



IBM Rational Software Conference 2009
As Real as It Gets!

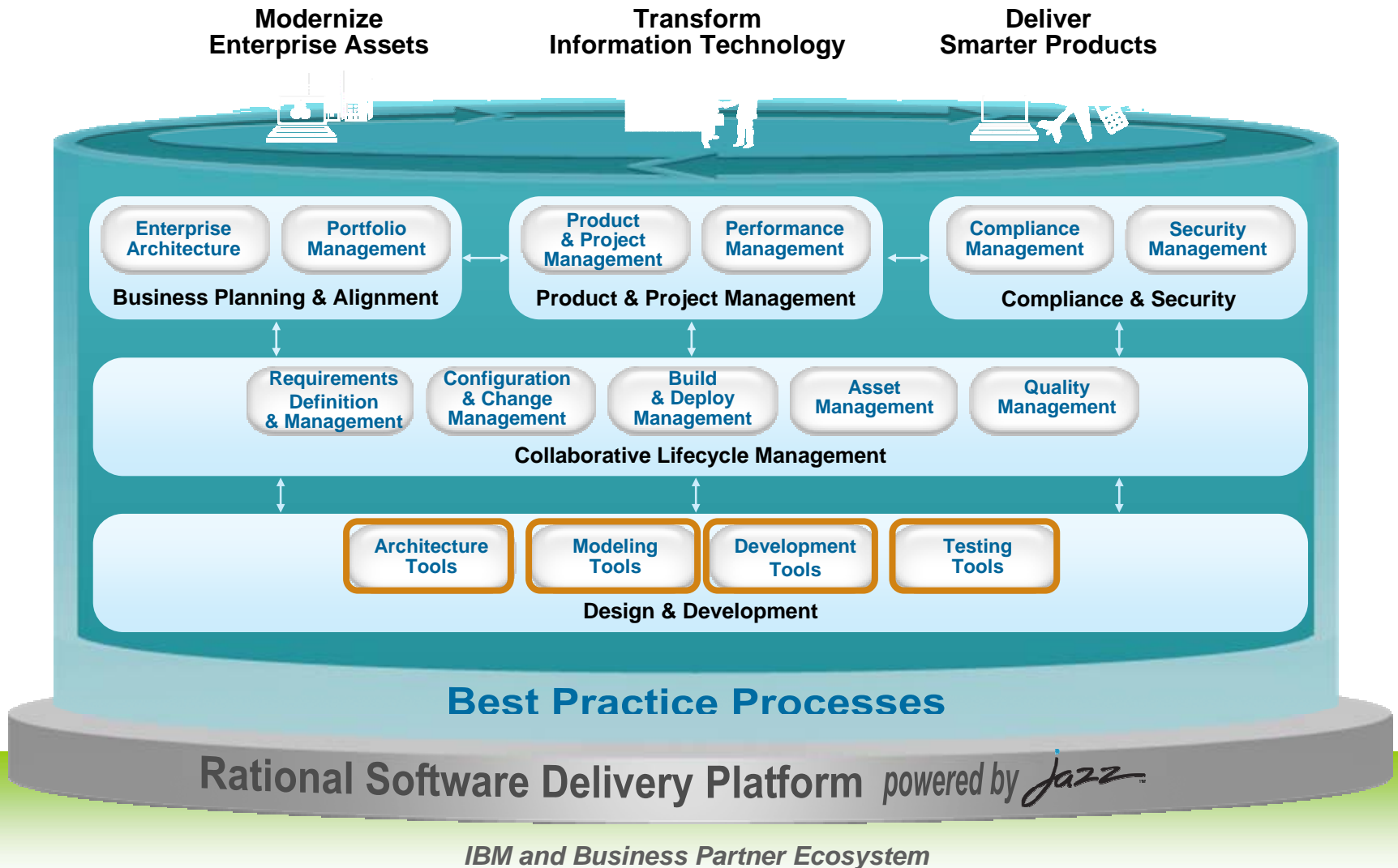


Introducción a IBM Rational Rhapsody

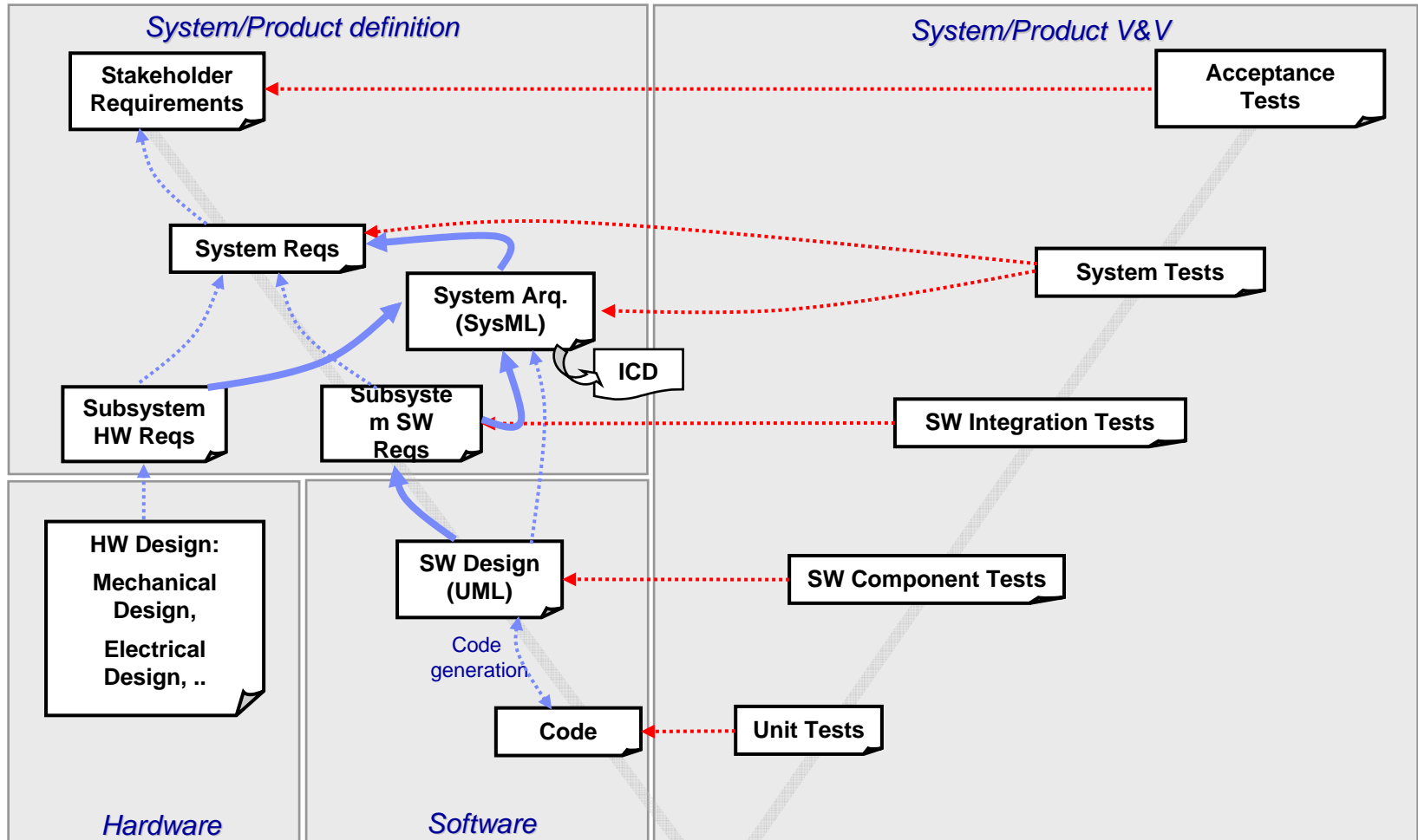
Francisco J. López Minaya
Rational Technical Solution Architect
francisco.lopezminaya@es.ibm.com

Rational. software

Rhapsody dentro del portfolio de IBM Rational



Rhapsody en el desarrollo de sistemas



Problemas comunes en el diseño de software y sistemas

- Se encuentran **errores** de diseño demasiado **tarde**, cuando más cuesta solucionarlos.
- O no se encuentran: el **coste** de los errores no encontrados es **muy alto** (retirada o devolución del sistema, penalizaciones, reputación).
- Mala **comunicación** entre diferentes disciplinas: ingenieros de sistemas, ingenieros mecánicos, eléctricos, de software, cada uno habla su propio lenguaje!!!
- **Colaboración** poco eficiente entre equipos de trabajo distribuidos
- Dificultad para gestionar los **cambios** y la evolución continua de los **requisitos**
- Dificultad en la gestión y la integración de **diseños** cada vez más **complejos**, y con sistemas existentes (**Legacy** systems)
- Las **regulaciones y normativas** nos invaden: DO-178B, FDA, ...
- **Baja productividad en procesos de prueba**. Procesos lentos y poco ágiles



El tiempo de desarrollo se debe emplear en desarrollar...

El 80% de los costes de desarrollo se emplean en identificar y solucionar errores



Source: 2008 GBS Industry standard study
 Defect cost derived in assuming it takes 8 hrs to find, fix and repair a defect when found in code and unit test.
 Defect FFR cost for other phases calculated by using the multiplier on a blended rate of \$80/hr

Los sistemas evolucionan, y cómo se construyen también...

ASM

```

format ELF execut
entry start

start:

mov esi,logos
call display_s

mov [command]
pop eax
lea esp,[esp+
pop eax
pop [environm
call get_param
jc informati

call init_memo

mov edi,chara
mov ecx,100h
xor al,al
make_characters_t
stosb
inc al
loop make_char
mov esi,characters+'a'
    
```

C

```

void CruiseControl_init(_C_CruiseControl *C)
{
    CruiseSpeedMgt_init(&C->CruiseSpeedMgt);
    CruiseStateMgt_init(&C->CruiseStateMgt);
    C->M_conduct_0 = true;
    ThrottleCmd_init(&C->ThrottleCmd);
    C->M_init = true;
}

void CruiseControl(_C_CruiseControl *C)
{
    bool BrakePressed;
    bool AcceleratorPressed;
    bool SpeedOutOffLimits;
    bool LL19;

    /*#code for node CruiseControl */
    /* call to node not expanded DetectPedalsPressed ! */
    C->Cn_DetectPedalsPressed_I0_Brake = C->Cn_DetectPedalsPressed_I1_Accelerator =
    DetectPedalsPressed(&C->Cn_DetectPedalsPressed);
    BrakePressed = C->Cn_DetectPedalsPressed_I00_AcceleratorPressed =
    AcceleratorPressed =
    C->Cn_DetectPedalsPressed_I01_AcceleratorPressed;
    /* call to node not expanded DetectSpeedLimits */
    C->Cn_DetectSpeedLimits_I0_speed = C->I8_Speed;
    DetectSpeedLimits(&C->Cn_DetectSpeedLimits);
    SpeedOutOffLimits = C->Cn_DetectSpeedLimits_I000;
    /* call to node not expanded CruiseStateMgt */
    C->C3_CruiseStateMgt_I0_BrakePressed = BrakeP
    
```

```

void CruiseControl_init(_C_CruiseControl *C)
{
    CruiseSpeedMgt_init(&C->CruiseSpeedMgt);
    CruiseStateMgt_init(&C->CruiseStateMgt);
    C->M_conduct_0 = true;
    ThrottleCmd_init(&C->ThrottleCmd);
    C->M_init = true;
}

void CruiseControl(_C_CruiseControl *C)
{
    bool BrakePressed;
    bool AcceleratorPressed;
    bool SpeedOutOffLimits;
    bool LL19;

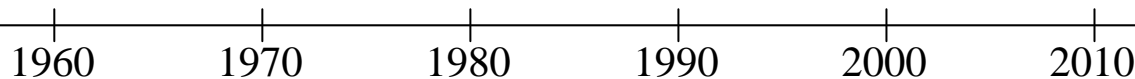
    /*#code for node CruiseControl */
    /* call to node not expanded DetectPedalsPressed ! */
    C->Cn_DetectPedalsPressed_I0_Brake = C->Cn_DetectPedalsPressed_I1_Accelerator =
    DetectPedalsPressed(&C->Cn_DetectPedalsPressed);
    BrakePressed = C->Cn_DetectPedalsPressed_I00_AcceleratorPressed =
    AcceleratorPressed =
    C->Cn_DetectPedalsPressed_I01_AcceleratorPressed;
    /* call to node not expanded DetectSpeedLimits ! */
    C->Cn_DetectSpeedLimits_I0_speed = C->I8_Speed;
    DetectSpeedLimits(&C->Cn_DetectSpeedLimits);
    SpeedOutOffLimits = C->Cn_DetectSpeedLimits_I000;
    /* call to node not expanded CruiseStateMgt */
    C->C3_CruiseStateMgt_I0_BrakePressed = BrakeP
    
```

Model Driven Development MDD

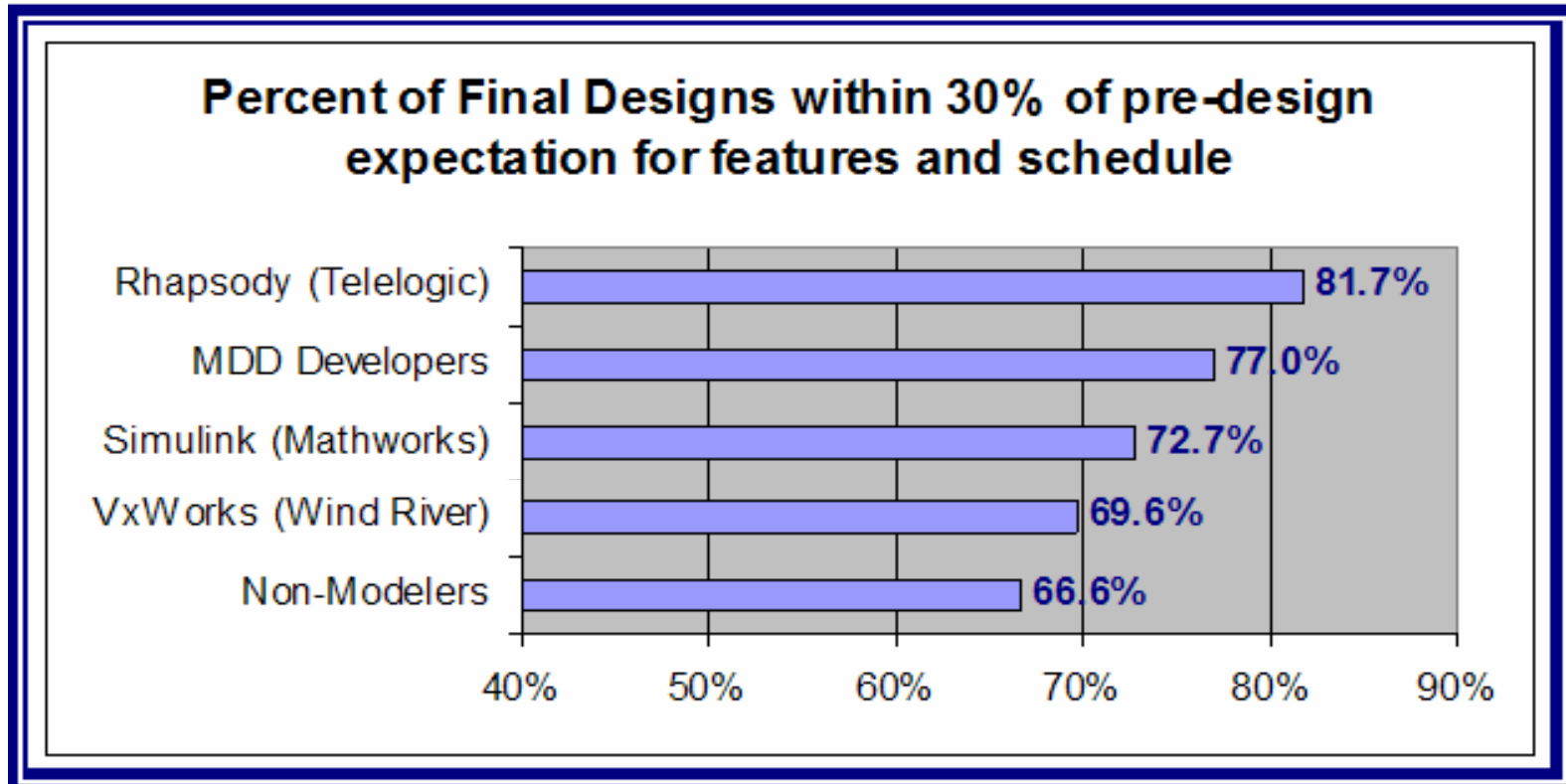
Model Driven Development (MDD):

- Mejora la comunicación
- Modelos más fácilmente trazables que el código
- Modelos permiten automatización:
 - ✓ Simulación del modelo
 - ✓ Generación de Código

t



Model Driven Development es la forma más productiva de desarrollar sistemas: **PROBADO!!!**

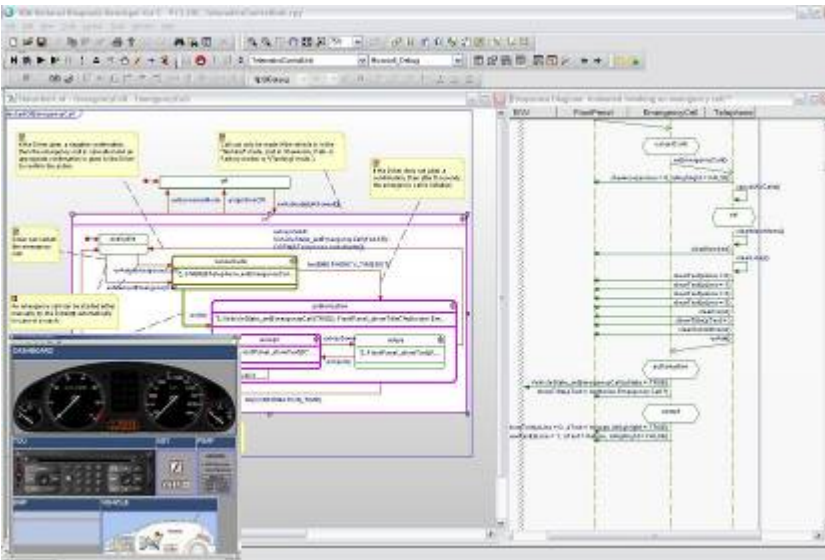


Embedded Market Forecasters

Documented in “What Do You Do When the Horse You’re Riding Drops Dead? Why Model Driven Design is Emerging as a Preferred Best Practice”, March 2007

IBM Rational Rhapsody®

Model Driven Development para software y sistemas complejos

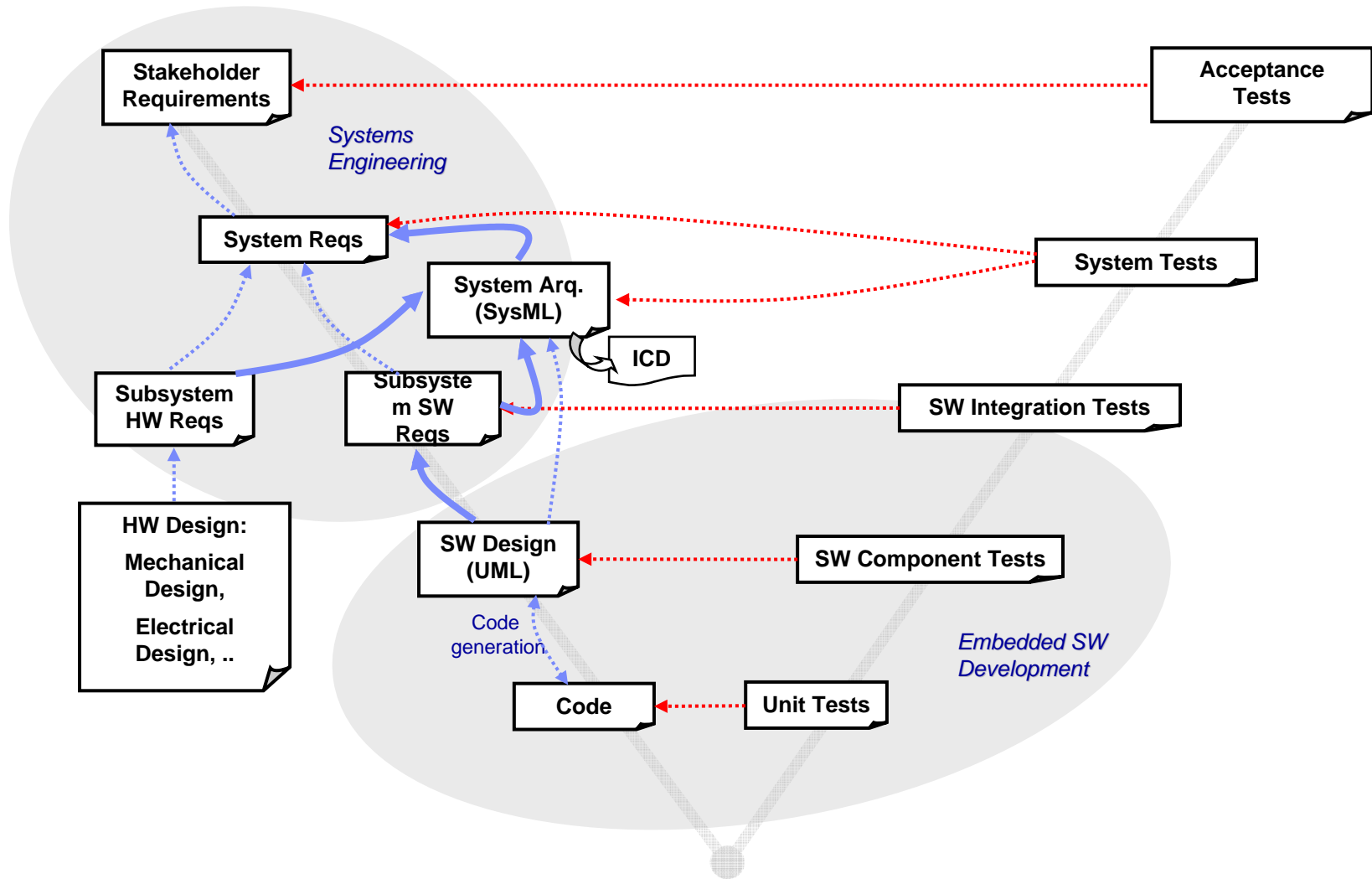


"Working with Rational Rhapsody, Ikerlan-IK4 reports that it is reducing the cost of development by as much as 25 percent and also estimates that they are able to reduce development time by a factor of 10 for each variation in their product line"

Ikerlan-IK4, organization specialized in mechanical and electronic energy technologies

- Detección y **eliminación** temprana de **errores** en diseño mediante la simulación del sistema (verificación temprana)
- **Incrementa la productividad del desarrollador** mediante la generación automática de código y de la documentación
- Mejora la **comunicación** con el cliente y entre disciplinas mediante la utilización de UML/SysML
- Visualización de los **requisitos** y **trazabilidad** a elementos del modelo
- Mejora la **colaboración** entre equipos de desarrollo distribuidos
- **Incrementa la productividad del equipo de pruebas:** Pruebas dirigidas por el modelo (*Model Driven Testing – MDT*)

Rhapsody en el desarrollo de sistemas



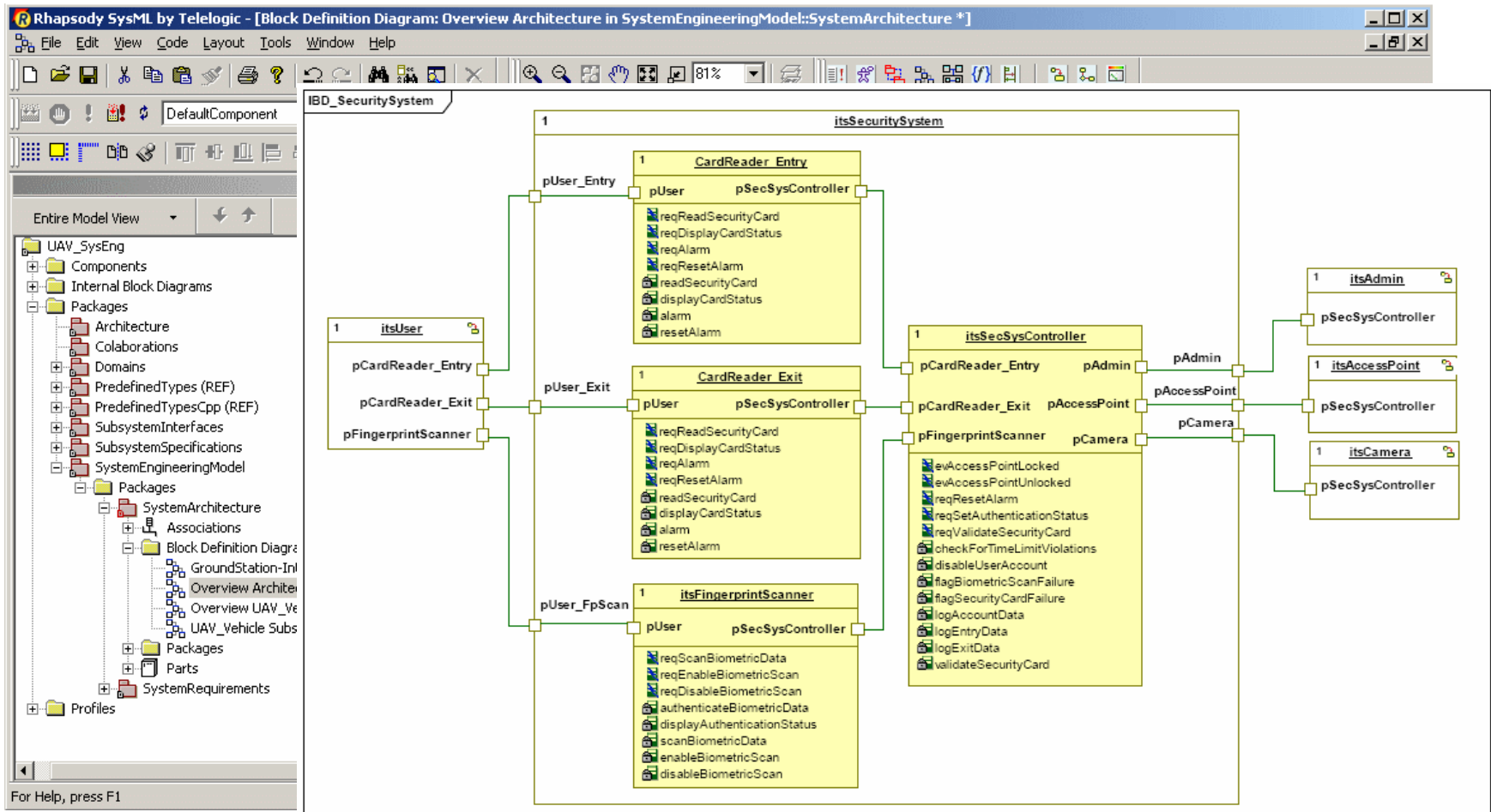
Arquitectura del sistema con Rhapsody



- Systems Modeling Language (SysML):
 - ▶ Extensión de UML 2.0 para dar soporte a las actividades y artefactos de Ingeniería de Sistemas
- SysML permite representar los **requisitos** como parte del **modelo**
 - ▶ **Visualización** de los requisitos en contexto con el modelo
 - ▶ **Trazabilidad** a requisitos: construcción del **sistema correcto**, impacto de **cambios**, mejor **comunicación** con el cliente y con otras disciplinas, cumplimiento de **normativas**
- SysML permite diseñar la **arquitectura** completa del **sistema**, modelando **componentes** de cualquier naturaleza (hardware, software, firmware..) y cómo se comunican (**interfaces**)
 - ▶ Mejora la comunicación entre disciplinas
- **Simulación** de modelos SysML
 - ▶ Verificación temprana: Detección de **errores** en el diseño en la fase de diseño de la arquitectura del sistema

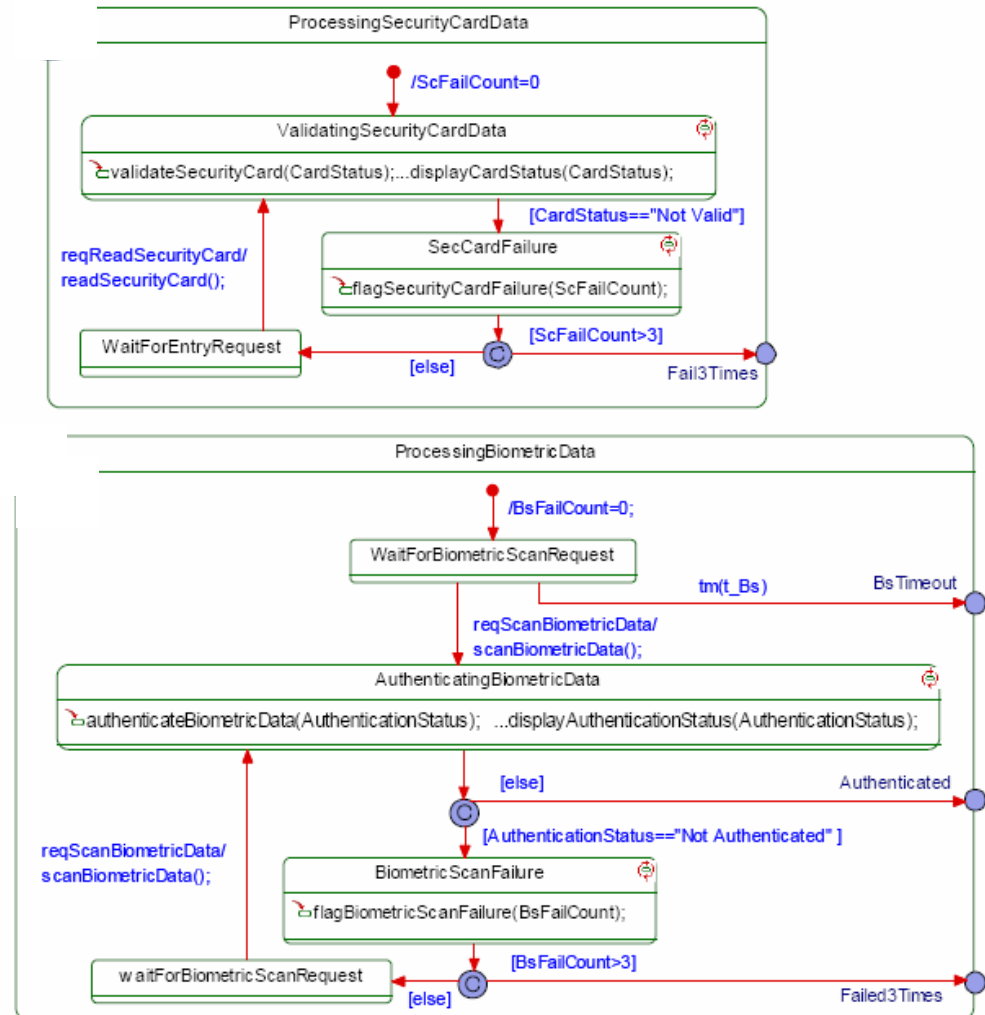
Diseño de la arquitectura del sistema

- Estructura: Block Definition Diagrams e Internal Block Diagrams



Diseño de la arquitectura del sistema

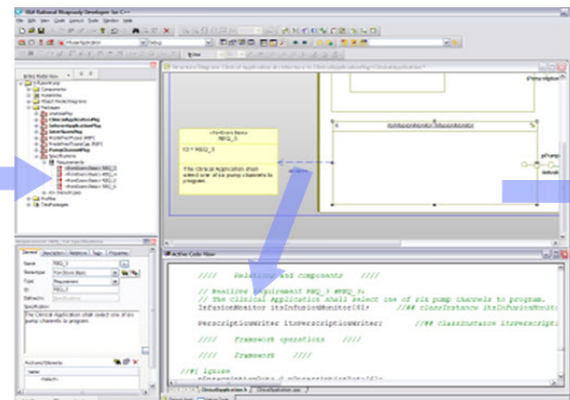
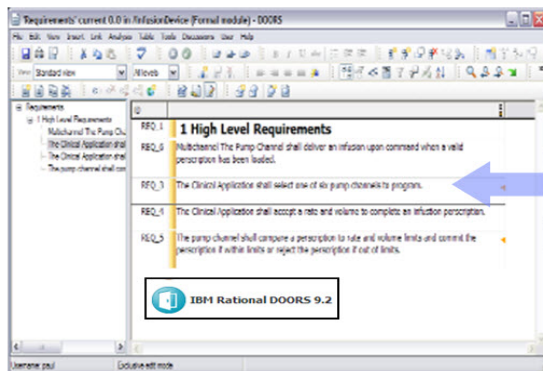
- Comportamiento:
 - ▶ Use Case Diagrams
 - ▶ Activity Diagrams
 - ▶ Sequence Diagrams
 - ▶ Statechart Diagrams





Los requisitos como parte del modelo

- Crear el diseño de la arquitectura del sistema a partir de los requisitos importados
 - ▶ Integración bi-direccional con Rational DOORS y RequisitePro
- Trazabilidad diseño requisitos
 - ▶ Visualización de los requisitos a los que responden las funcionalidades implementadas
 - ▶ Generación automática de informes de trazabilidad: conformidad con normativas
 - ▶ Facilita actividades de análisis:
 - Identificar requisitos no implementados por ningún componente
 - Identificar artefactos de diseño que no responden a ningún requisito



Trazabilidad de los requisitos de sistema al modelo

Asegurar que la arquitectura cumple los requisitos del sistema...

'System Requirements Specification' current 0.0 in /2. System Requirements Specifications (Formal modul...

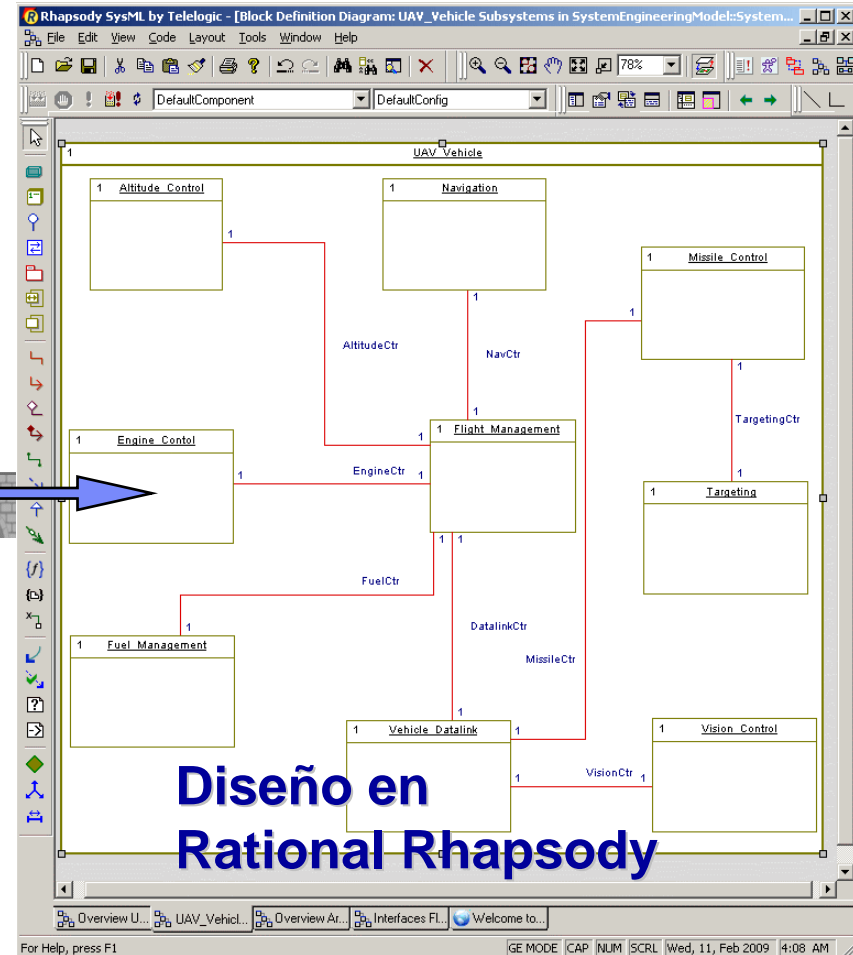
File Edit View Insert Link Analysis Table Tools Discussions User Publish WEXP Rhapsody 7.4 RG Analyst
Change Management Harmony/ESW Harmony/ITSW Harmony/SE TAU Help

View B1 - Edit View All levels

ID	Object Type	System Requirements Specification
1 Overview		
SRSUAV-1	*	
SRSUAV-2	*	The Unmanned Air Vehicle System is a system solution to a medium-range reconnaissance in hostile environments with limited attack capability.
SRSUAV-3	*	It is a medium-range long endurance UAV system that can carry a variety of payloads to assist in ground, air and sea operations.
SRSUAV-4	*	A full UAVS consist of four UAVs and a ground Mission Planning and Control System.
SRSUAV-5	*	
2 System Requirements		
2.1 UAV Vehicle		
SRSUAV-6	Requirement	The CUAV is shall be a multipurpose and reusable UAV with multimission capability.
SRSUAV-78	Requirement	This is a new system requirement..
SRSUAV-7	Requirement	It shall operate at altitudes of up to 30,000 feet.
SRSUAV-8	Requirement	It shall reach ground speeds of up to 100 knots in cruise mode.
SRSUAV-9	Requirement	It shall reach ground speeds of 150 knots in dash mode.
SRSUAV-10	Requirement	It shall carry payloads up to 450 lbs for durations exceeding 24 hours.
SRSUAV-11	Requirement	The UAV shall fly unimpeded in low visibility environments while carrying reconnaissance or attack payloads.
SRSUAV-12	Requirement	The UAV shall be controllable from the ground station CMPCS.
SRSUAV-13	Requirement	The UAV shall be capable of flying complex flight plans with the operational goal of systematic area search.
SRSUAV-14	Requirement	The UAV shall be capable of flying complex flight plans with the operational goal of ground route or road based (synonym) search.
SRSUAV-15	Requirement	The UAV shall be capable of flying complex flight plans with the operational goal of orbit surveillance of point targets.
SRSUAV-16	Requirement	The UAV and manned control capability from the ground station shall provide sustained 24 hour flight with real-time visual, infra-red or radar telemetry with target recognition preprocessing.
SRSUAV-17	Requirement	The Communications shall be jam resistant in environments that are not high ECM.
SRSUAV-18	Requirement	Control commands shall be encrypted.
SRSUAV-19	Requirement	Telemetry shall be encrypted and un-protected.
SRSUAV-20	Requirement	Telemetry rates for visual telemetry shall support 30 frames-per-second at 640 x 400 resolution.
SRSUAV-21	Requirement	Visual flight shall be supported within line of sight (LOS) range of ground stations. Because the Coyote is capable of being passed among different CMPCSs.
SRSUAV-22	Requirement	For navigation the Coyote has on-board global positioning system based navigation and shall be directly controllable from the ground station.

Username: Administrator Exclusive edit mode

¿Cómo?
Rhapsody Gateway



Requisitos en Rational DOORS



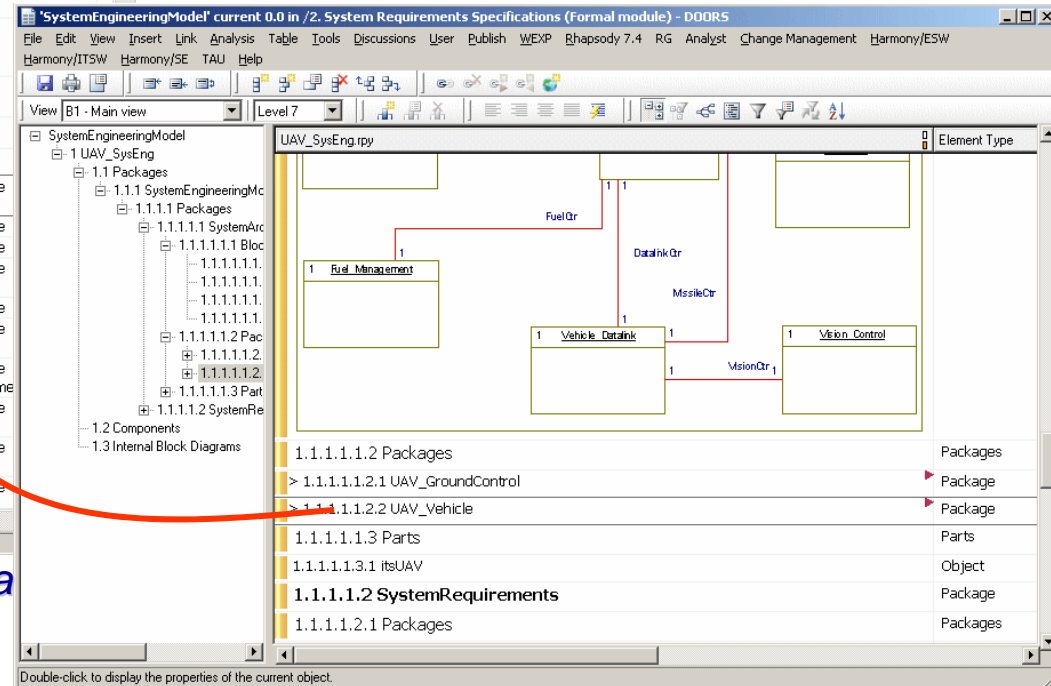
Trazabilidad de los requisitos de sistema al modelo

Asegurar que la arquitectura cumple los requisitos del sistema...

- Las dependencias pueden establecerse desde DOORS

ID	Object Type	System Requirements Specification	Allocated to (System Architecture)
SRSUAV-3	*	It is a medium-range long endurance UAV system that can carry a variety of payloads to assist in ground, air and sea operations.	
SRSUAV-4	*	A full UAVS consist of four UAVs and a ground Mission Planning and Control System.	
SRSUAV-5	*	2 System Requirements	
SRSUAV-74	*	2.1 UAV Vehicle	
SRSUAV-6	Requirement	The CUAV is shall be a multipurpose and reusable UAV with multimission capability.	Package: UAV_Vehicle
SRSUAV-78	Requirement	This is a new system requirement..	Package: UAV_Vehicle
SRSUAV-7	Requirement	It shall operate at altitudes of up to 30,000 feet.	Package: UAV_Vehicle
SRSUAV-8	Requirement	It shall reach ground speeds of up to 100 knots in cruise mode.	Package: UAV_Vehicle
SRSUAV-9	Requirement	It shall reach ground speeds of 150 knots in dash mode.	Package: UAV_Vehicle
SRSUAV-10	Requirement	It shall carry payloads up to 450 lbs for durations exceeding 24 hours.	Package: UAV_Vehicle
SRSUAV-11	Requirement	The UAV shall fly unimpeded in low visibility environments while carrying reconnaissance or attack payloads.	Package: UAV_Vehicle block: Flight_Management
SRSUAV-12	Requirement	The UAV shall be controllable from the ground station CMPCS.	Package: UAV_Vehicle
SRSUAV-13	Requirement	The UAV shall be capable of of flying complex flight plans with the operational goal of systematic area search.	Package: UAV_Vehicle
SRSUAV-14	Requirement	The UAV shall be capable of of flying complex flight plans with the operational goal of ground route or road based. (synonym)	Package: UAV_Vehicle

Módulo DOORS con una copia de la información del modelo



Crear relación desde el modelo a los requisitos

Gateway mantiene actualizada la información del modelo, incluyendo relaciones con reqs



Trazabilidad de los requisitos de sistema al modelo

Asegurar que la arquitectura cumple los requisitos del sistema...

- ...o desde Rhapsody

Rhapsody SysML by Telelogic - UAV_SysEng.rpy - [Block Definition Diagram: UAV_Vehicle Subs

File Edit View Code Layout Tools Window Help

DefaultComponent DefaultConfig

Entire Model View

- SubsystemInterfaces
- SubsystemSpecifications
- SystemEngineeringModel
 - Packages
 - SystemArchitecture
 - SystemRequirements
 - OverviewUseCase
 - System_Specifications
 - Packages
 - «fromDoors Advanced EADS» _2_System_Requirements
 - Packages
 - «fromDoors Advanced EADS» _2_UAV_Vehicle
 - Requirements
 - «fromDoors Advanced EADS» SRSUAV-10
 - «fromDoors Advanced EADS» SRSUAV-11
 - «fromDoors Advanced EADS» SRSUAV-12
 - «fromDoors Advanced EADS» SRSUAV-13
 - «fromDoors Advanced EADS» SRSUAV-14
 - «fromDoors Advanced EADS» SRSUAV-15
 - «fromDoors Advanced EADS» SRSUAV-16
 - «fromDoors Advanced EADS» SRSUAV-17
 - «fromDoors Advanced EADS» SRSUAV-18
 - «fromDoors Advanced EADS» SRSUAV-19
 - «fromDoors Advanced EADS» SRSUAV-20

Requirement: SRSUAV-16 in _2_1 UAV_Vehicle

General Description Relations Tags Properties

The UAV and manned control capability from the ground station shall provide sustained 24 hour flight with real-time visual, infra-red or radar telemetry with target recognition preprocessing.

Locate OK Apply

For Help, press F1

Requisitos en Rhapsody.

Gateway sincroniza la información desde DOORS

Rhapsody SysML by Telelogic - UAV_SysEng.rpy - [Block Definition Diagram: UAV_Vehicle Subs

File Edit View Code Layout Tools Window Help

DefaultComponent DefaultConfig

Entire Model View

- SubsystemSpecifications
- SystemEngineeringModel
 - Packages
 - SystemArchitecture
 - Associations
 - Block Definition Diagrams
 - GroundStation-Interface Relationship
 - Overview Architecture
 - Overview UAV_Vehicle
 - UAV_Vehicle Subsystems
 - Packages
 - UAV_GroundControl
 - UAV_Vehicle
 - blocks
 - Dependencies
 - «fromDoors Advanced EADS» SRSUAV_10
 - «fromDoors Advanced EADS» SRSUAV_11
 - «fromDoors Advanced EADS» SRSUAV_12
 - «fromDoors Advanced EADS» SRSUAV_13
 - «fromDoors Advanced EADS» SRSUAV_14
 - «fromDoors Advanced EADS» SRSUAV_15
 - «fromDoors Advanced EADS» SRSUAV_16
 - «fromDoors Advanced EADS» SRSUAV_17
 - «fromDoors Advanced EADS» SRSUAV_18
 - «fromDoors Advanced EADS» SRSUAV_19
 - «fromDoors Advanced EADS» SRSUAV_20
 - «fromDoors Advanced EADS» SRSUAV_21
 - «fromDoors Advanced EADS» SRSUAV_22
 - «fromDoors Advanced EADS» SRSUAV_23
 - «fromDoors Advanced EADS» SRSUAV_24
 - «fromDoors Advanced EADS» SRSUAV_25
 - «fromDoors Advanced EADS» SRSUAV_6
 - «fromDoors Advanced EADS» SRSUAV_7
 - «fromDoors Advanced EADS» SRSUAV_78
 - «fromDoors Advanced EADS» SRSUAV_8
 - «fromDoors Advanced EADS» SRSUAV_9

Parts

SystemRequirements

Profiles

For Help, press F1

Requisitos relacionados con el paquete "UAV_Vehicle".

Trazabilidad de los requisitos de sistema al modelo

Asegurar que la arquitectura cumple los requisitos del sistema...

- Impacto y cobertura desde Gateway

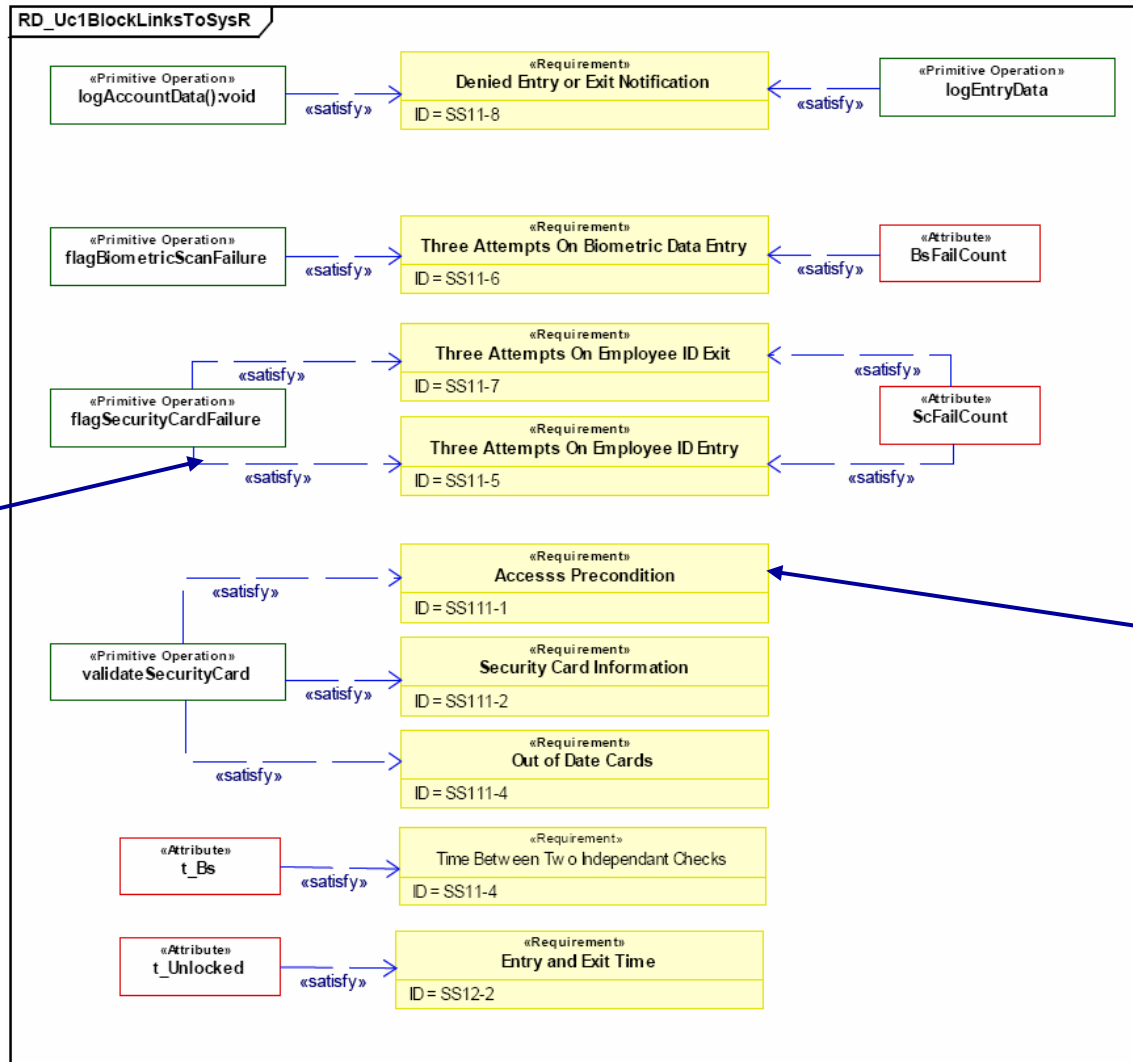
The screenshot displays the Telelogic Rhapsody Gateway - UAV_SysEng application interface. The window title is "Telelogic Rhapsody Gateway - UAV_SysEng". The menu bar includes File, Edit, View, Tools, Reports, and Help. The toolbar contains various icons for file operations and analysis. The main interface is divided into three panes:

- Upstream Impact Information:** A tree view showing requirements (SRSUAV-6 to SRSUAV-15) and their associated UAV models (UAV-6 to UAV-15).
- Selection:** A tree view showing the current selection path: Rule check > UML Model Rhapsody SysML > UAV_SysEng > System Specifications Doors Advanced EADS > 1 Overview > 2 System Requirements > 2.1 UAV Vehicle. The selected item is "2.1 UAV Vehicle".
- Downstream Impact Information:** A tree view showing the downstream impact of the selection, including UAV_Vehicle models (UAV_Vehicle 6-15) and Flight_Management.

At the bottom of the interface, there are three tabs: "Texts and Reference Attributes", "Attributes", and "Messages". The "Texts and Reference Attributes" tab is active, showing fields for "Upstream", "Selection", and "Downstream" with "Text:" and "Reference Attributes:" labels. The status bar at the bottom indicates the current selection: "System Specifications Doors Advanced EADS/2 System Requirements/2.1 UAV Vehicle".

Visualización de los requisitos y su trazabilidad

Diagramas de Requisitos



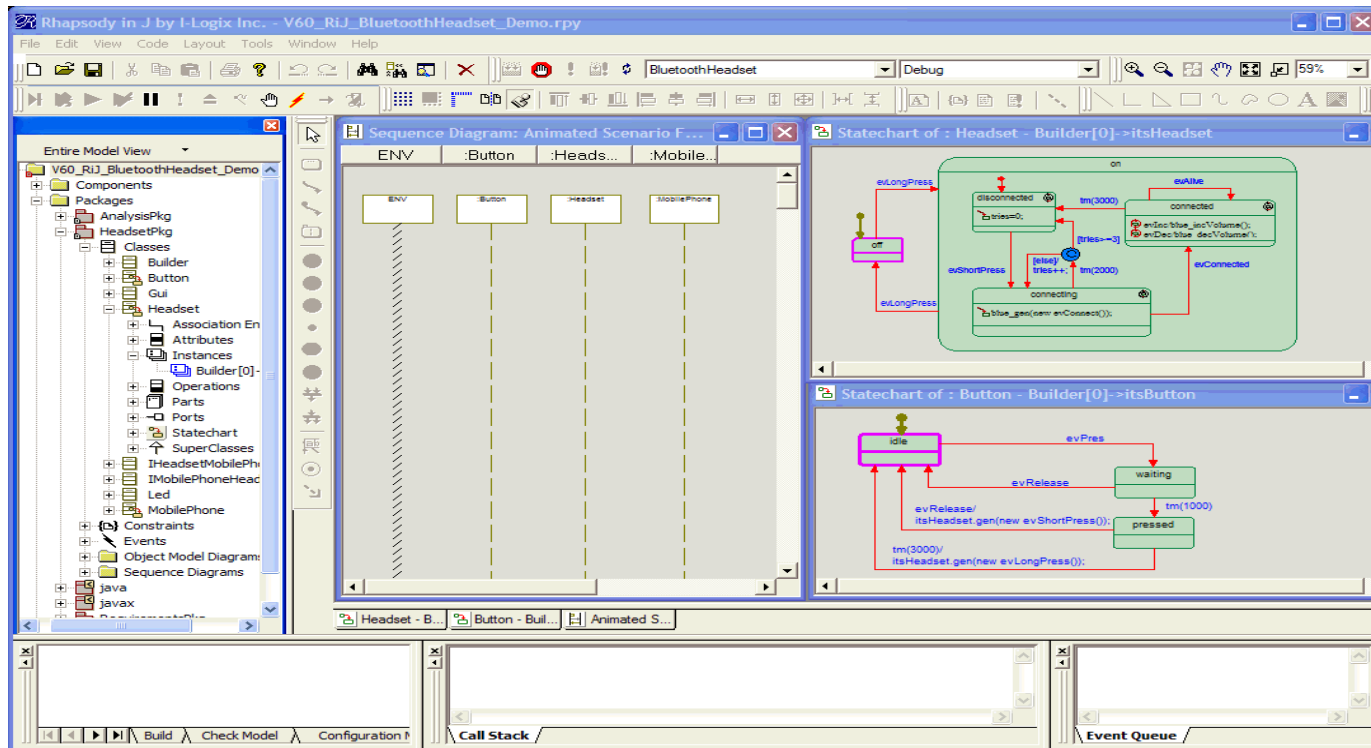
Trazabilidad, relación de satisfacción

Requisito



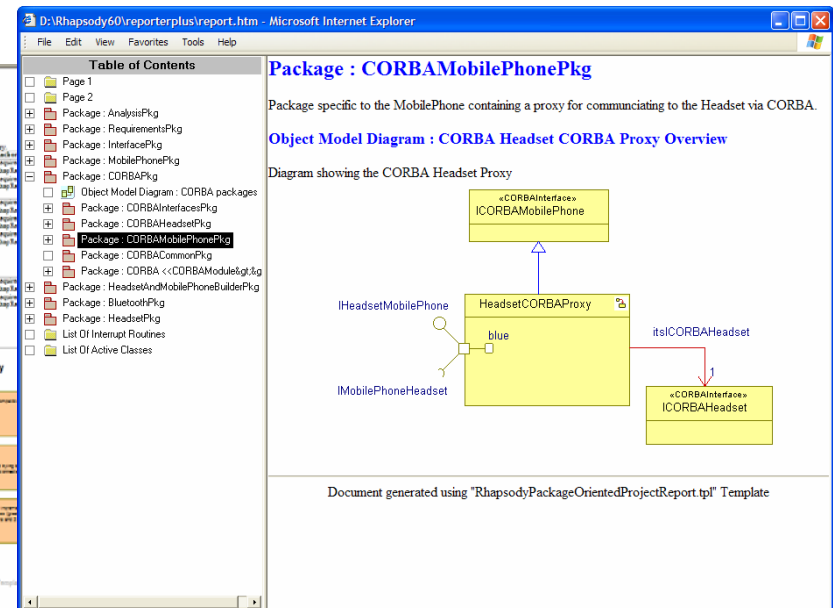
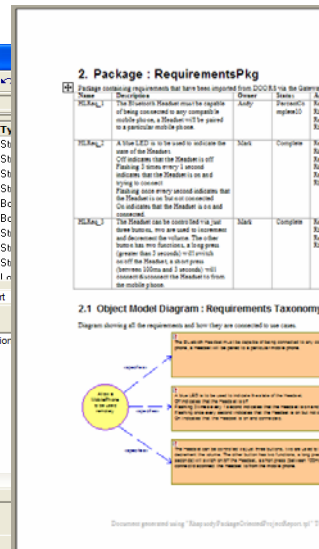
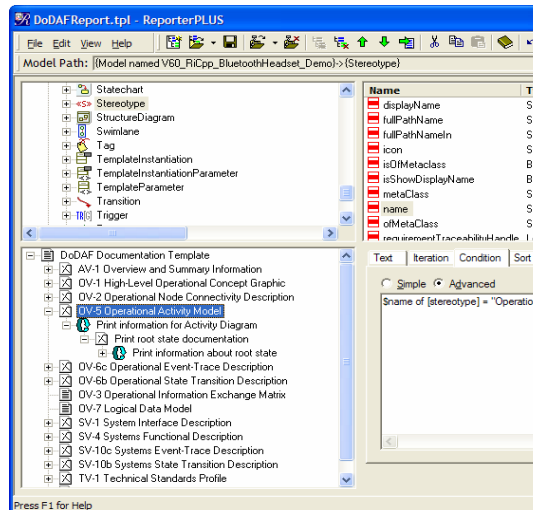
Verificación y validación temprana

- La ejecución del modelo ayuda a **detectar errores** en las fases tempranas, permite analizar si el comportamiento del sistema es correcto
- Visualización de la ejecución**: permite mostrar y comunicar visualmente al cliente el comportamiento del sistema



Generación automática de documentación desde el modelo

- Informes personalizables
- Diferentes formatos: HTML, Microsoft® Word®, PowerPoint®, Rich Text Format
 - ▶ Crea enlaces para una navegación rápida del informe
- Amplia variedad de plantillas predefinidas
- Fácil de utilizar



Generación de documentación: diagramas, descripción textual..

2. Subsystem Interface Description

2.1 Interfaces of first Level of Subsystem division

2.1.1 Interface IDatalink

General Description

Datalink is an High level description of the communication protocol established to communicate the GroundStation and the UAV_Vehicle.

This GroundStation can control 2-5 UAV_Vehicles.

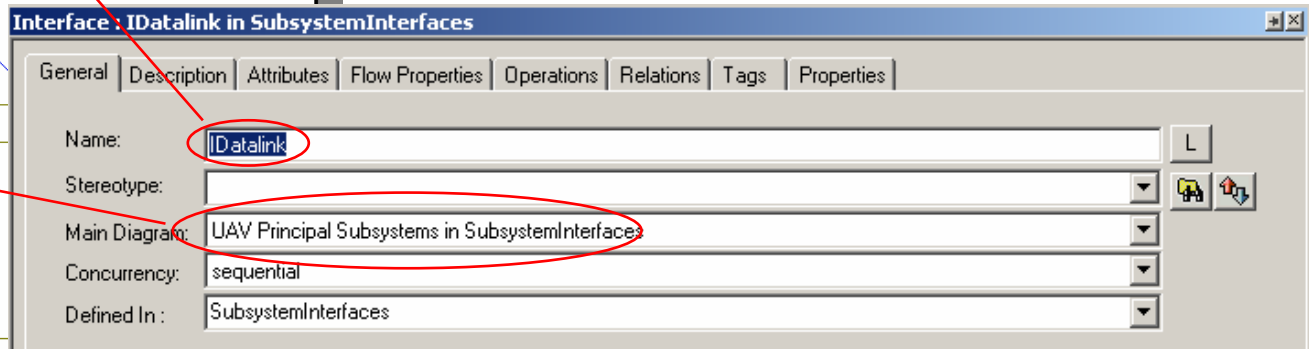
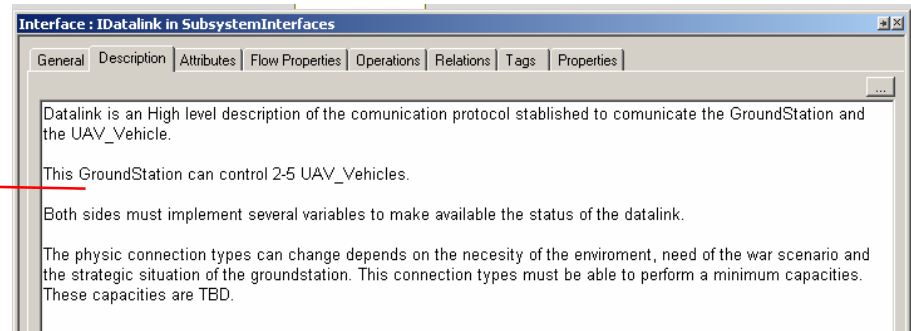
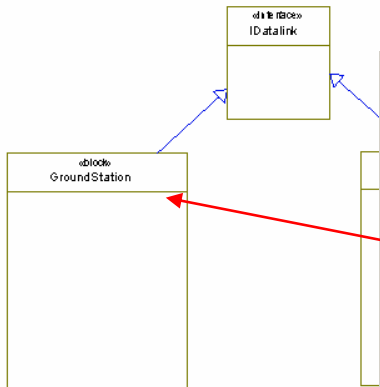
Both sides must implement several variables to make available the status of the datalink.

The physic connection types can change depends on the necessity of the enviroment, need of the war scenario and the strategic situation of the groundstation. This connection types must be able to perform a minimum capacities. These capacities are TBD.

Interface uses diagrams

This Diagram is created to describe the interfaces generated to the communication between the 2 first level Subsystem.

The Datalink Interface reflect the message that could be send/received among the 2 subsystems.



Generación de documentación: mensajes en tablas, tipos de datos...

2.1.1.1 Attributes in the interface IDatalink

| Name | Type | Range |
|----------------|----------------|--------------------|
| LinkType | LinkType | See LinkType |
| LinkStatus | LinkStatusType | See LinkStatusType |
| LinkSpeed | int | 9800..54000000 |
| CongestionFlag | int | 0..100 |

2.1.1.2 Messages in the Interface IDatalink

In this section will be described the messages that each subsystem must implement to communicate with the rest of the Subsystems that expect to use this interface to communicate.

2.1.1.2.1 Message: InitComms

Description:
Initialize the communications with the requested address.

Fields in this message

| Name | Type | Range | Length |
|-------------|----------------|--------------------|---------|
| OrgAddress | int | 0..2632 | 32 bits |
| OrgPort | int | 0..2632 | 32 bits |
| DestAddress | int | 0..2632 | 32 bits |
| DestPort | int | 0..2632 | 32 bits |
| Type | LinkType | See LinkType | 4 bits |
| DLOperation | DataLinkOpType | See DataLinkOpType | 3 bits |

2.1.1.2.2 Message: CloseComms

Description:
Finalize the communications with the requested address.

Fields in this message

| Name | Type | Range | Length |
|-------------|----------------|--------------------|---------|
| OrgAddress | int | 0..2632 | 32 bits |
| OrgPort | int | 0..2632 | 32 bits |
| DestAddress | int | 0..2632 | 32 bits |
| DestPort | int | 0..2632 | 32 bits |
| Type | LinkType | See LinkType | 4 bits |
| DLOperation | DataLinkOpType | See DataLinkOpType | 3 bits |

2.1.1.2.3 Message: ReceiveHSMsg

Description:
A Message of High Speed and High priority is sent from an origin address-port pair to a destination address-port pair. These pairs represent the 2 subsystems that is connected.

Fields in this message

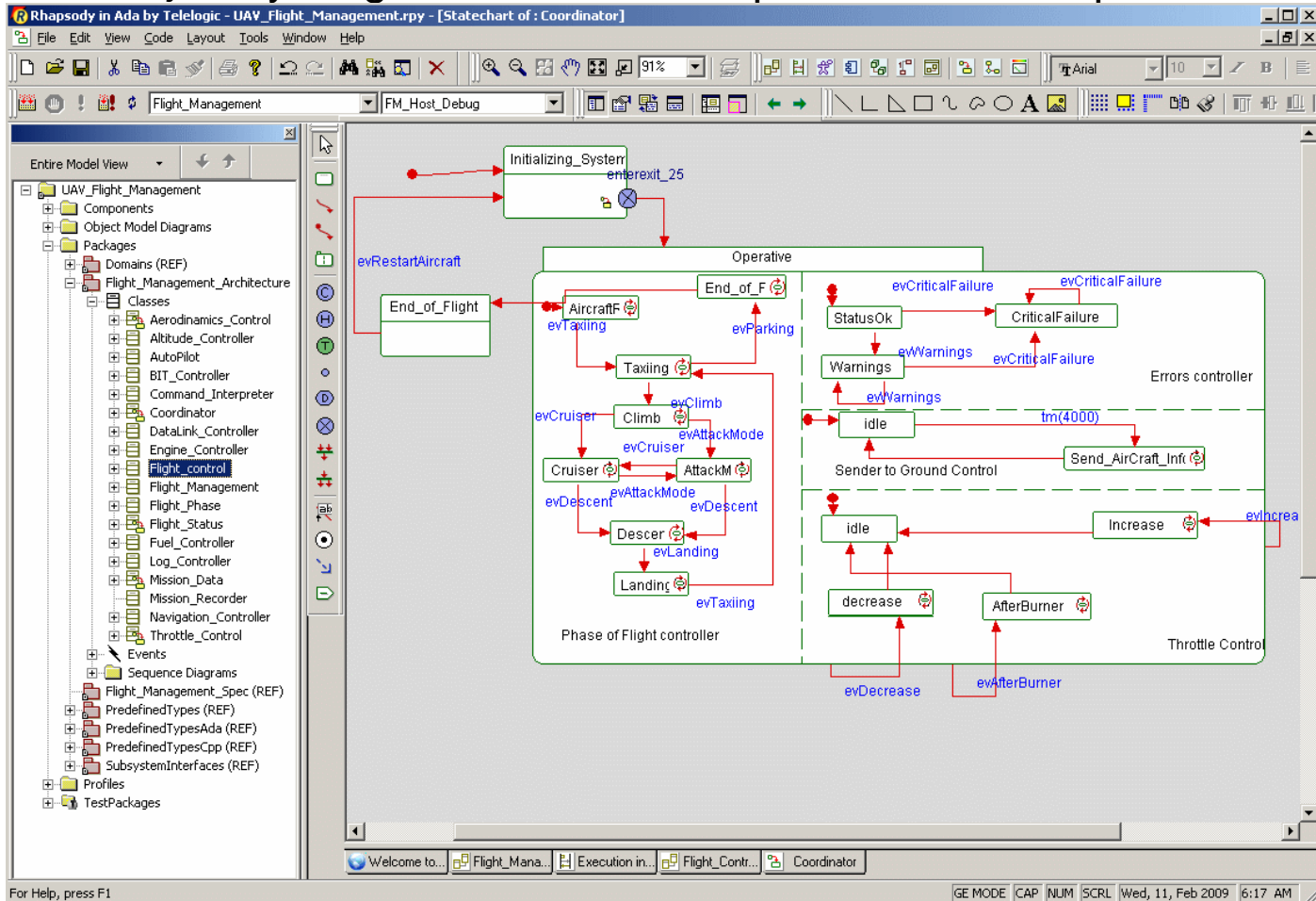
| Name | Type | Range | Length |
|-------------|------|---------|---------|
| OrgAddress | int | 0..2632 | 32 bits |
| OrgPort | int | 0..2632 | 32 bits |
| DestAddress | int | 0..2632 | 32 bits |
| DestPort | int | 0..2632 | 32 bits |





Diseño del software

- Diagrama de objetos y diagrama de estados que define el comportamiento



Si no quiere, no es necesario codificar...

- Añadir detalle al modelo a través de formularios

The screenshot displays the IBM Rational Software Conference 2009 interface. The main window is titled "Rhapsody in Ada by Telelogic - [Throttle_Control.ads]". The interface is divided into several panes:

- Entire Model View (Left Pane):** A tree view showing the project structure. The "Throttle_Control" package is expanded, and the "Attributes" folder is selected. The "currentPower" attribute is highlighted with a red arrow pointing to the code editor.
- Code Editor (Right Pane):** Displays the Ada source code for "Throttle_Control.ads". The code defines a reactive part type and a tagged record type. The line `currentPower : Float := 0.0; ---+ attribute currentPower` is circled in red, with a red arrow pointing to it from the model view.
- Bottom Pane:** Shows the status bar with the date "Wed, 11, Feb 2009" and time "6:29 AM".

```
type reactive_part_t is new Oxf.Reactive.Reactive_t with
record
  its_Parent : Throttle_Control_wide_acc_t;
end record;

end reactive_part;

type Throttle_Control_t is tagged
record
  its_Reactive : aliased reactive_part.reactive_part_t;
  its_Reactive_acc : reactive_part.reactive_part_acc_t;
  Operative_Sub_State : Integer := Non_State;
  root_state_Sub_State : Integer := Non_State;

  -- Fields
  currentPower : Float := 0.0; ---+ attribute currentPower
end record;

procedure Start_Behavior (this : access Throttle_Control_t;
success : out boolean
);
procedure Terminate_Behavior (this : access Throttle_Control_t);
```



Paneles gráficos para la simulación y animación

- Creación de **maquetas de la interfaz** para comunicar eficazmente el comportamiento de diseño destinado a los clientes
- Modificar, monitorizar y analizar los valores de datos durante la simulación para ayudar a garantizar que el **diseño es correcto** en una fase temprana del proceso

The screenshot displays the IBM Rational software interface, illustrating the integration of graphical panels for simulation and animation. The main window, titled "Rhapsody In C++ by Telelogic - [Panel: Dashboard in Simulate]", shows a simulated dashboard for a red sports car. The dashboard includes a speedometer, a digital display showing "1234" and "5678", a "GABC" indicator, a "45%" gauge, and a license plate "LMH 962175".

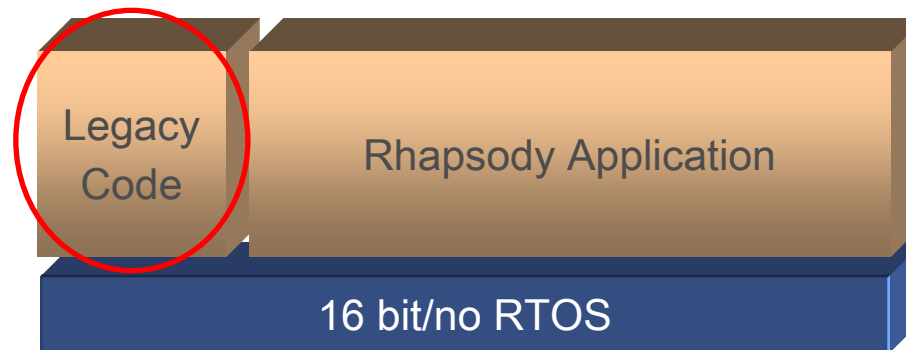
Overlaid on the dashboard are several statechart diagrams:

- Statechart of - Headset - Headset:** This diagram shows states like "disconnected", "connecting", "connected", and "off". Transitions are triggered by events such as "evLongPress", "evShortPress", and "evActive".
- Statechart of - Button - Button:** This diagram shows states like "idle", "debounce", and "pressed". Transitions are triggered by "evPress" and "evRelease".
- Sequence Diagram: Animated Behaviour:** This diagram shows the interaction between "ENV", "Button", "Headset", and "MobileP". It includes messages like "Create()", "Create()", and "Create()", and states like "idle", "off", and "off".

A small window titled "Bluetooth Headset..." shows a 3D model of a blue headset. The bottom of the interface includes a "Call Stack" and "Event Queue" panel.

Generación automática de código desde el modelo

- Generar código completo de aplicaciones C, C++, Java y Ada
- Desplegar rápidamente su diseño en cualquier plataforma de destino.
- Reutilizar código externo:
 - ▶ Referenciado: visualización de código externo
 - ▶ Incluido automáticamente en el modelo mediante ingeniería inversa



Reutilización de modelos ya existentes

- Importación y reutilización de modelos existentes en otras herramientas Rational o XMI-compliant

The image illustrates the process of reusing existing models from Rational Rose in Rhapsody Designer. It shows two screenshots of the software interfaces connected by a large arrow, indicating the transition of the model.

Left Screenshot (Rational Rose): Displays the 'Use-Case Model of the Order System'. The diagram shows actors like 'Order Administrator' and 'Store Administrator' connected to use cases such as 'Manage Order', 'Manage Customer Register', and 'Execute Order'. A 'Class Specification for Order Administrator' dialog is open, showing details for the actor. A console window at the bottom shows system messages.

Right Screenshot (Rhapsody Designer): Shows the same 'Use-Case Model of the Order System' imported into Rhapsody Designer. The diagram is now rendered in Rhapsody's style, with actors and use cases clearly visible. A 'Rose Import' dialog box is open, showing the 'Rose Project' tree with 'Use case View' selected. The 'Import Options' dialog shows 'Import statecharts' and 'Import object model diagrams' checked.

Dialog Boxes:

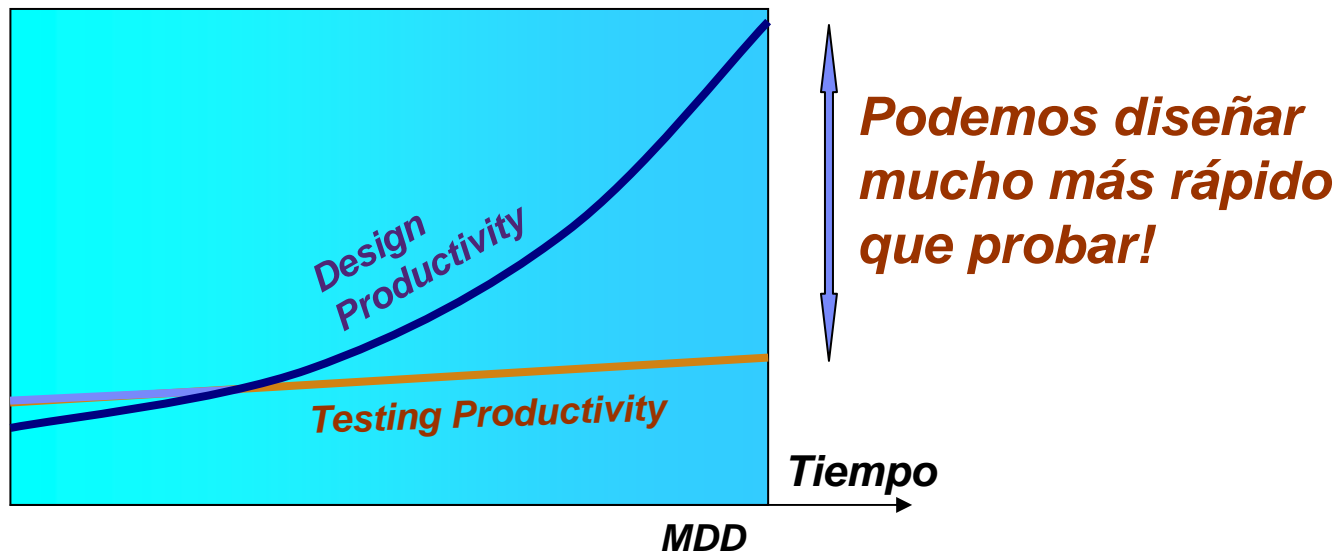
- Class Specification for Order Administrator:** Shows components like 'Name', 'Type', 'Stereotype', and 'Export Control'. The actor is described as 'A person who is responsible for customer register and for receiving and registering orders'.
- Rose Import:** Shows the file to import as 'C:\Program Files\Rational\Rose\samples\ordersystem\ordersys.mdl'. The 'Rose Project' tree is expanded to show 'Use case View'.
- Import Options:** Shows 'Import statecharts' and 'Import object model diagrams' checked.

Annotations: Several yellow callout boxes provide additional information:

- 'The use case model specifies the behavior and surroundings of the system in terms of use cases and actors. The description of an actor or use case is found in its specification, which is displayed when double-clicking on the actor or use case. The documentation field on the General tab, or displayed in the application window when you select and actor or use case. (If not, click Documentation on the View menu.)'
- 'Instead of describing the flow of events of a use case in the documentation field of its specification, an external document with the description can be connected to the use case. That can be done on the Files tab in the Use-Case Specification or by dragging the file from the Windows Explorer and dropping it on the use case in the Browser.'
- 'The use cases constitute the functional requirements on the system. They also define how the requirements are distributed among the design objects of the system, i.e. how the objects interact to perform each use case. In the Browser you can see that the Manage Order use case is described in two interaction diagrams of the type sequence diagram.'

Proceso de pruebas poco productivo...

- Está comprobado que la adopción de MDD mejora la productividad de los desarrolladores
- El equipo de pruebas sigue siendo mucho menos productivo que el de desarrollo: pruebas a nivel de código!!!



- ¿Por qué no trasladar la idea de MDD a Testing?

➔ Model Driven Testing

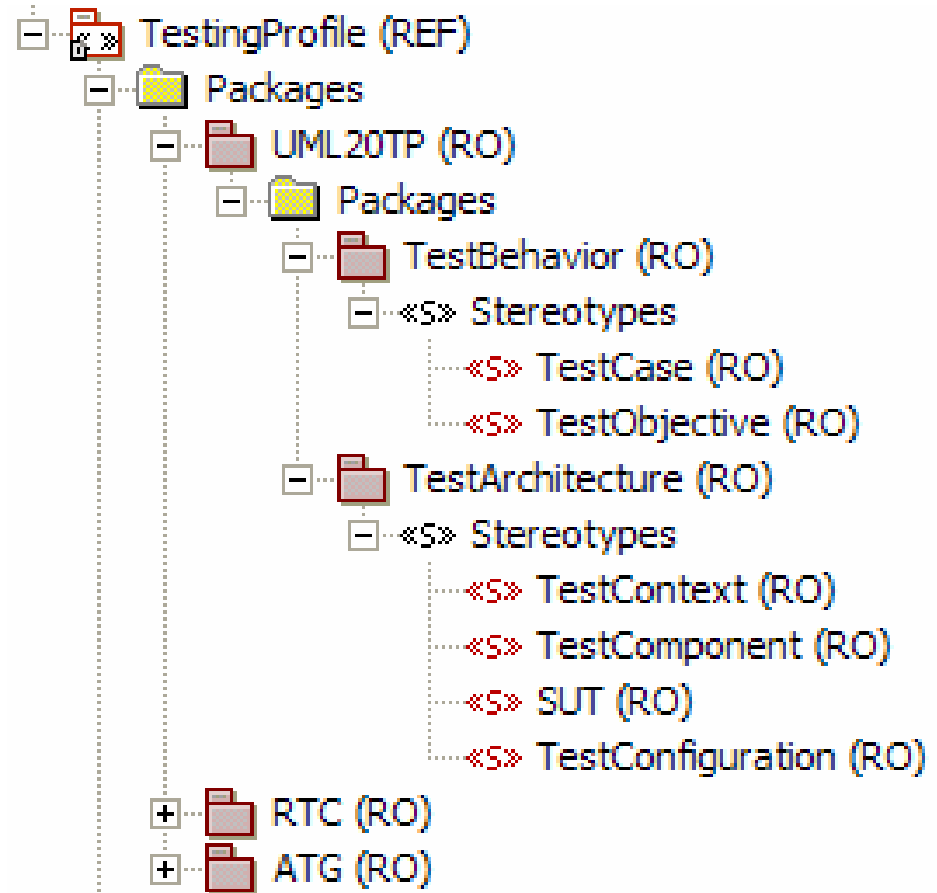
Pruebas dirigidas por modelos (Model Driven Testing - MDT)

- Wikipedia, la enciclopedia libre en la World Wide Web (www), se refiere a *model driven testing* como “*software testing where test cases are derived in whole or in part from a model that describes some (if not all) aspects of the system under test (SUT)*”
- UML es el lenguaje para modelado, mientras que los tests derivados del modelo se representan utilizando **UML Testing Profile (UTP)**
- UTP se puede considerar como un lenguaje para **visualizar, especificar, analizar, construir** y **documentar** todos los artefactos involucrados en un proceso de pruebas.



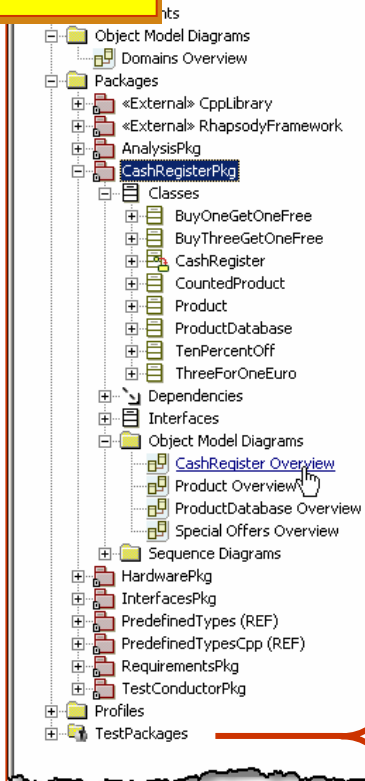
UML Testing Profile

- El UML 2 Testing Profile extiende UML con conceptos específicos de pruebas:
 - ▶ **Test Behavior** : actividades a realizar durante la prueba y resultados
 - ▶ **Test Architecture** : elementos del modelo a probar y sus relaciones con otro elementos

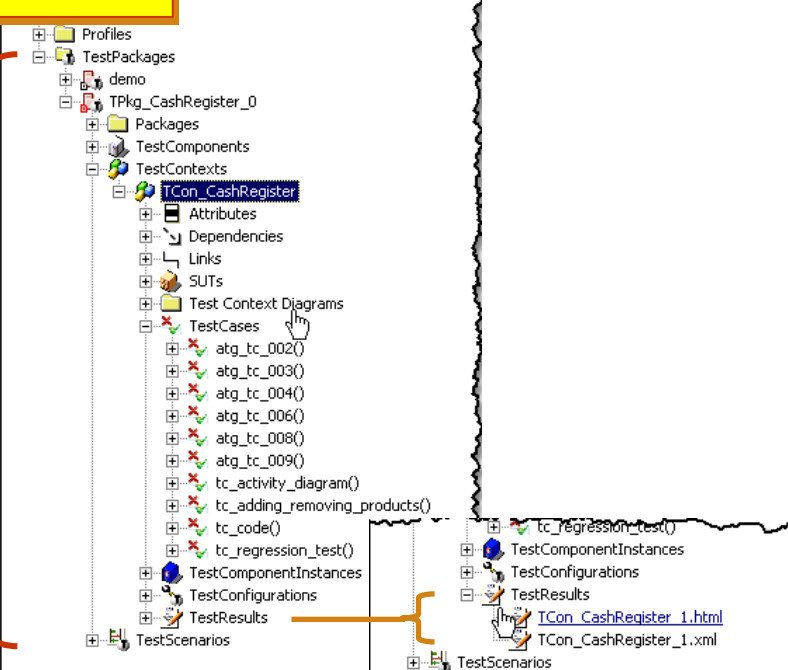


Procesos de diseño y pruebas completamente integrados

Diseño



Pruebas



- Fácil navegación entre el diseño y los artefactos de prueba, son parte del mismo modelo
- Trazabilidad de requisitos a test cases
- Diseño y pruebas siempre sincronizados y actualizados
- Generación automática de informes de resultados

Test Context Result

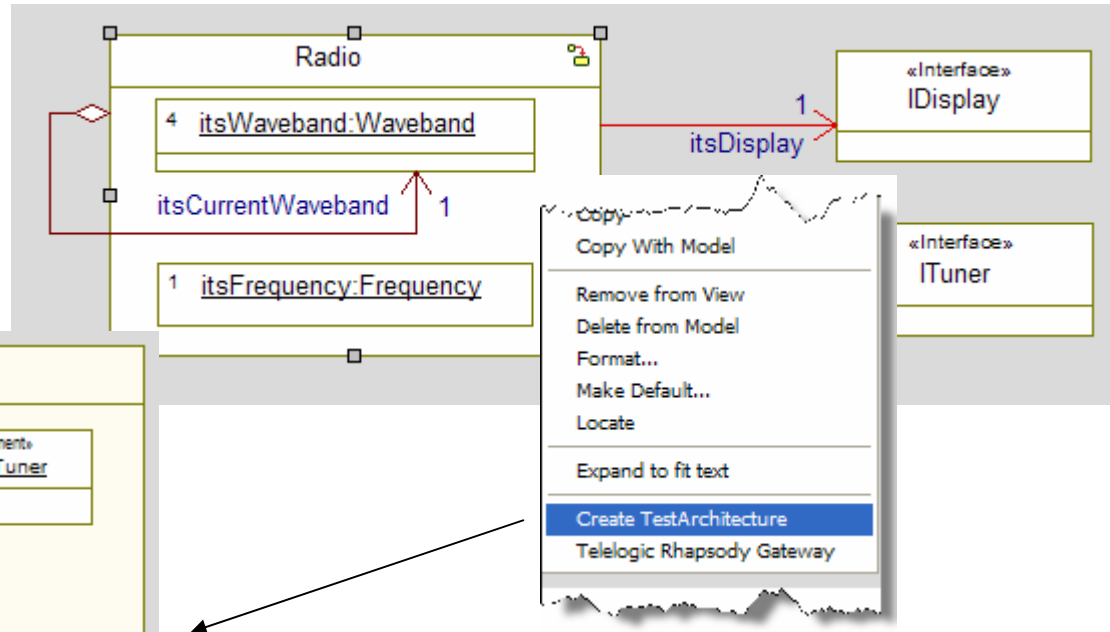
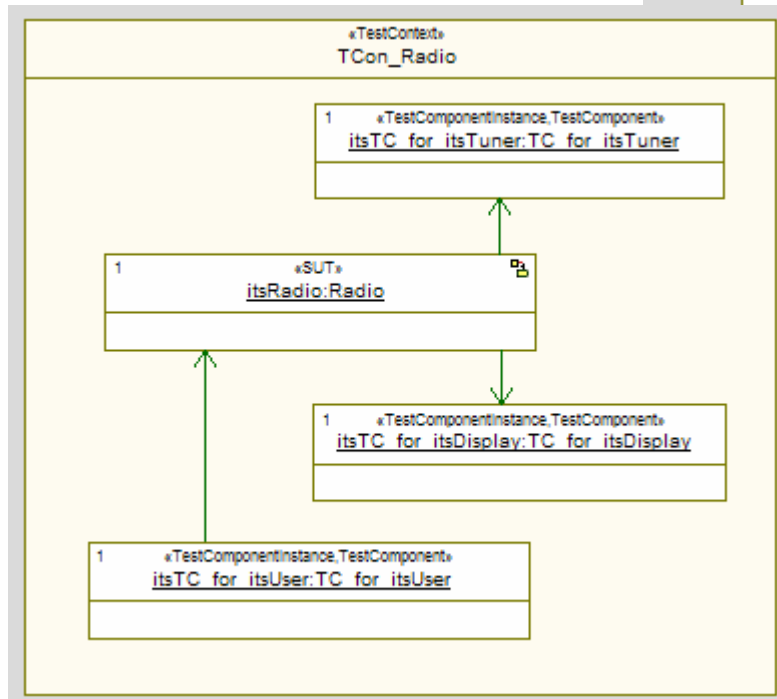
Informe de resultados

| Environment Info | |
|---------------------------------|---------------------------|
| Test executed on machine: | NBOSC-21-1 |
| Test executed by user: | ubrockmeyer |
| Used OS version: | Windows 2000 / Windows XP |
| Used Rhapsody version: | Aries, build 799102 |
| Used TestConductor version: | 2.0, build 530 |
| Tested Project | |
| Project: | CashRegister |
| Active Component: | TCon_CashRegister_5 |
| Active Configuration: | DefaultConfig |
| Test Context: TCon_CashRegister | Summary: PASSED |
| tc_code | PASSED |
| tc_activity_diagram | PASSED |
| tc_adding_removing_products | PASSED |
| tc_regression_test | PASSED |
| atg_tc_008 | PASSED |
| atg_tc_009 | PASSED |
| atg_tc_006 | PASSED |
| atg_tc_002 | PASSED |
| atg_tc_003 | PASSED |
| atg_tc_004 | PASSED |

Proceso de pruebas: Crear arquitectura de pruebas

- Crear Test Architecture

- ▶ Creación automática
- ▶ Actualización automática, o
- ▶ Actualización manual



- Graphical Test Architecture

- ▶ Es sólo arquitectura, todavía no hay casos de prueba

Proceso de pruebas: creación de casos de prueba

■ Crear Test Architecture

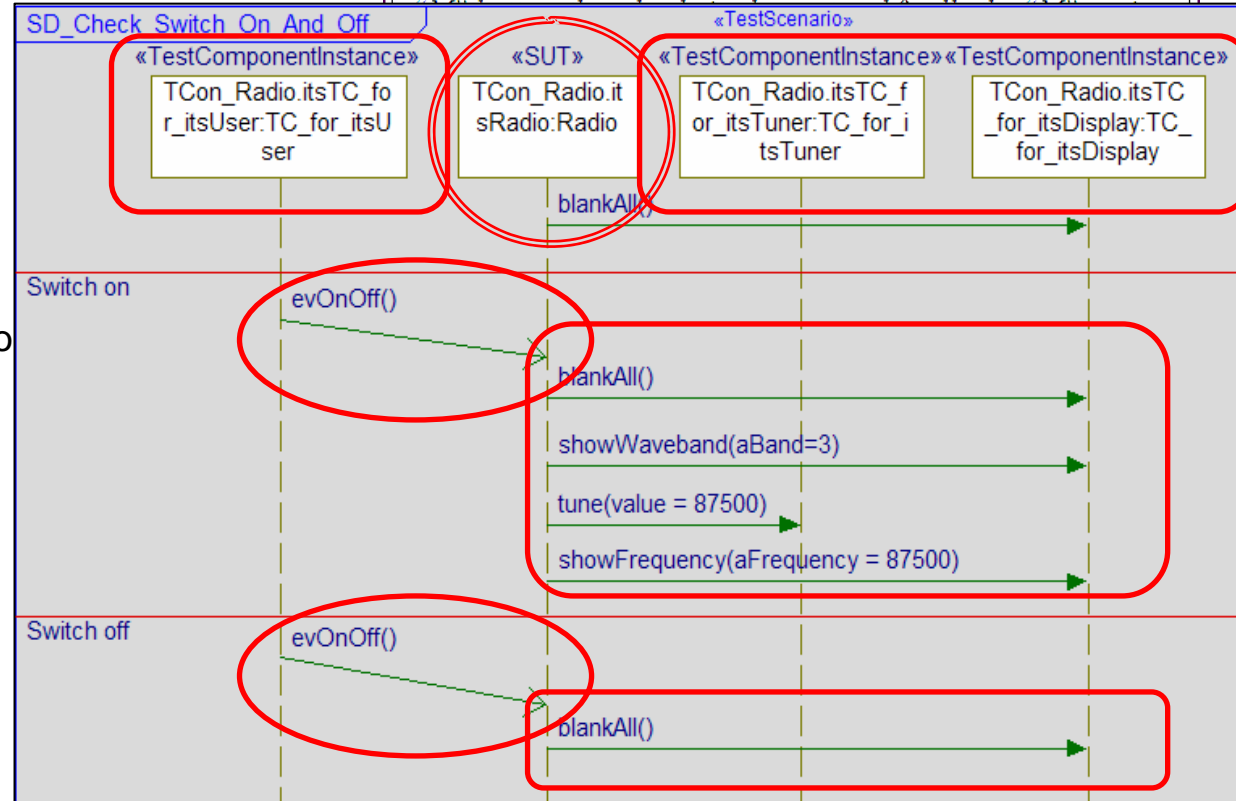
■ Casos de prueba como diag de secuencia

▶ Manualmente

- En base a
 - Requisitos Textuales
 - Experiencia previa o en el dominio
- Se define el comportamiento esperado del elemento a probar o SuT (System under Test)

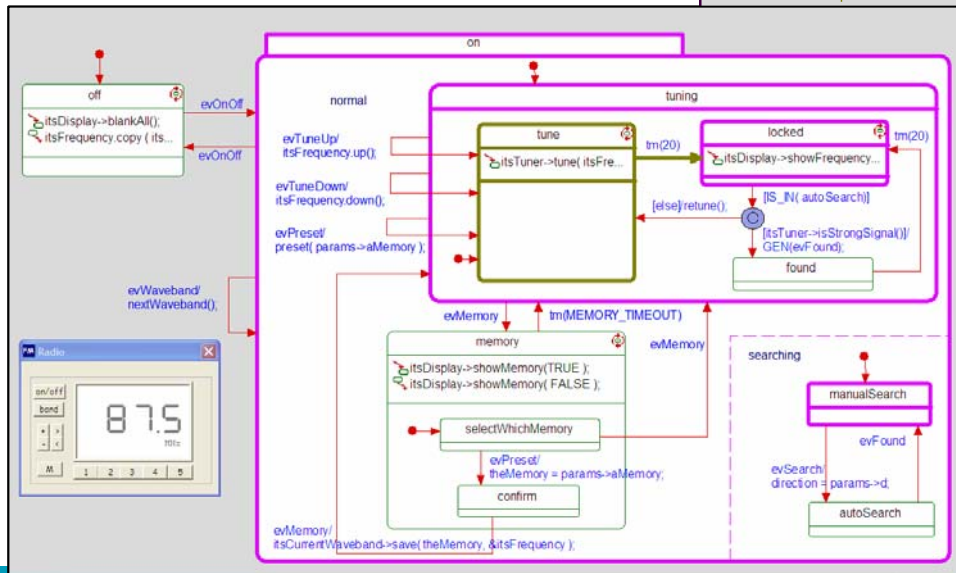
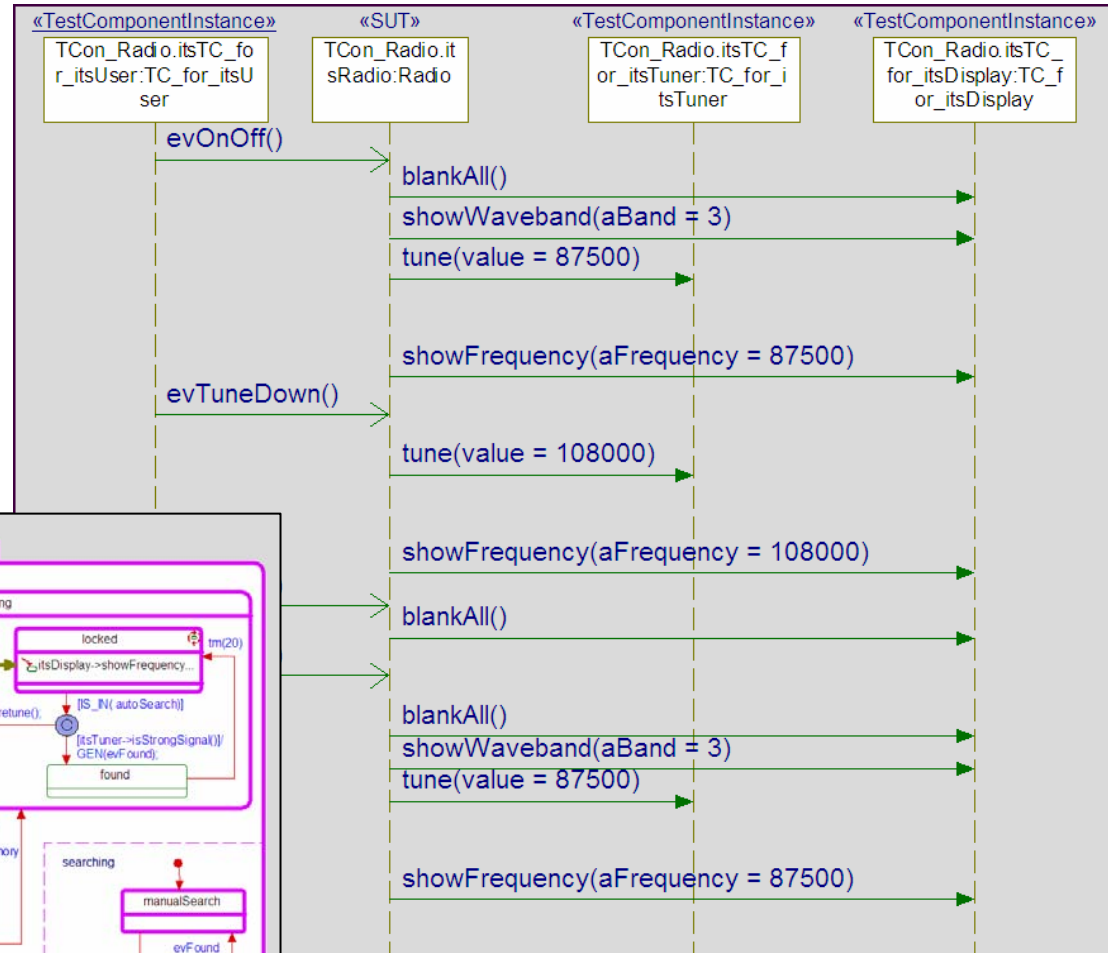
REQ014: *There are 5 memory preset buttons. When pressed, they recall a frequency. This frequency depends on the selected waveband. Thus it is possible to have 20 frequencies memorized, five for each waveband.*

REQ015: *There is an "M" button allowing the presets to be set to the current frequency. This is done by pressing the*



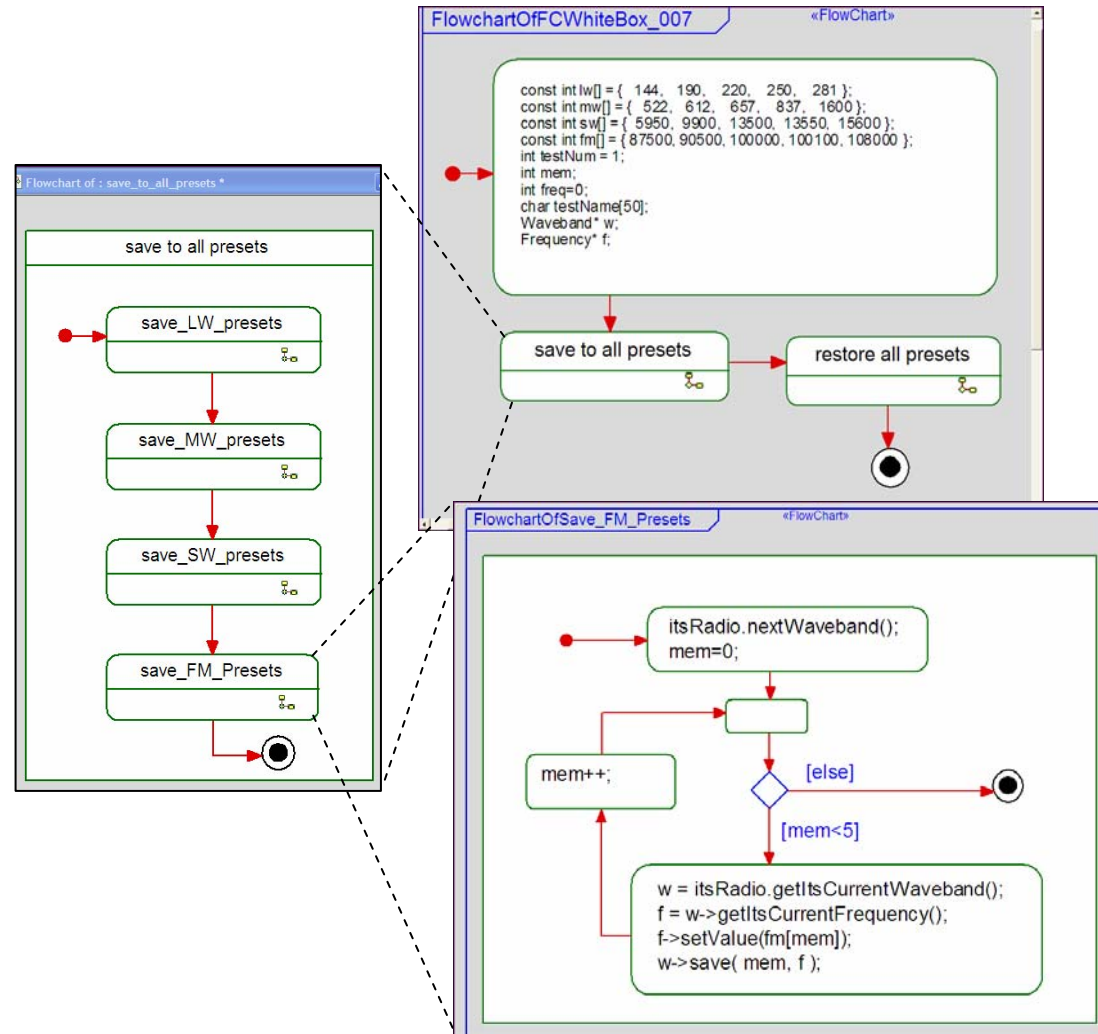
Proceso de pruebas: creación de casos de prueba

- Crear Test Architecture
- Casos de prueba como diag de secuencia
 - ▶ Manualmente
 - ▶ Grabación de simulaciones (diagramas de secuencia animados)



Proceso de pruebas: creación de casos de prueba

- Crear a Test Architecture
 - ▶ Manualmente
 - ▶ Grabación de simulaciones
- Casos de prueba como diag de secuencia
 - ▶ “Programación gráfica”
 - ▶ Intuitivos, potentes y fáciles de usar



Proceso de pruebas: creacion de casos de prueba

- Crear Test Architecture

- Casos de prueba como diag de secuencia

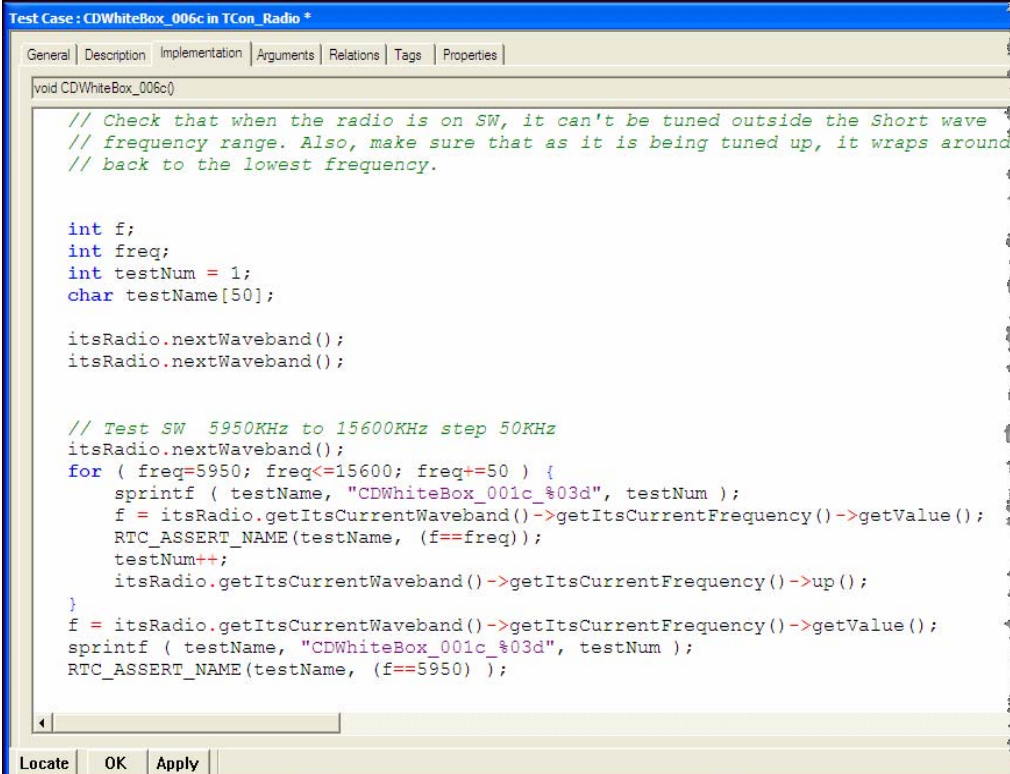
- ▶ Manualmente
- ▶ Grabación de simulaciones

- Casos de prueba como diagramas de flujo

- ▶ “Programación gráfica”
- ▶ Intuitivos, potentes y fáciles de usar

- Casos de prueba como código

- ▶ El comportamiento del caso de prueba se puede introducir directamente codificando



```
Test Case: CDWhiteBox_006c in TCon_Radio *
General | Description | Implementation | Arguments | Relations | Tags | Properties |
void CDWhiteBox_006c()
// Check that when the radio is on SW, it can't be tuned outside the Short wave
// frequency range. Also, make sure that as it is being tuned up, it wraps around
// back to the lowest frequency.

int f;
int freq;
int testNum = 1;
char testName[50];

itsRadio.nextWaveband();
itsRadio.nextWaveband();

// Test SW 5950KHz to 15600KHz step 50KHz
itsRadio.nextWaveband();
for ( freq=5950; freq<=15600; freq+=50 ) {
    sprintf ( testName, "CDWhiteBox_001c_%03d", testNum );
    f = itsRadio.getItsCurrentWaveband()->getItsCurrentFrequency()->getValue();
    RTC_ASSERT_NAME(testName, (f==freq));
    testNum++;
    itsRadio.getItsCurrentWaveband()->getItsCurrentFrequency()->up();
}
f = itsRadio.getItsCurrentWaveband()->getItsCurrentFrequency()->getValue();
sprintf ( testName, "CDWhiteBox_001c_%03d", testNum );
RTC_ASSERT_NAME(testName, (f==5950) );

Locate OK Apply
```

Proceso de pruebas: ejecucion y resultados

- Execute TestCase
- de flujo
- Casos de prueba como código
- Ejecución e informe de resultados
 - ▶ Comportamientos inesperados se resaltan

| Name | File | Line | Result |
|---------------------|----------------|------|--------|
| CDWhiteBox_001b_114 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_115 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_116 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_117 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_118 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_119 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_120 | TCon_Radio.cpp | 143 | PASSED |
| CDWhiteBox_001b_121 | TCon_Radio.cpp | 149 | FAILED |

Buttons: Activate, Show Assertion, Quit

bx_002b, Instance switch on FM Radio again, Iteration 1

«ComponentInstance» «SUT» «TestComponentInstance» «TestComponentInstance»

radio.itsTC_b TCon_Radio.it TCon_Radio.itsTC_f TCon_Radio.itsTC

r.TC_for_itsU sRadio:Radio or itsTunerTC_for i for itsDisplay:TC

evOnO ff)

Mozilla Firefox

file:///D:/Profiles/ismoco/Desktop/My%20Papers/Radio%20Model/Full%20Test%20Report.htm

Table of Contents

- Test Report of Model V71_RICpp_Radio
 - TCon_Radio
 - System Under Test (SUT)
 - Test Component Instances
 - Test Context Diagrams
 - Test Cases
 - Test Case CDWhiteBox_006a
 - Test Objectives
 - Test Objective WB_TST006
 - Test Case CDWhiteBox_006b
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case CDWhiteBox_006c
 - Test Objectives
 - Test Scenario check radio can be
 - Test Case CDWhiteBox_006d
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case FCWhiteBox_007
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case SDWhiteBox_001
 - Test Objectives
 - Test Scenario check radio can be
 - Test Case SDWhiteBox_002a
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case SDWhiteBox_002b
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case SDWhiteBox_003
 - Test Objectives
 - Test Objective WB_TST007
 - Test Case SDWhiteBox_004
 - Test Objectives
 - Test Objective WB_TST007

Test Context Result

Test Context: TCon_Radio

Fri Sep 21 22:01:37 2007

Environment Info

| | |
|-----------------------------|---------------------------|
| Test executed on machine: | TEMPRANILLO |
| Test executed by user: | ukmari |
| Used OS version: | Windows 2000 / Windows XP |
| Used Rhapsody version: | 7.1, build896637 |
| Used TestConductor version: | 2.0, build 646 |

Tested Project

| | |
|-----------------------|-----------------|
| Project: | V71_RICpp_Radio |
| Active Component: | TPkg_Radio_Comp |
| Active Configuration: | TC |

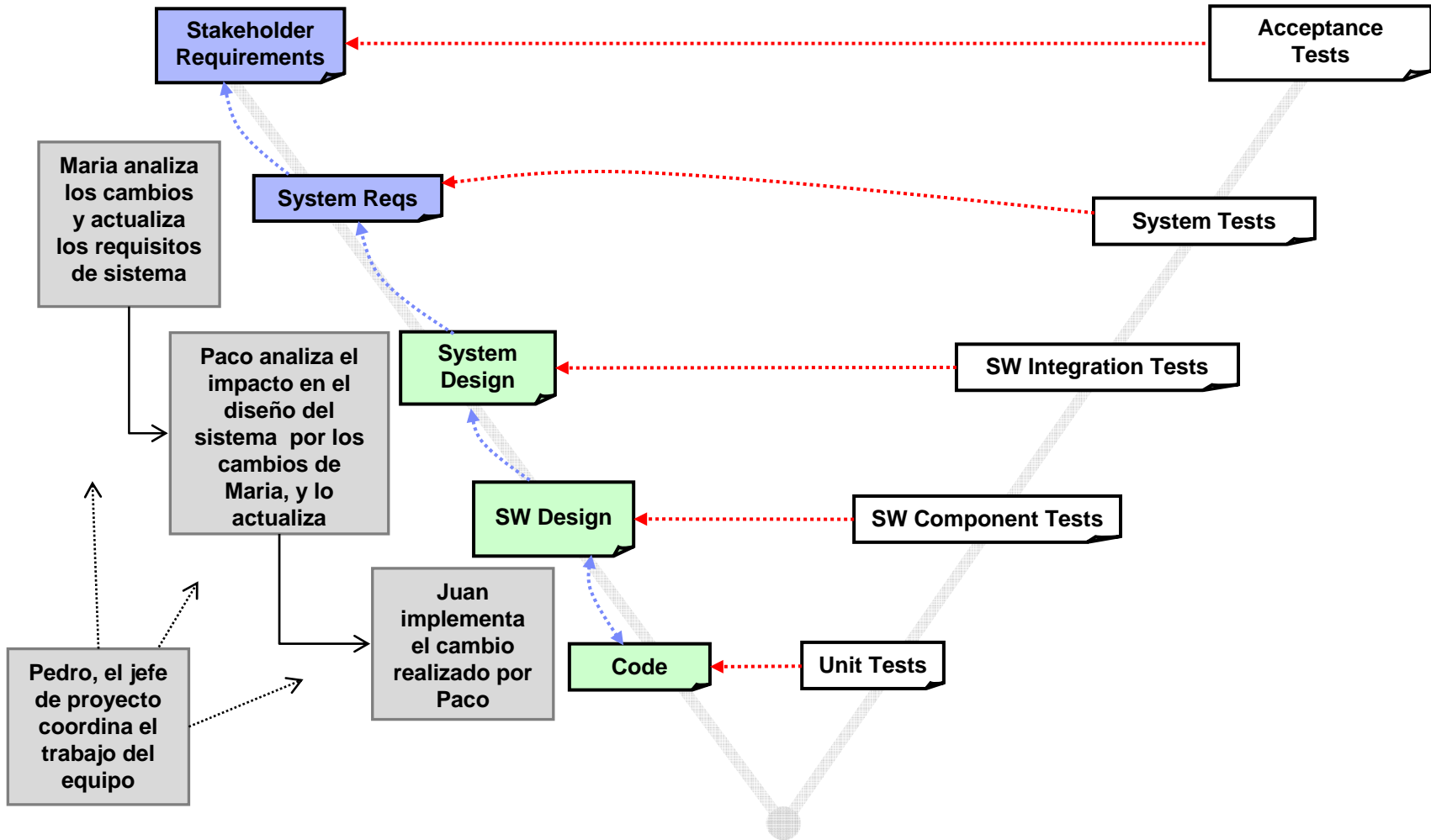
| Test Context: TCon_Radio | Summary: FAILED |
|--------------------------|-----------------|
| CDWhiteBox_006a | PASSED |
| CDWhiteBox_006b | PASSED |
| CDWhiteBox_006c | FAILED |
| CDWhiteBox_006d | PASSED |
| FCWhiteBox_007 | PASSED |
| SDWhiteBox_001 | PASSED |
| SDWhiteBox_002a | PASSED |
| SDWhiteBox_002b | FAILED |
| SDWhiteBox_003 | PASSED |
| SDWhiteBox_004 | PASSED |

Colaboración

- Soporte a equipos de desarrollos de cualquier tamaño
 - ▶ En el mayor despliegue hay 1200 usuarios
- Gestión del desarrollo en paralelo con comparador gráfico de diferencias entre modelos (también merge)
- Integración con herramientas de control de configuración de software como Rational ClearCase®, Rational Synergy™ y Rational Team Concert



Colaboración – Integración con Rational Team Concert



Colaboración – Integración con Rational Team Concert

Escenario ejemplo

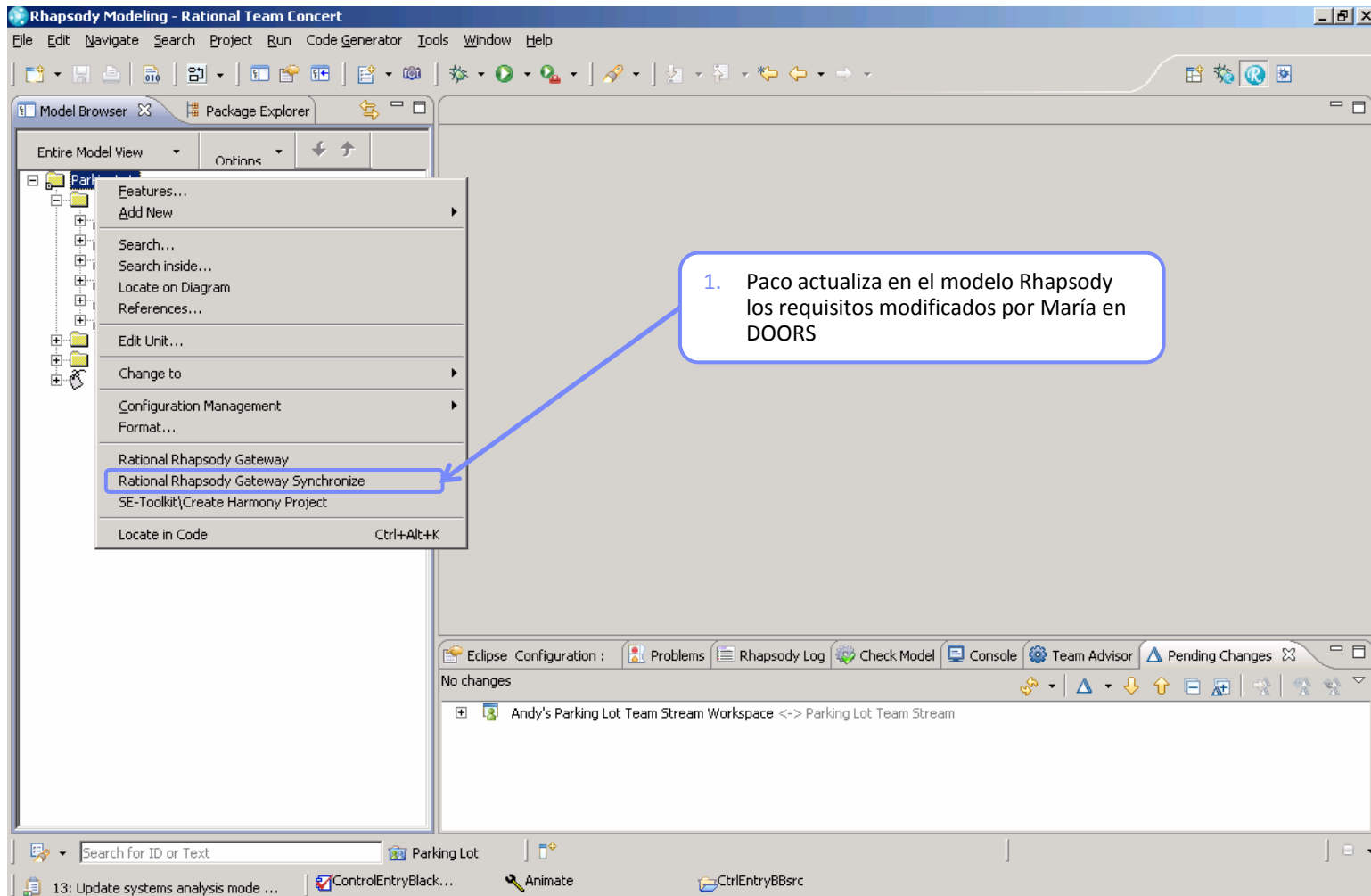
The screenshot displays the Rational Team Concert (RTC) interface. On the left, a 'My Work' tab is highlighted with a red box. Below it, the 'Inbox (Parking Lot)' shows a work item: 'Update systems analysis model for functional requirement 25 and allocate functionality to subsystems design as needed'. The main workspace shows a list of work items categorized by time: 'Today (1 items)', 'Later This Week (1 items)', and 'Next Week (1 items)'. A callout box with a blue border contains the following steps:

1. Pedro asigna trabajo a Paco
2. Paco revisa el trabajo que le han asignado y lo acepta
3. Paco comienza a trabajar
4. RTC guarda todas las modificaciones realizadas a partir de ahora

Arrows from the callout box point to the 'My Work' tab, the work item in the 'Today' list, and the RTC interface. The RTC interface shows a 'Pending Changes' section with 'No changes' and a workspace view for 'Andy's Parking Lot Team Stream Workspace <-> Parking Lot Team Stream'.

Colaboración – Integración con Rational Team Concert

Escenario ejemplo



Colaboración – Integración con Rational Team Concert

Escenario ejemplo

The screenshot displays the Rational Team Concert interface. On the left, the Model Browser shows a package hierarchy for 'ParkingLot', including 'ControlEntryPkg'. The main workspace shows a statechart diagram for 'ProcessingGuest'. The diagram includes states 'PG_IDLE', 'PgOpenEntry', and 'CheckingLot'. Transitions are labeled with guard conditions and actions, such as '[correctLot]/ reportGuest()' and '[else]/ reportWrongLot()'. A callout box highlights a change in the diagram.

1. Paco modifica el diagrama de estados para reflejar en el diseño el cambio en indicado en el requisito

Colaboración – Integración con Rational Team Concert

Escenario ejemplo

For Help, press F1

ModelComponent

- Unresolved
- ParkingLot
- ParkingLot/ParkingLot_rpy
- snapshots
- ControlEntryPkg.sbs
- DOORSReqs.sbs
- UseCaseDiagramsPkg.sbs

LOCK

Unlock

Compare With

Expand Children

Search for ID or Text

Parking Lot

13: Update systems analysis mode ...

1. Paco revisa los cambios realizados
2. Comparación de diagramas

Colaboración – Integración con Rational Team Concert

Escenario ejemplo

The screenshot displays the Rational Team Concert interface. The main window shows a statechart diagram for a parking lot system. The diagram includes states like 'CheckingLot' and 'processGuest', with transitions labeled with conditions and actions such as '[correctLot]/ reportGuest()' and '[else]/ reportWrongLot()'. A context menu is open over the diagram, listing various actions. The 'Check-in and Deliver...' option is highlighted, and a blue callout box points to it with the text: '1. Paco tras revisar y aceptar comparte el trabajo realizado con el resto del equipo'. The interface also shows a Package Explorer on the left and a bottom status bar.

1. Paco tras revisar y aceptar comparte el trabajo realizado con el resto del equipo

Colaboración – Integración con Rational Team Concert

Escenario ejemplo

1. Tras compartir el trabajo realizado con el resto del equipo Paco da por concluido el trabajo "Stop Working"
2. Paco cambia el estado del work item a "Resolve"
3. Pedro, es notificado de que Paco ha concluido su trabajo
4. Pedro, asigna trabajo a Juan para que implemente los cambios realizados por Paco.

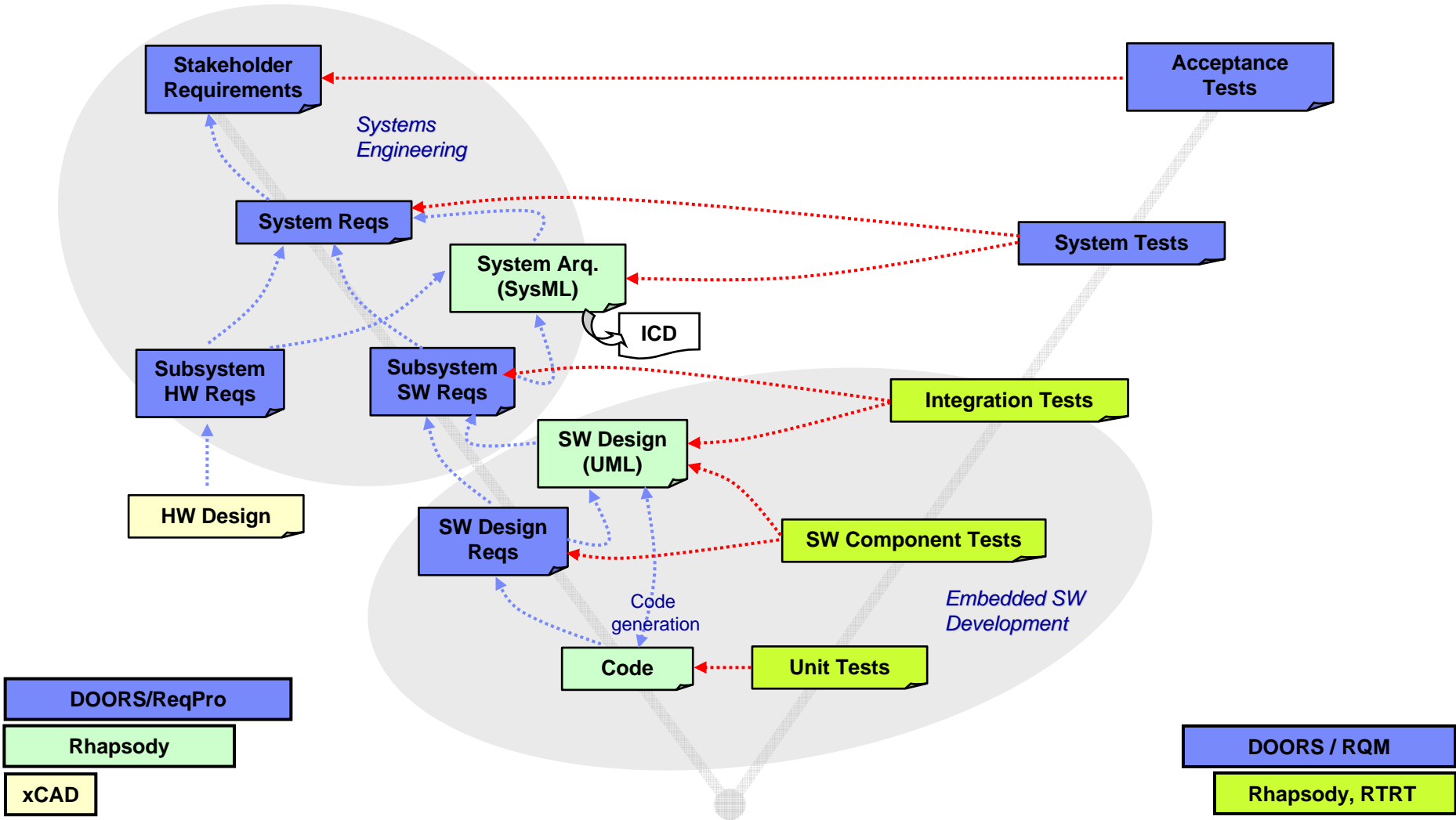
The screenshot displays the Rational Team Concert (RTC) interface. The main window shows a work item titled "Task 13" with the summary "Update systems analysis model for functional requirement 25 and allocate functionality to subsystems design as needed". The state dropdown menu is open, showing options: "New", "Start Working", "Resolve", and "Triage". The "Resolve" option is currently selected. The interface also shows a "Current Work" sidebar with a list of tasks, including "Update systems analysis model for functional requirement 25 and allocate functionality to subsystems design as needed" (due 4d, 13) and "Allocate Actions To Sub-Blocks" (due 3d, 9). The bottom status bar indicates "No changes" and "Andy's Parking Lot Team Stream Workspace".

Resumen - IBM Rational Rhapsody®

- Solución MDD de IBM Rational para el diseño y desarrollo de sistemas complejos, embebidos y de tiempo real
- Detección y **eliminación** temprana de **errores** en diseño mediante la simulación del sistema (verificación temprana)
- **Incrementa la productividad del desarrollador** mediante la generación automática de código y de la documentación, y DMCA.
- Mejora la **comunicación** con el cliente y entre disciplinas mediante la utilización de UML/SysML
- Visualización de los **requisitos** y trazabilidad al modelo y **trazabilidad**:
 - ▶ Asegura que el **diseño cumple los requisitos**
 - ▶ **Impacto de cambios**
- Mejora la **colaboración** entre equipos de desarrollo distribuidos
- **Incrementa la productividad del equipo de pruebas**:
Pruebas dirigidas por el modelo (*Model Driven Testing – MDT*)

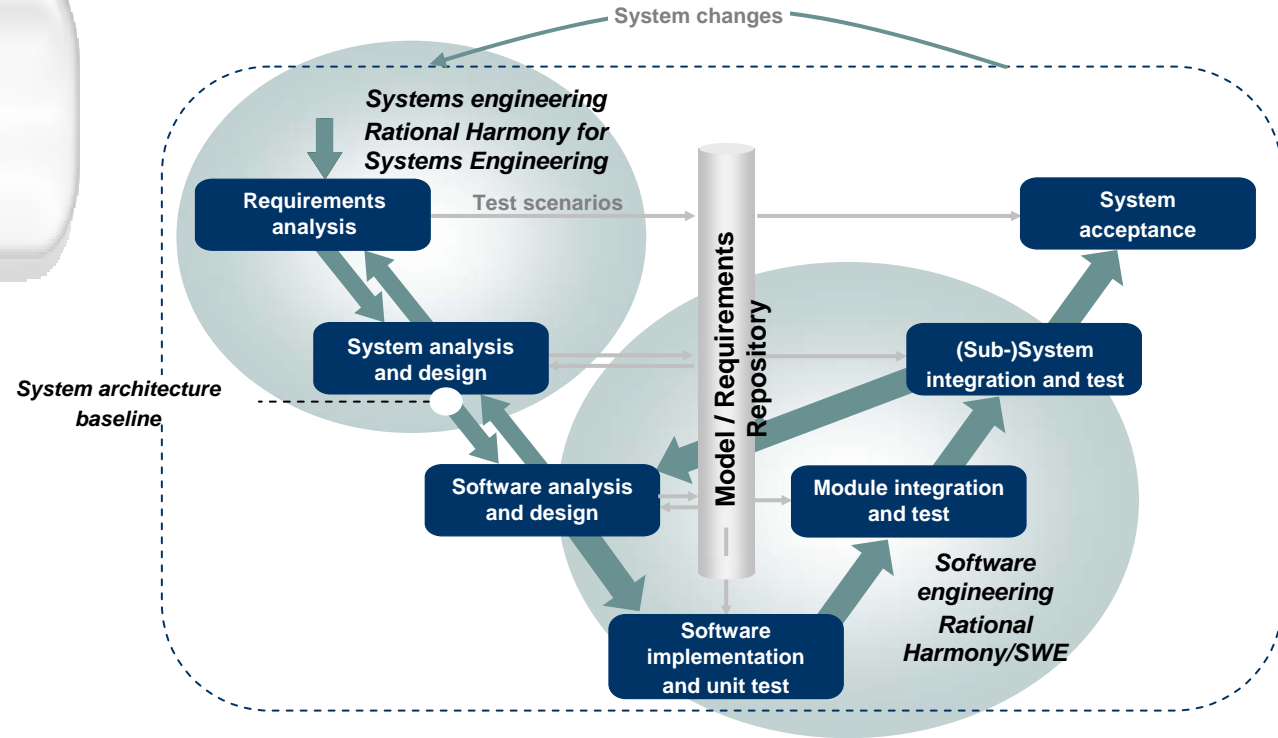


Proceso, herramientas...



... y Rational Harmony

Harmony es la librería de buenas prácticas para el desarrollo de software y sistemas



Dos versiones:

- Rational Harmony/SE
- Rational Harmony/SWE



Francisco J. López Minaya
Rational Technical Solution Architect
francisco.lopezminaya@es.ibm.com

© Copyright IBM Corporation 2008. 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.