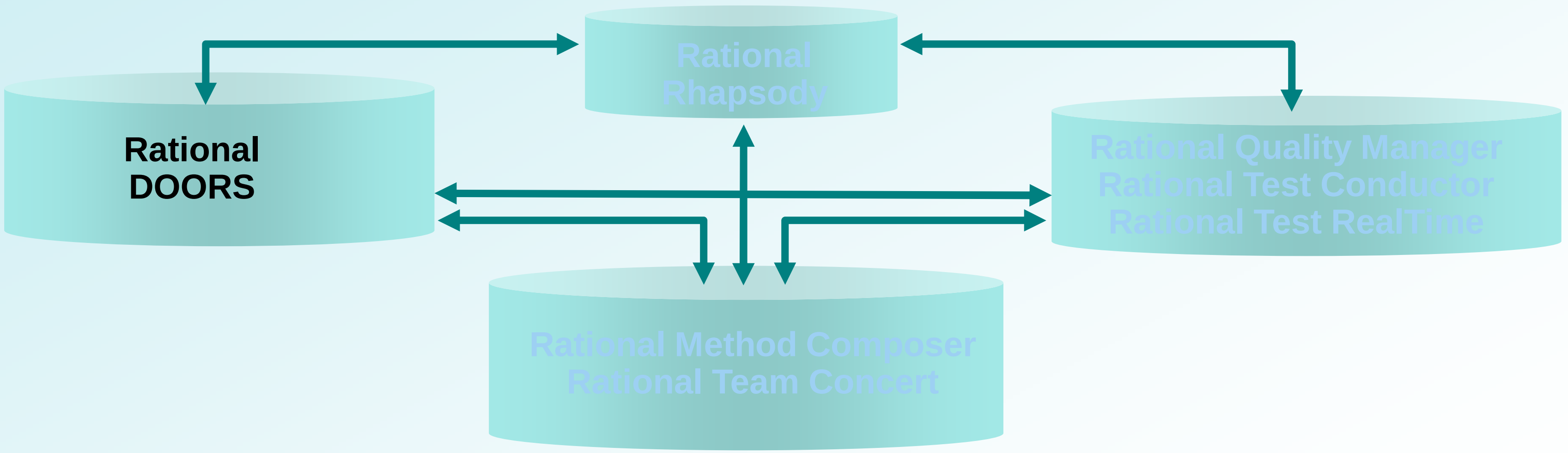
9-8 Safety analyses

10. (Informative) Guidelines on ISO 26262

Original image courtesy of ISO-26262 specification

A look to the inside: How IBM Platform for Automotive Systems supports ISO 26262

Requirements Engineering



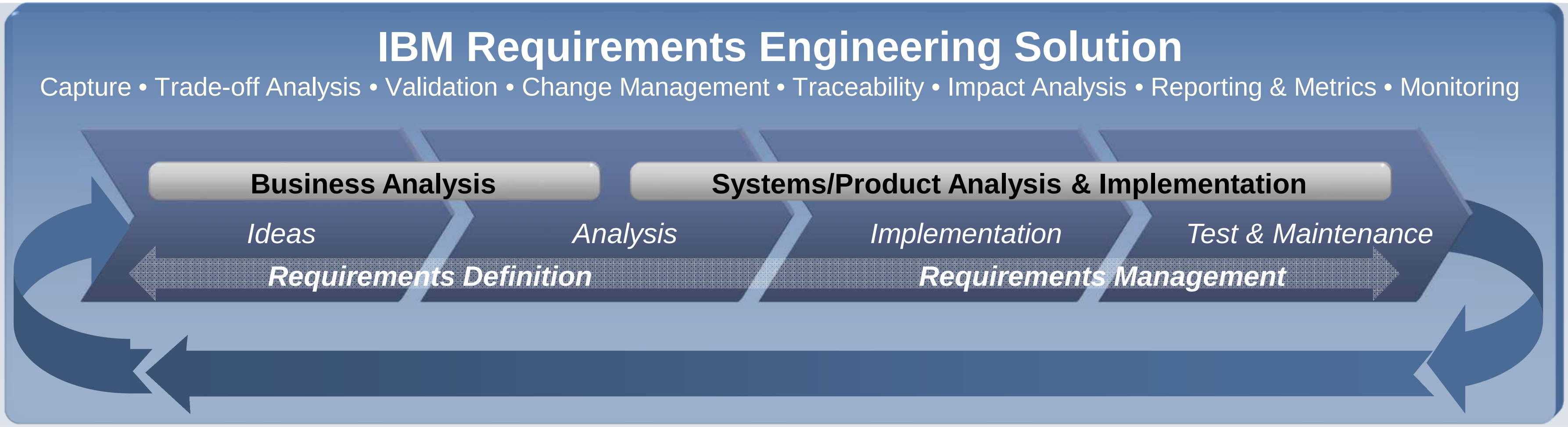
Rational DOORS

Rational Rhapsody

Rational Quality Manager
Rational Test Conductor
Rational Test RealTime

Rational Method Composer
Rational Team Concert

Quality begins with Requirements: IBM Requirements Engineering Solution

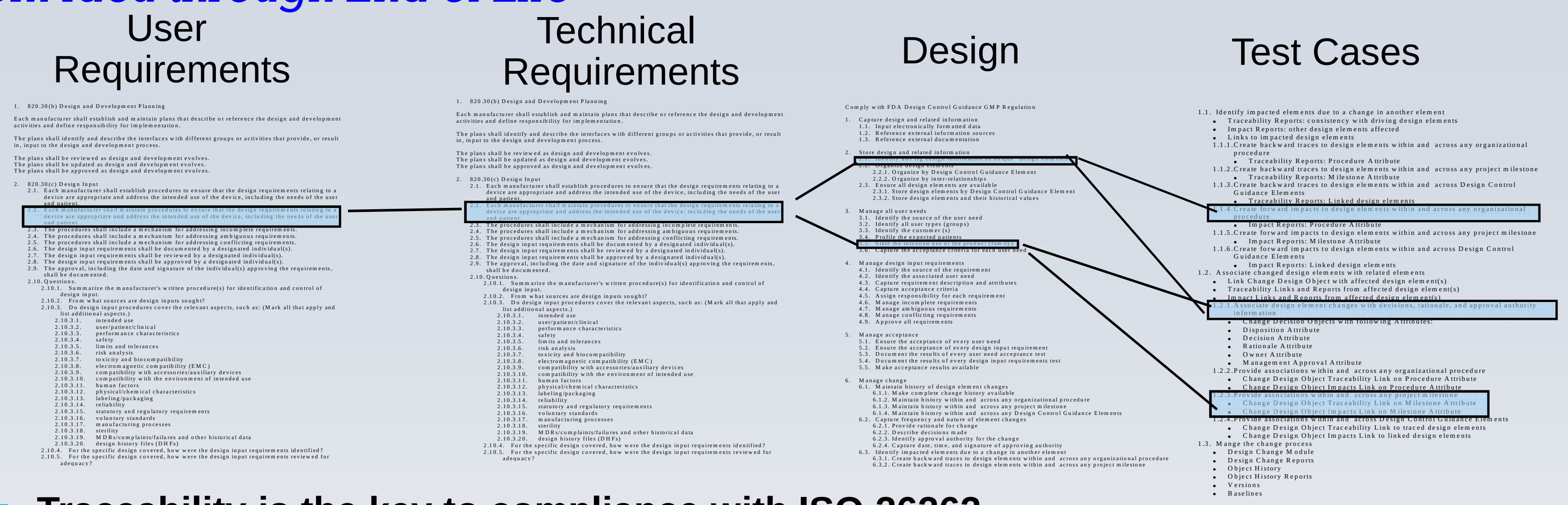

IBM Requirements Engineering Solution
Capture • Trade-off Analysis • Validation • Change Management • Traceability • Impact Analysis • Reporting & Metrics • Monitoring
Business Analysis
Systems/Product Analysis & Implementation
Ideas
Analysis
Implementation
Test & Maintenance
Requirements Definition
Requirements Management

- Getting everyone on the same page
  - Includes suppliers and subcontractors
- Managing scope, plus assessing and controlling the impact of change
- Ensuring end-to-end traceability
  - From ideas, feature definitions, product specifications and models…
  - To mechanical, electric/electronic and embedded software implementation, test and maintenance
- Ensuring conformance to contractual agreements
- **From ISO 26262 Part 8, 2011, paragraph 6.2 (Requirements, DOORS)**

In order to support the management of safety requirements, the use of suitable requirements management tools is recommended.

Requirements Management Must Provide Lifecycle Traceability

From Idea through End of Life



User Requirements

1. 820.30(b) Design and Development Planning

Each manufacturer shall establish and maintain plans that describe or reference the design and development activities and define responsibility for implementation.

The plans shall identify and describe the interfaces with different groups or activities that provide, or result in, input to the design and development process.

The plans shall be reviewed as design and development evolves.
The plans shall be updated as design and development evolves.
The plans shall be approved as design and development evolves.

2. 820.30(c) Design Input
2.1. Each manufacturer shall establish procedures to ensure that the design requirements relating to a device are appropriate and address the intended use of the device, including the needs of the user and patient.
2.2. Each manufacturer shall maintain procedures to ensure that the design requirements relating to a device are appropriate and address the intended use of the device, including the needs of the user and patient.
2.3. The procedures shall include a mechanism for addressing incomplete requirements.
2.4. The procedures shall include a mechanism for addressing ambiguous requirements.
2.5. The procedures shall include a mechanism for addressing conflicting requirements.
2.6. The design input requirements shall be documented by a designated individual(s).
2.7. The design input requirements shall be reviewed by a designated individual(s).
2.8. The design input requirements shall be approved by a designated individual(s).
2.9. The approval, including the date and signature of the individual(s) approving the requirements, shall be documented.
2.10. Questions.
2.10.1. Summarize the manufacturer's written procedure(s) for identification and control of design input.
2.10.2. From what sources are design inputs sought?
2.10.3. Do design input procedures cover the relevant aspects, such as: (Mark all that apply and list additional aspects.)
2.10.3.1. intended use
2.10.3.2. user/patient/clinical
2.10.3.3. performance characteristics
2.10.3.4. safety
2.10.3.5. limits and tolerances
2.10.3.6. risk analysis
2.10.3.7. toxicity and biocompatibility
2.10.3.8. electromagnetic compatibility (EMC)
2.10.3.9. compatibility with accessories/auxiliary devices
2.10.3.10. compatibility with the environment of intended use
2.10.3.11. human factors
2.10.3.12. physical/chemical characteristics
2.10.3.13. labeling/packaging
2.10.3.14. reliability
2.10.3.15. statutory and regulatory requirements
2.10.3.16. voluntary standards
2.10.3.17. manufacturing processes
2.10.3.18. sterility
2.10.3.19. MDRs/complaints/failures and other historical data
2.10.3.20. design history files (DHFs)
2.10.4. For the specific design covered, how were the design input requirements identified?
2.10.5. For the specific design covered, how were the design input requirements reviewed for adequacy?

Technical Requirements

1. 820.30(b) Design and Development Planning

Each manufacturer shall establish and maintain plans that describe or reference the design and development activities and define responsibility for implementation.

The plans shall identify and describe the interfaces with different groups or activities that provide, or result in, input to the design and development process.

The plans shall be reviewed as design and development evolves.
The plans shall be updated as design and development evolves.
The plans shall be approved as design and development evolves.

2. 820.30(c) Design Input
2.1. Each manufacturer shall establish procedures to ensure that the design requirements relating to a device are appropriate and address the intended use of the device, including the needs of the user and patient.
2.2. Each manufacturer shall maintain procedures to ensure that the design requirements relating to a device are appropriate and address the intended use of the device, including the needs of the user and patient.
2.3. The procedures shall include a mechanism for addressing incomplete requirements.
2.4. The procedures shall include a mechanism for addressing ambiguous requirements.
2.5. The procedures shall include a mechanism for addressing conflicting requirements.
2.6. The design input requirements shall be documented by a designated individual(s).
2.7. The design input requirements shall be reviewed by a designated individual(s).
2.8. The design input requirements shall be approved by a designated individual(s).
2.9. The approval, including the date and signature of the individual(s) approving the requirements, shall be documented.
2.10. Questions.
2.10.1. Summarize the manufacturer's written procedure(s) for identification and control of design input.
2.10.2. From what sources are design inputs sought?
2.10.3. Do design input procedures cover the relevant aspects, such as: (Mark all that apply and list additional aspects.)
2.10.3.1. intended use
2.10.3.2. user/patient/clinical
2.10.3.3. performance characteristics
2.10.3.4. safety
2.10.3.5. limits and tolerances
2.10.3.6. risk analysis
2.10.3.7. toxicity and biocompatibility
2.10.3.8. electromagnetic compatibility (EMC)
2.10.3.9. compatibility with accessories/auxiliary devices
2.10.3.10. compatibility with the environment of intended use
2.10.3.11. human factors
2.10.3.12. physical/chemical characteristics
2.10.3.13. labeling/packaging
2.10.3.14. reliability
2.10.3.15. statutory and regulatory requirements
2.10.3.16. voluntary standards
2.10.3.17. manufacturing processes
2.10.3.18. sterility
2.10.3.19. MDRs/complaints/failures and other historical data
2.10.3.20. design history files (DHFs)
2.10.4. For the specific design covered, how were the design input requirements identified?
2.10.5. For the specific design covered, how were the design input requirements reviewed for adequacy?

Design

Comply with FDA Design Control Guidance GMP Regulation

1. Capture design and related information
1.1. Input electronically formatted data
1.2. Reference external information sources
1.3. Reference external documentation

2. Store design and related information
2.1. Identify and tag design information as unique design elements
2.2. Organize design elements
2.2.1. Organize by Design Control Guidance Element
2.2.2. Organize by inter-relationships
2.3. Ensure all design elements are available
2.3.1. Store design elements by Design Control Guidance Element
2.3.2. Store design elements and their historical values

3. Manage all user needs
3.1. Identify the source of the user need
3.2. Identify all user types (groups)
3.3. Identify the customer (s)
3.4. Profile the expected patients
3.5. State the intended use of the product (family)
3.6. Capture the acceptance criteria for each user need

4. Manage design input requirements
4.1. Identify the source of the requirement
4.2. Identify the associated user need
4.3. Capture requirement description and attributes
4.4. Capture acceptance criteria
4.5. Assign responsibility for each requirement
4.6. Manage incomplete requirements
4.7. Manage ambiguous requirements
4.8. Manage conflicting requirements
4.9. Approve all requirements

5. Manage acceptance
5.1. Ensure the acceptance of every user need
5.2. Ensure the acceptance of every design input requirement
5.3. Document the results of every user need acceptance test
5.4. Document the results of every design input requirements test
5.5. Make acceptance results available

6. Manage change
6.1. Maintain history of design element changes
6.1.1. Make complete change history available
6.1.2. Maintain history within and across any organizational procedure
6.1.3. Maintain history within and across any project milestone
6.1.4. Maintain history within and across any Design Control Guidance Elements
6.2. Capture frequency and nature of element changes
6.2.1. Provide rationale for change
6.2.2. Describe decisions made
6.2.3. Identify approval authority for the change
6.2.4. Capture date, time, and signature of approving authority
6.3. Identify impacted elements due to a change in another element
6.3.1. Create backward traces to design elements within and across any organizational procedure
6.3.2. Create backward traces to design elements within and across any project milestone

Test Cases

1.1. Identify impacted elements due to a change in another element
- Traceability Reports: consistency with driving design elements
- Impact Reports: other design elements affected
- Links to impacted design elements

1.1.1. Create backward traces to design elements within and across any organizational procedure
- Traceability Reports: Procedure Attribute

1.1.2. Create backward traces to design elements within and across any project milestone
- Traceability Reports: Milestone Attribute

1.1.3. Create backward traces to design elements within and across Design Control Guidance Elements
- Traceability Reports: Linked design elements

1.1.4. Create forward impacts to design elements within and across any organizational procedure
- Impact Reports: Procedure Attribute

1.1.5. Create forward impacts to design elements within and across any project milestone
- Impact Reports: Milestone Attribute

1.1.6. Create forward impacts to design elements within and across Design Control Guidance Elements
- Impact Reports: Linked design elements

1.2. Associate changed design elements with related elements
- Link Change Design Object with affected design element(s)
- Traceability Links and Reports from affected design element(s)
- Impact Links and Reports from affected design element(s)

1.2.1. Associate design element changes with decisions, rationale, and approval authority information
- Change Decision Objects with following Attributes:
- Disposition Attribute
- Decision Attribute
- Rationale Attribute
- Owner Attribute
- Management Approval Attribute

1.2.2. Provide associations within and across any organizational procedure
- Change Design Object Traceability Link on Procedure Attribute
- Change Design Object Impacts Link on Procedure Attribute

1.2.3. Provide associations within and across any project milestone
- Change Design Object Traceability Link on Milestone Attribute
- Change Design Object Impacts Link on Milestone Attribute

1.2.4. Provide associations within and across Design Control Guidance Elements
- Change Design Object Traceability Link to traced design elements
- Change Design Object Impacts Link to linked design elements

1.3. Mange the change process
- Design Change Module
- Design Change Reports
- Object History
- Object History Reports
- Versions
- Baselines

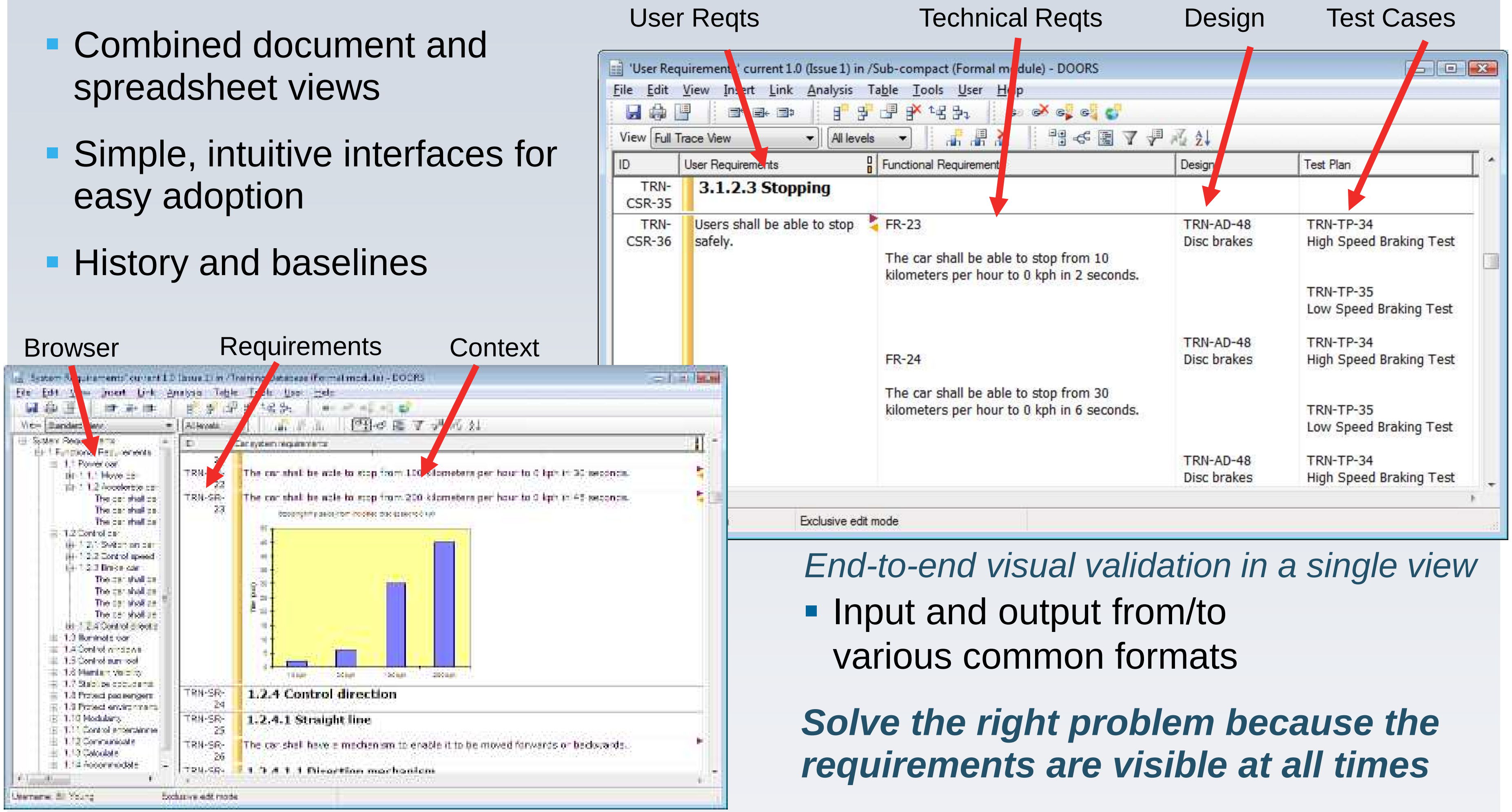
- **Traceability is the key to compliance with ISO 26262**
  - Initial requirements will be decomposed, which creates traceability relationships
  - Other relationships can also be traced such as “consists of”, “verifies”, etc.
  - Traceability must be enforced in order to ensure consistency and completeness

- Traceability from customer requirements through product development to test and delivery enables organizations to:
  - Know which requirements are implemented and tested vs. those which are not
  - Manage and defend against scope creep

IBM Rational DOORS

Manage All Requirements Across the Lifecycle and Across Disciplines

- Combined document and spreadsheet views
- Simple, intuitive interfaces for easy adoption
- History and baselines


User Reqts
Technical Reqts
Design
Test Cases
'User Requirements' current 1.0 (Issue 1) in /Sub-compact (Formal module) - DOORS
File Edit View Insert Link Analysis Table Tools User Help
View Full Trace View
All levels
Exclusive edit mode
Browser
Requirements
Context
File Edit View Insert Link Analysis Table Tools User Help
View Standard view
All levels
Username: Bill Young
Exclusive edit mode

ID	User Requirements	Functional Requirement	Design	Test Plan
TRN-CSR-35	**3.1.2.3 Stopping**			
TRN-CSR-36	Users shall be able to stop safely.	FR-23 The car shall be able to stop from 10 kilometers per hour to 0 kph in 2 seconds.	TRN-AD-48 Disc brakes	TRN-TP-34 High Speed Braking Test TRN-TP-35 Low Speed Braking Test
		FR-24 The car shall be able to stop from 30 kilometers per hour to 0 kph in 6 seconds.	TRN-AD-48 Disc brakes	TRN-TP-34 High Speed Braking Test TRN-TP-35 Low Speed Braking Test
			TRN-AD-48 Disc brakes	TRN-TP-34 High Speed Braking Test

System Requirements
1 Functional Requirements
1.1 Power car
1.1.1 Move car
1.1.2 Accelerate car
The car shall be
The car shall be
The car shall be
1.2 Control car
1.2.1 Switch on car
1.2.2 Control speed
1.2.3 Brake car
The car shall be
The car shall be
The car shall be
The car shall be
1.2.4 Control directio
1.3 Illuminate car
1.4 Control windows
1.5 Control sun roof
1.6 Maintain visibility
1.7 Stabilise occupants
1.8 Protect passengers
1.9 Protect environment
1.10 Modularity
1.11 Control entertainme
1.12 Communicate
1.13 Calculate
1.14 Accommodate

ID	Car system requirements
TRN-SR-22	The car shall be able to stop from 100 kilometers per hour to 0 kph in 30 seconds.
TRN-SR-23	The car shall be able to stop from 200 kilometers per hour to 0 kph in 45 seconds.
TRN-SR-24	**1.2.4 Control direction**
TRN-SR-25	**1.2.4.1 Straight line**
TRN-SR-26	The car shall have a mechanism to enable it to be moved forwards or backwards.
TRN-SR-	**1.2.4.1.1 Direction mechanism**

Writing Requirements within Context

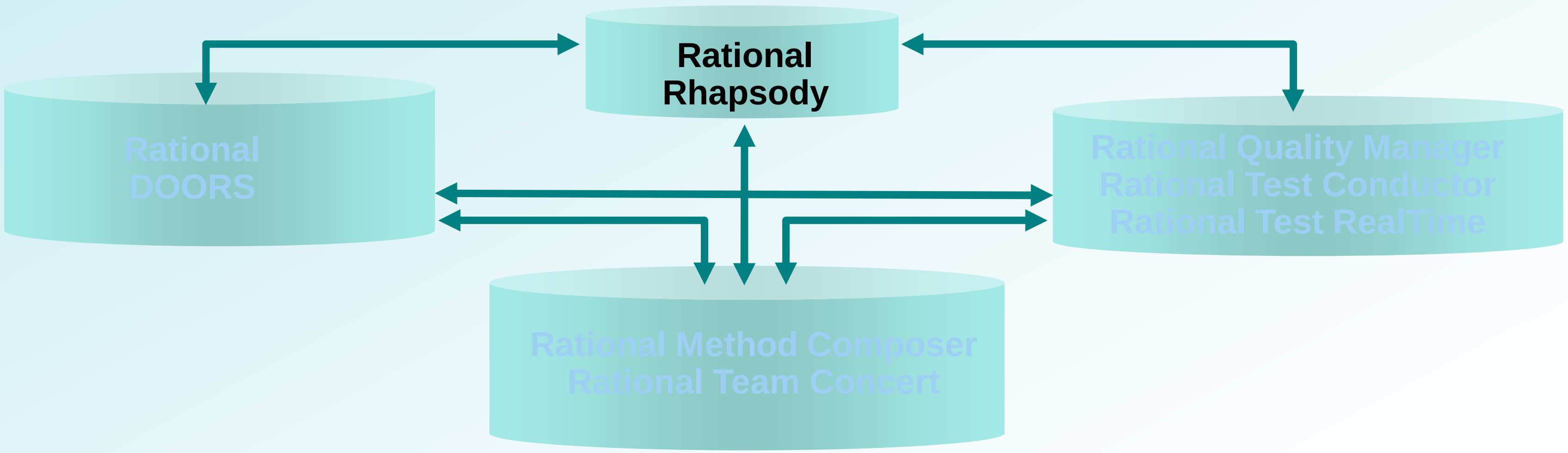
End-to-end visual validation in a single view

- Input and output from/to various common formats

Solve the right problem because the requirements are visible at all times

A look to the inside: How IBM Platform for Automotive Systems supports ISO 26262

A modelling approach for a safety based development lifecycle



Rational Rhapsody

Rational DOORS

Rational Quality Manager
Rational Test Conductor
Rational Test RealTime

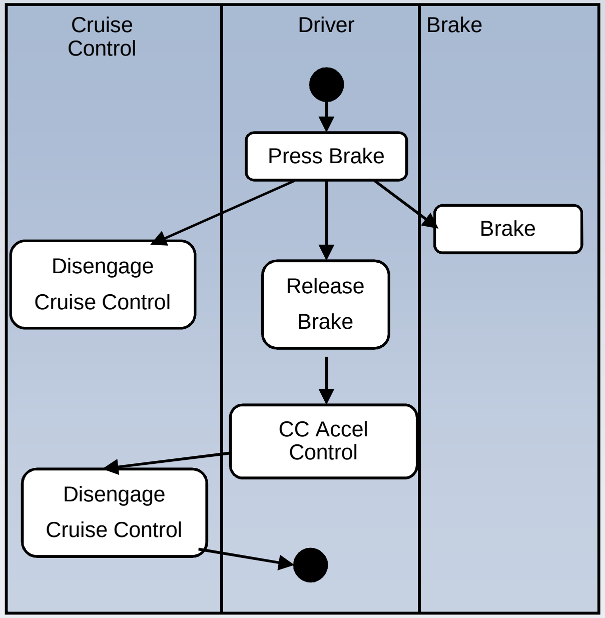
Rational Method Composer
Rational Team Concert

Modern Approaches for Describing Systems Are Evolving

To Better Manage Complexity and Reduce Time-to-market

Past

Future



Cruise Control

Driver

Brake

Press Brake

Brake

Disengage Cruise Control

Release Brake

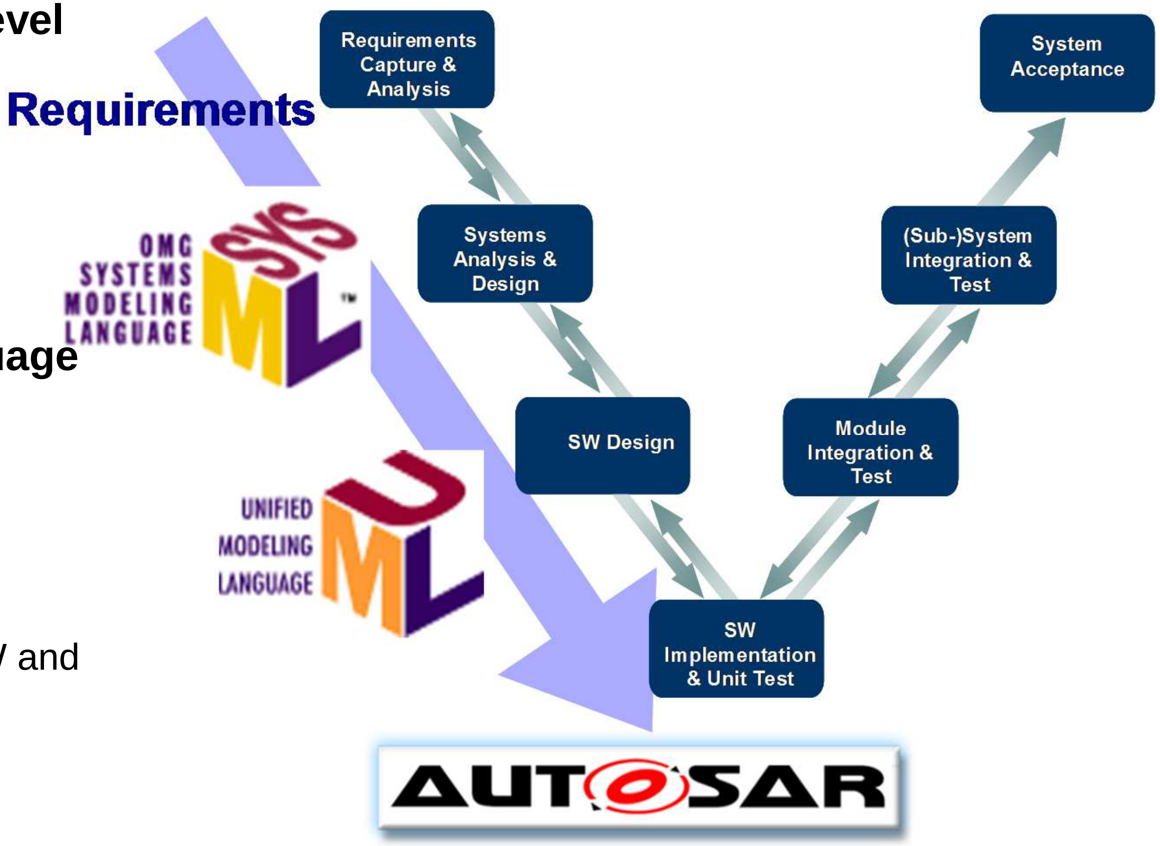
CC Accel Control

Disengage Cruise Control

Moving from Document-centric to Model-centric

Allowing abstraction, hierarchies and modularization with domain-focused, standards-based languages

- **SysML – Systems Modeling Language for modeling high-level**
  - vehicle functions
  - logical and technical architecture
  - vehicle and E/E system behavior
- **UML – Unified Modeling Language for modeling**
  - ECU and SW architecture
  - Client-specific profiles
- **AUTOSAR**
  - Detailed E/E System and ECU HW and SW architecture


Requirements
Requirements Capture & Analysis
System Acceptance
OMG SYSTEMS MODELING LANGUAGE
SYS ML
Systems Analysis & Design
(Sub-)System Integration & Test
SW Design
Module Integration & Test
UNIFIED MODELING LANGUAGE
UML
SW Implementation & Unit Test
AUTOSAR

Maximize *your* budget & boost productivity with effective *Systems Engineering*

IBM Rational Rhapsody® software Family

Safety driven Systems Design

- Understand Safety requirements early in the development cycle
- Design safety into the system to begin with

Model Driven Testing:

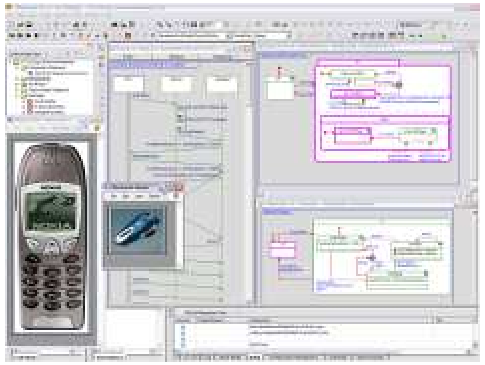
- Bring the benefits of abstraction and automation to testing
- Deliver products that meet customer expectations faster, cheaper

Simulation, Execution and Automation:

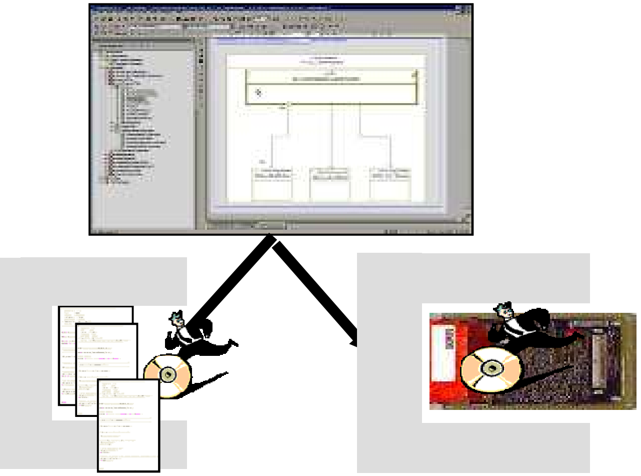
- Identify and eliminate errors early when they are less costly to fix
- Visually communicate intended behavior to customers to deliver the right product
- Perform design level debugging

Requirements Driven Testing:

- Reduce overall dev costs by dramatically reducing time in the testing phase
- Automatic regression testing, Change impact and analysis



Simulation

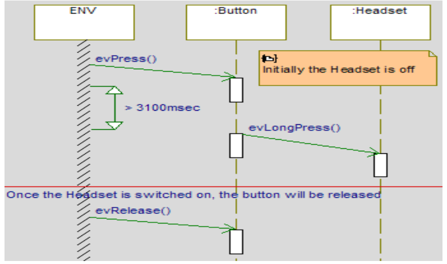


Host based

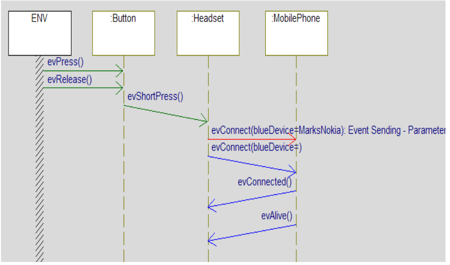
Target based

Automated unit testing

Sequence Diagrams


ENV
:Button
:Headset
evPress()
Initially the Headset is off
> 3100msec
evLongPress()
Once the Headset is switched on, the button will be released
evRelease()

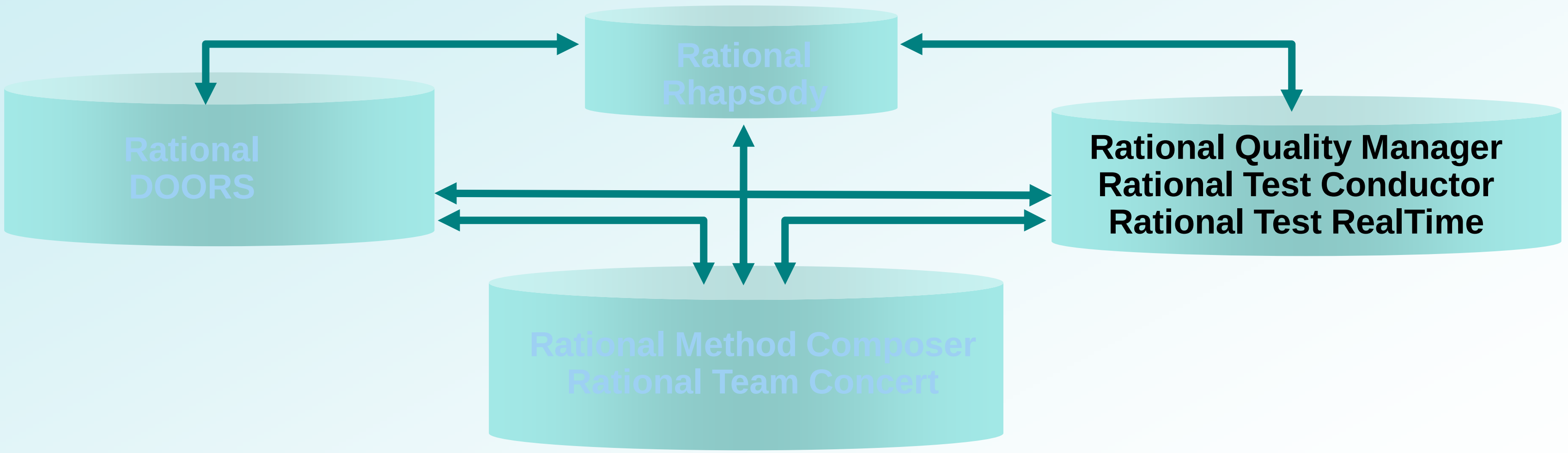
Finding & Correcting Errors



Watson IoT

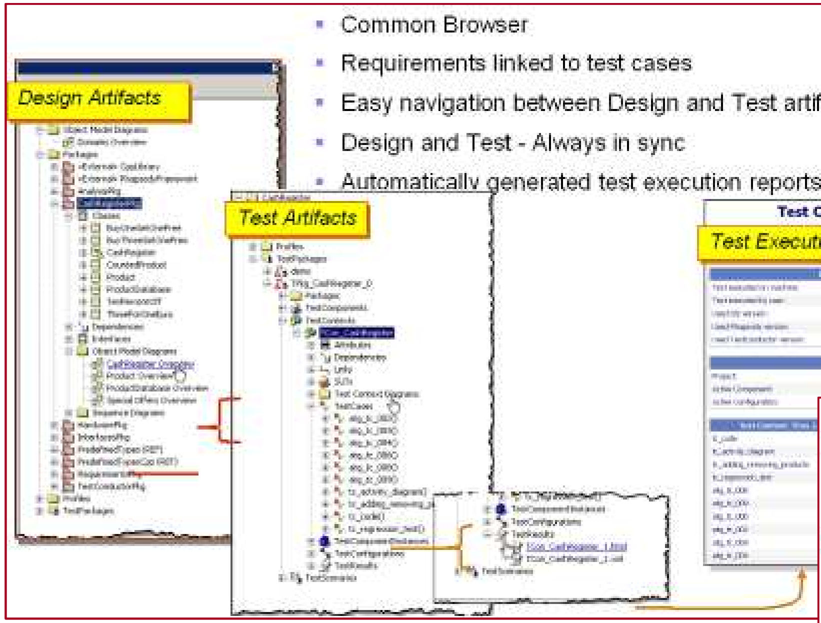
A look to the inside: How IBM Platform for Automotive Systems supports ISO 26262

Quality Management



Model Driven Testing

IBM Rhapsody Test Conductor (qualified by TüV for ISO 26262/IEC 61508)



- Common Browser
- Requirements linked to test cases
- Easy navigation between Design and Test artifacts;
- Design and Test - Always in sync
- Automatically generated test execution reports

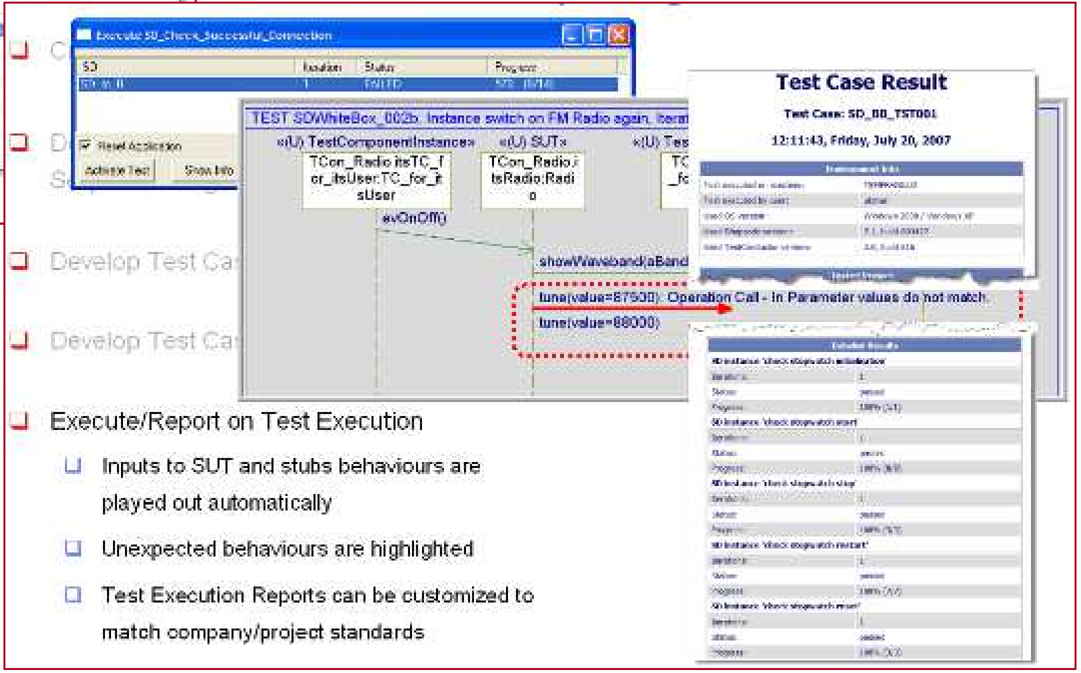
Test Execution & Test Reporting

Practices/guidance for test management and RTC, DOORS, RQM integration

Design & Test Processes Fully Integrated

Use for AUTOSAR SWC test

Practices for Back to back and Requirements based testing

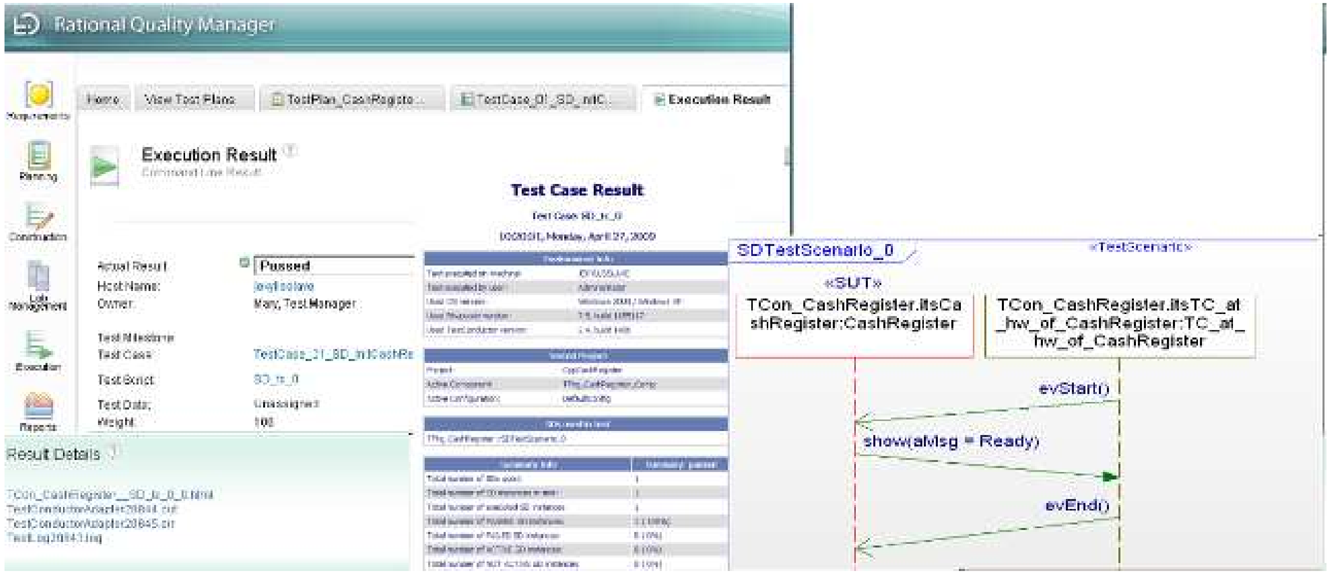

Execute SD_Check_Successful_Connection
SD
Iteration
Status
Progress
1
FAILED
Reset Application
Activate Test
Show Info
TEST SDWhiteBox_002b: Instance switch on FM Radio again, iterat
«(U) TestComponentInstance»
«(U) SUT»
«(U) Tes
TCon_Radio itsTC_for_itsUser:TC_for_itsUser
TCon_Radio::itsRadio:Radio
evOnOff()
showWaveband(aBand
tune(value=87500): Operation Call - In Parameter values do not match.
tune(value=88000)
Test Case Result
Test Case: SD_BB_TST001
12:11:43, Friday, July 20, 2007
Develop Test Ca
Develop Test Ca

- Execute/Report on Test Execution
  - Inputs to SUT and stubs behaviours are played out automatically
  - Unexpected behaviours are highlighted
  - Test Execution Reports can be customized to match company/project standards

Watson IoT

Rational Rhapsody TestConductor integration with Rational Quality Manager

- Enables full execution control & management of model based Rhapsody TestConductor test cases from RQM
- Execution status (passed/failed) and result reports (Execution Results, Coverage Results) accessible through RQM
- RQM can utilize TestConductor execution results to continuously provide transparent & up to date QA statistics and QA reports
- Test conductor now has a certificate for use with Rhapsody on ISO 26262/IEC 61508 projects from the TUV



Execution Result

Commercial Line Result

Actual Result: Passed

Host Name: [illegible]

Owner: Mary, Test Manager

Test Milestone:

Test Case: TestCase_01_SD_inICashRe

Test Script: SD_tc_0

Test Data: Unassigned

Weight: 100

Result Details

TCon_CashRegister__SD_tc_0_0.html
TestConductorAdapter20844.out
TestConductorAdapter20845.err
TestLog20841.log

Test Case Result

Test Case: SD_tc_0

[illegible], Monday, April 27, 2009

[illegible]	
Test executed on machine:	[illegible]
Test executed by user:	Administrator
Used OS version:	Windows 2000 / Windows XP
Used Rhapsody version:	7.5 build [illegible]
Used TestConductor version:	2.4 build [illegible]

[illegible]	
Project:	CppCashRegister
Active Component:	TPkg_CashRegister_Comp
Active Configuration:	DefaultConfig

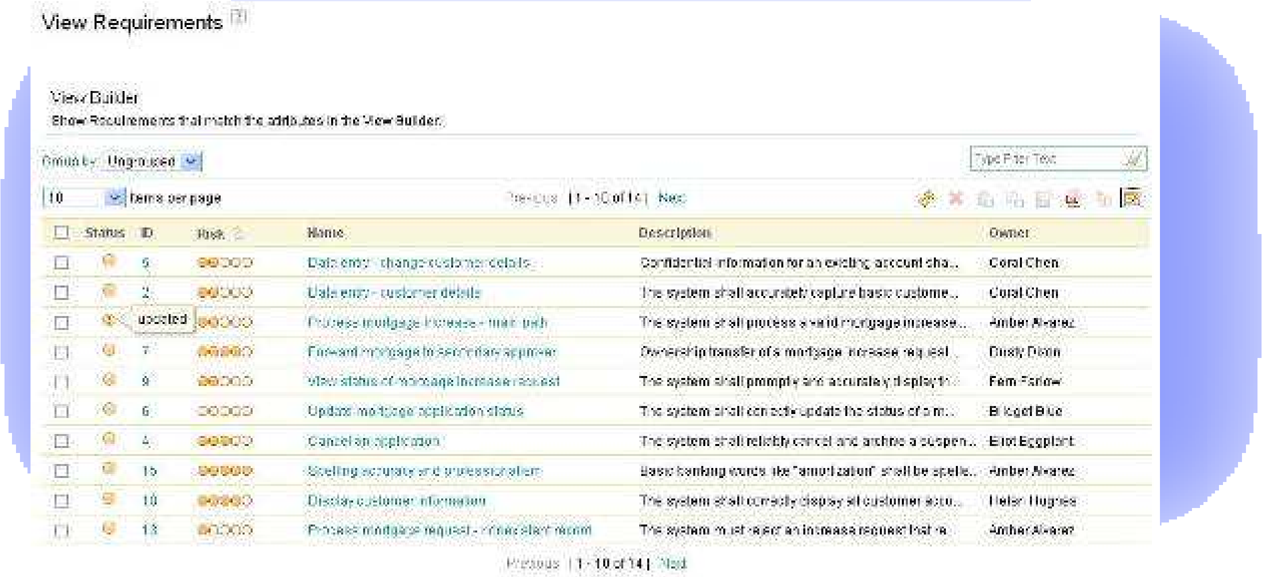
[illegible]
TPkg_CashRegister::SDTestScenario_0

[illegible]	[illegible]
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 [illegible] SD instances:	0 (0%)
Total number of NOT ACTIVE SD instances:	0 (0%)

Watson IoT

Requirements driven testing

Knowing what to test



View Requirements

View Builder

Show Requirements that match the attributes in the View Builder.

Group by: Ungrouped

10 items per page

Previous | 1 - 10 of 14 | Next

□	Status	ID	Risk	Name	Description	Owner
□		5	●●○○○	Data entry - change customer details	Confidential information for an existing account cha...	Coral Chen
□		2	●●○○○	Data entry - customer details	The system shall accurately capture basic custome...	Coral Chen
□	updated		●●○○○	Process mortgage increase - main path	The system shall process a valid mortgage increase...	Amber Alvarez
□		7	●●●●○	Forward mortgage to secondary approver	Ownership transfer of a mortgage increase request...	Dusty Dixon
□		9	●●○○○	View status of mortgage increase request	The system shall promptly and accurately display th...	Fern Farlow
□		6	○○○○○	Update mortgage application status	The system shall correctly update the status of a m...	Bridget Blue
□		4	●●●○○	Cancel an application	The system shall reliably cancel and archive a suspen...	Eliot Eggplant
□		15	●●●●●	Spelling accuracy and professionalism	Basic banking words like "amortization" shall be spelle...	Amber Alvarez
□		10	●●●●○	Display customer information	The system shall correctly display all customer accu...	Helen Hughes
□		13	●○○○○	Process mortgage request - reject client record	The system must reject an increase request that re...	Amber Alvarez

Previous | 1 - 10 of 14 | Next

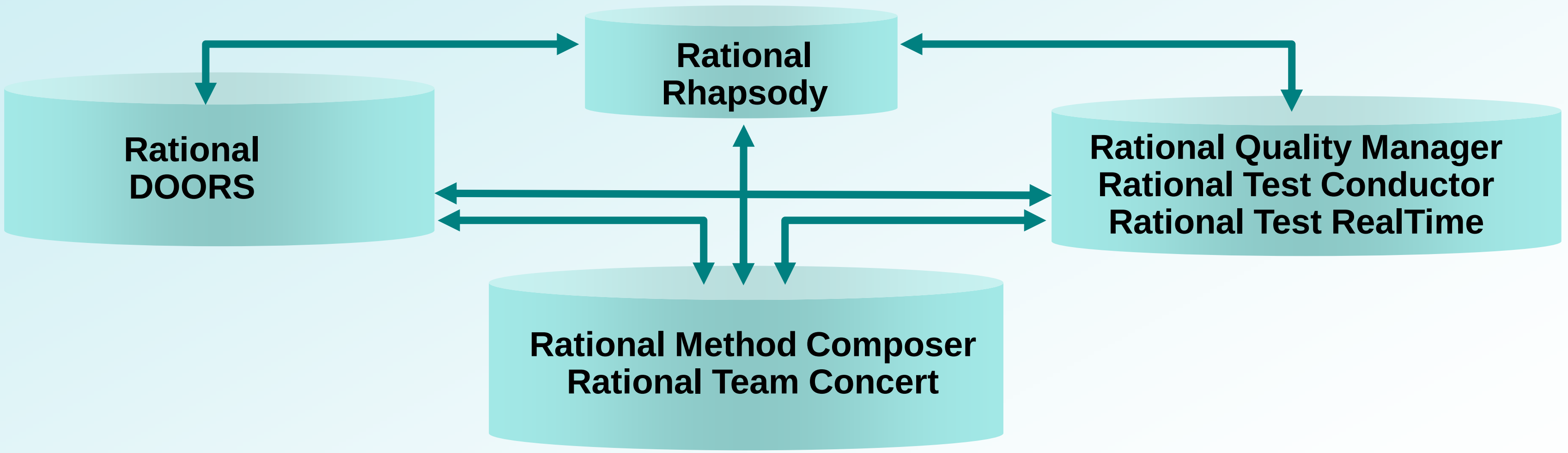
- Requirements tracking built into the test management tooling
- Customizable attributes enable you to track what is important to your team

- Real-time impact analysis of requirements changes
- Traceability of test results to user needs

Know you are testing the right things

A look to the inside: How IBM Platform for Automotive Systems supports ISO 26262

An integrated support ISO 26262



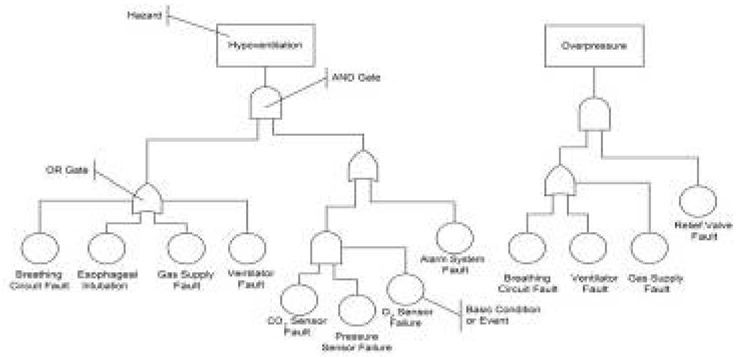
Rational Rhapsody

Rational DOORS

Rational Quality Manager
Rational Test Conductor
Rational Test RealTime

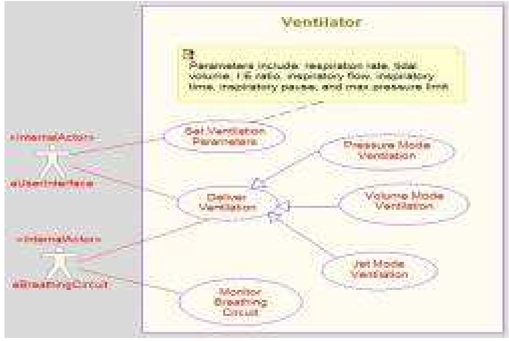
Rational Method Composer
Rational Team Concert

Integrate Safety Design into Design from the beginning



Safety Analysis:

- Fault Tree Analysis (FTA)
- Fault Means and Effective Analysis (FMEA)
- Hazard Analysis



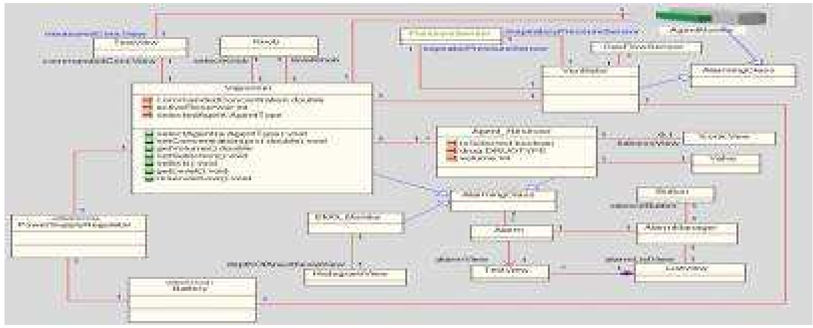
Parameters include: respiration rate, tidal volume, I:E ratio, inspiratory flow, inspiratory time, inspiratory pause, and max pressure limit

The ventilator shall allow settings of inspiratory flow from 1 to 100 L/min with a default of 50 (neonate) or 1 to 180 L/min with a default of 100 (adult).

Inspiratory time shall be settable from 0.1 to 3.0 seconds with a default of 2.0 (neonate) or from 0.1 to 5.0 seconds with a default of 3.0 (adult).

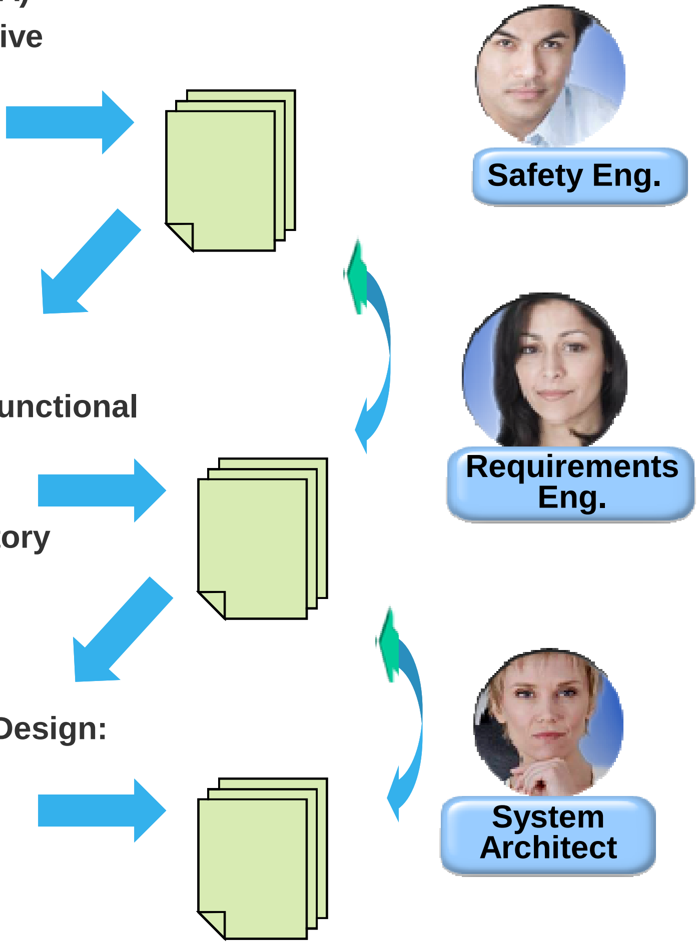
Requirements Analysis:

- Functional and Non-Functional Requirements
- Safety Requirements
- Business and Regulatory Requirements



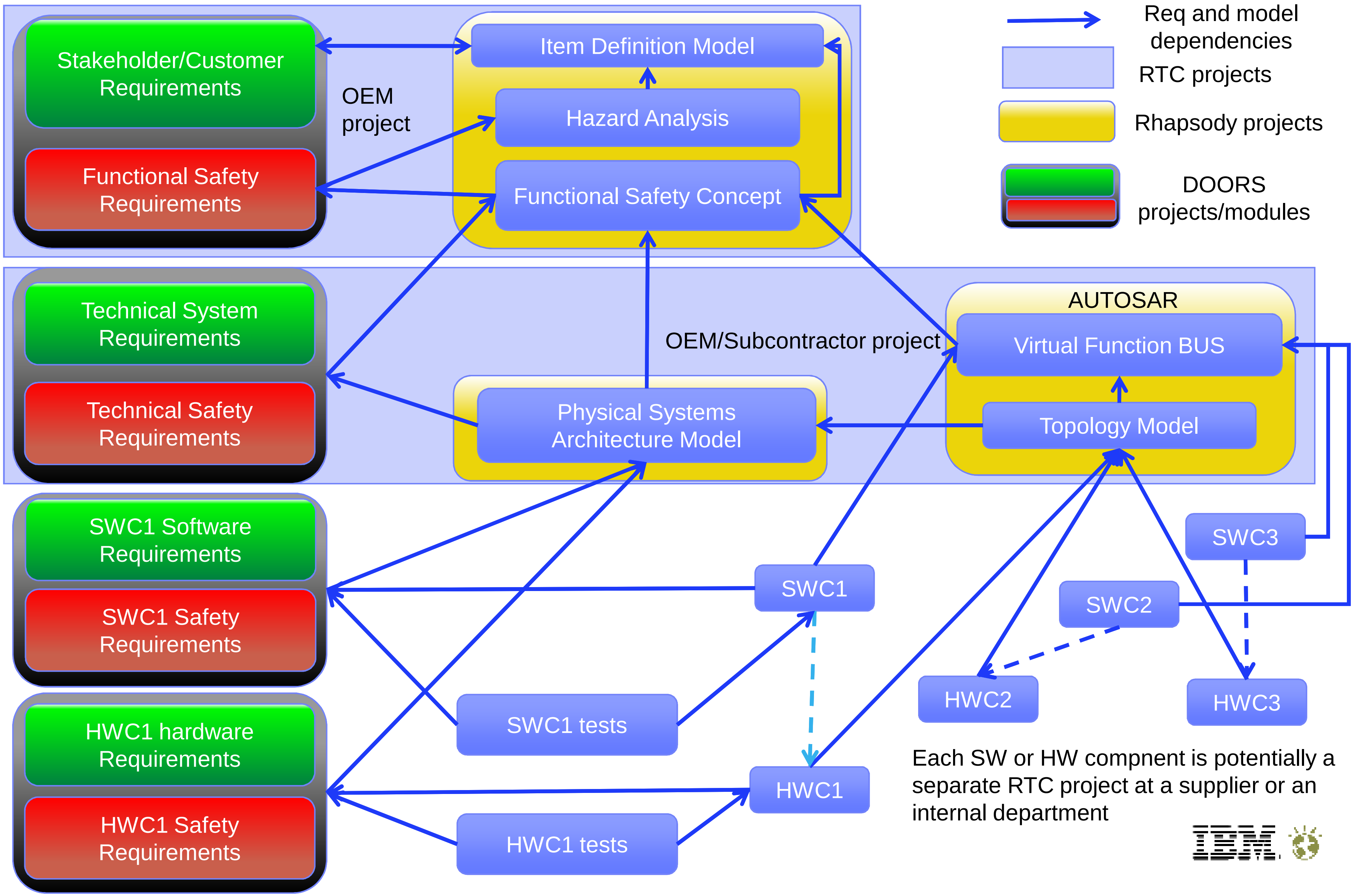
System and Software Design:

- Structural
- Behavioral
- Temporal
- ...



Watson IoT

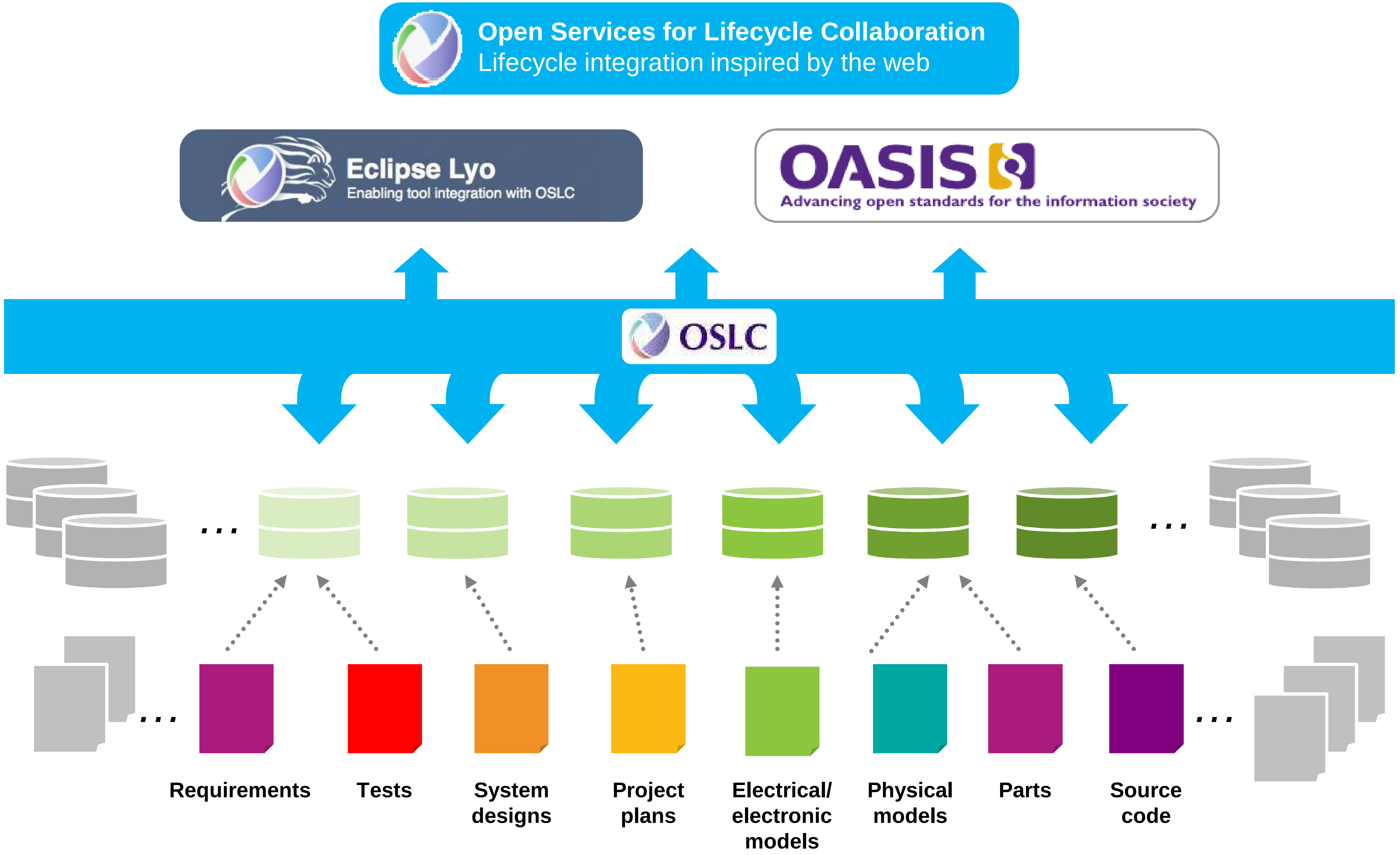
Support for the entire development lifecycle across organizations



Stakeholder/Customer Requirements
Functional Safety Requirements
OEM project
Item Definition Model
Hazard Analysis
Functional Safety Concept
Req and model dependencies
RTC projects
Rhapsody projects
DOORS projects/modules
Technical System Requirements
Technical Safety Requirements
OEM/Subcontractor project
AUTOSAR
Virtual Function BUS
Topology Model
Physical Systems Architecture Model
SWC1 Software Requirements
SWC1 Safety Requirements
HWC1 hardware Requirements
HWC1 Safety Requirements
SWC1
SWC1 tests
HWC1
HWC1 tests
SWC2
SWC3
HWC2
HWC3

Each SW or HW compnent is potentially a separate RTC project at a supplier or an internal department

Realizing an open, connected, systems approach with OSLC

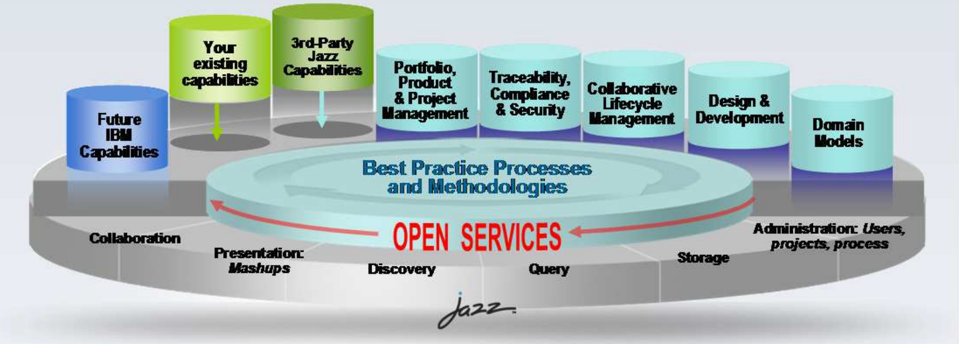

Open Services for Lifecycle Collaboration
Lifecycle integration inspired by the web
Eclipse Lyo
Enabling tool integration with OSLC
OASIS
Advancing open standards for the information society
OSLC
...
...
...
...
Requirements
Tests
System designs
Project plans
Electrical/ electronic models
Physical models
Parts
Source code

IBM Platform for Automotive Systems

Key Results of an E/E engineering lifecycle management

Key Results

- Improved collaboration along the design chain and the entire lifecycle
- Increased productivity due to comprehensive electrical, electronics and software asset management
- Lower cost of safety and regulatory compliance


Future IBM Capabilities
Your existing capabilities
3rd-Party Jazz Capabilities
Portfolio, Product & Project Management
Traceability, Compliance & Security
Collaborative Lifecycle Management
Design & Development
Domain Models
Best Practice Processes and Methodologies
OPEN SERVICES
Collaboration
Presentation: *Mashups*
Discovery
Query
Storage
Administration: *Users, projects, process*
jazz

ISO 26262

AUTOSAR

Automotive SPICE©

Effective product development processes can increase productivity by 40% and reduce defects by 75%

Jaguar Land Rover cuts software validation time by up to 90 percent

IBM Rational software helps manage near-real-time requirements updates for its developers worldwide

The need:

Jaguar Land Rover plc wanted to implement a standard requirements methodology that would enable it to manage increasingly complex requirements and speed new features to market.

The solution:

The company implemented a suite of IBM® Rational® software to create a new requirements management and modeling system.

The benefits:

- Reduces the time required to fully validate software by up to 90 percent, from six to eight weeks to only three days
- Decreases bug-cause detection time by up to 90 percent, from three days of human intervention to 30 seconds
- Accelerates time to market for in-vehicle entertainment systems and helps increase innovation

Solution components:

- IBM Rational DOORS®
- IBM Rational Rhapsody®

www.ibm.com/ibm/continuousengineering

Watson IoT

IBM