IBM Software Group

Diseño y Modelado de Sistemas

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

Go to IBM

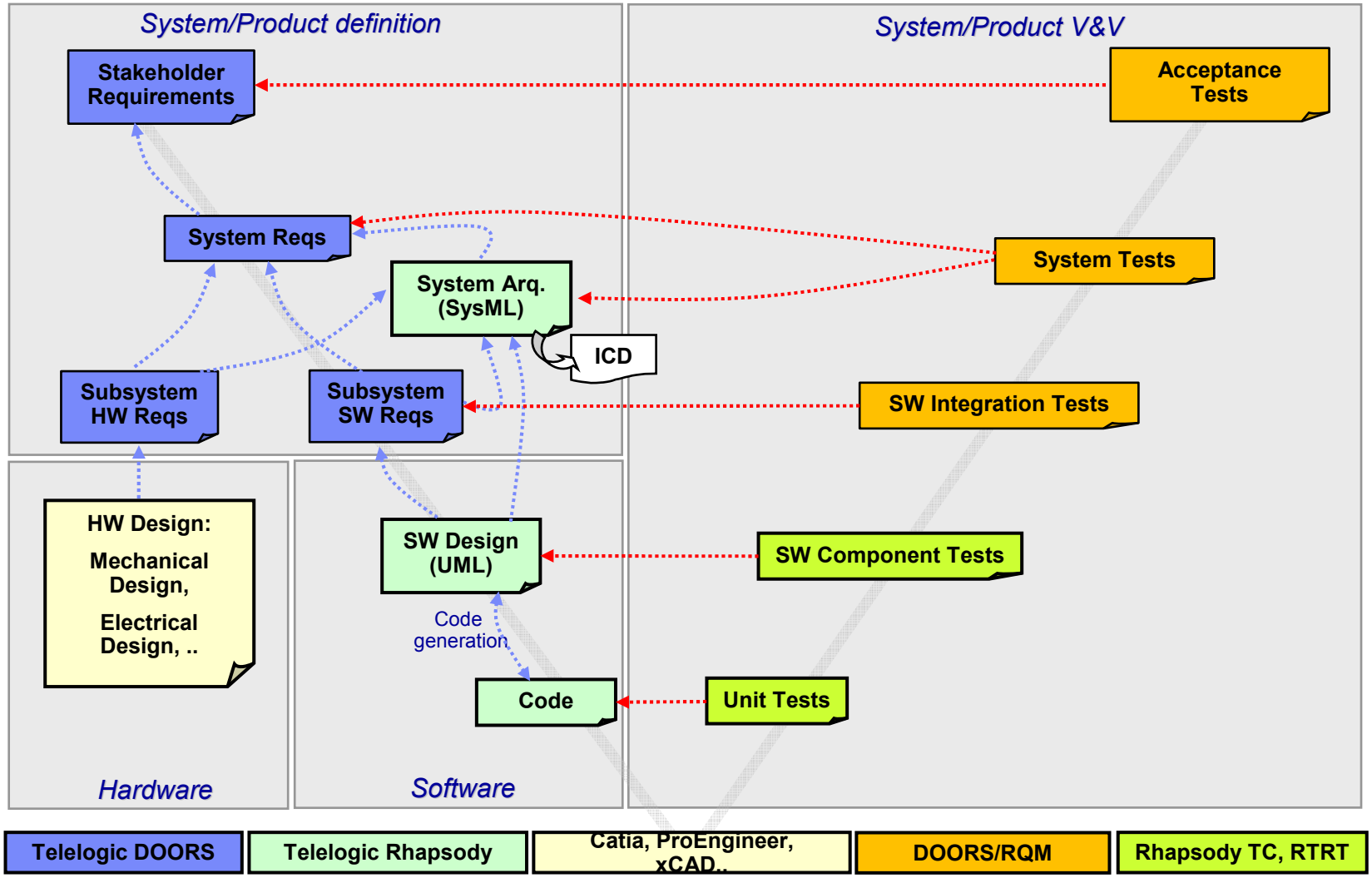
IV Mejores Prácticas en el Desarrollo de Sistemas
27 Mayo 2009 - Madrid

IBM Software Group

El proceso de desarrollo y construcción de sistemas

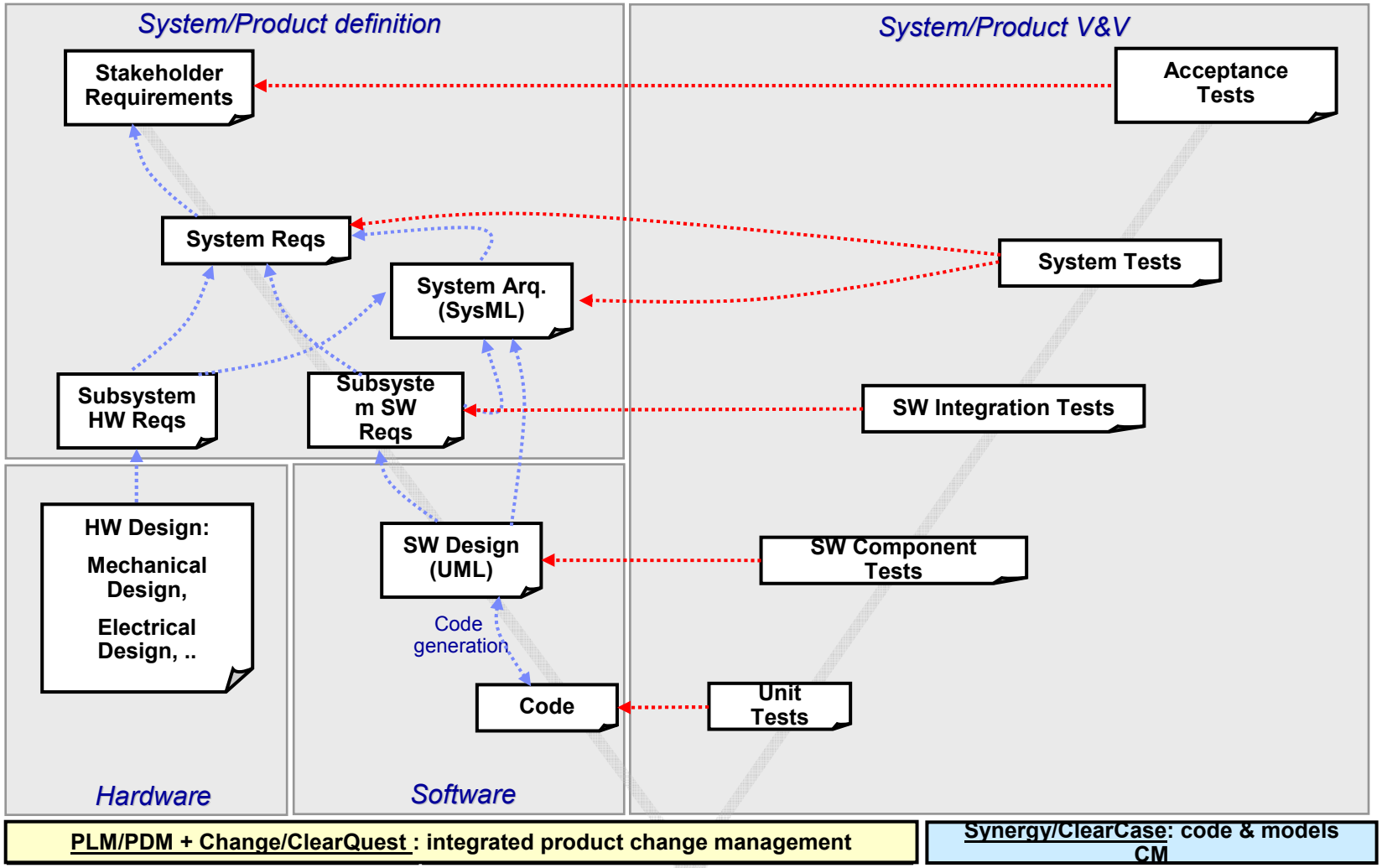
Plataforma para desarrollo de sistemas (I)

Definición, Desarrollo, Construcción y Verificación & Validación



Plataforma para desarrollo de sistemas (II)

Gestión de Cambios y de la Configuración

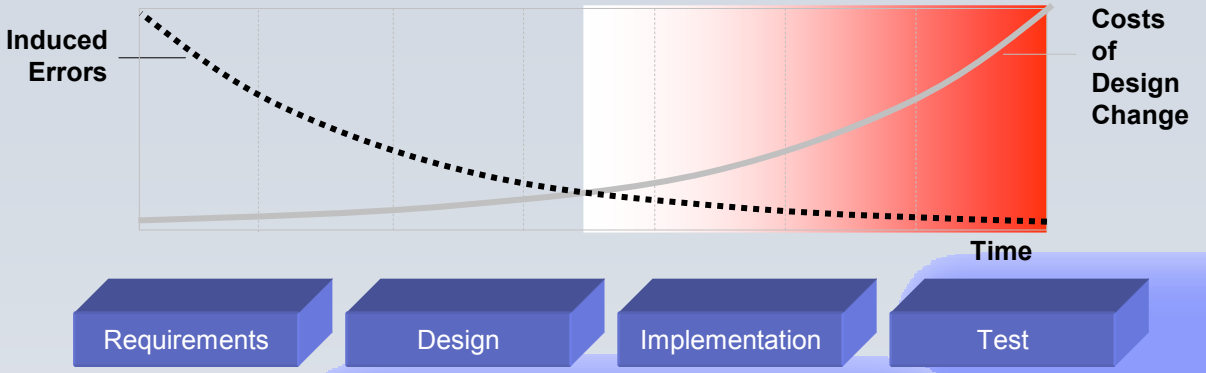


¿A qué nos enfrentamos en la realidad?

- **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).**
- **Baja productividad en procesos de prueba. Procesos lentos y poco ágiles**
- **La colaboración entre equipos grandes de desarrollo que están distribuidos es cada vez más difícil.**
- **No hay comunicación efectiva entre diferentes disciplinas: ingenieros de sistemas, ingenieros mecánicos, ingenieros eléctricos, ingenieros de software**
- **Cada vez es más difícil la gestión y la integración de diseños cada vez más complejos**
- **Poca reutilización, se vuelve a construir algo que ya existía**

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

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



$80/defecto

Durante la fase de requisitos

$240/defecto

Durante la fase de diseño y construcción

$960/defecto

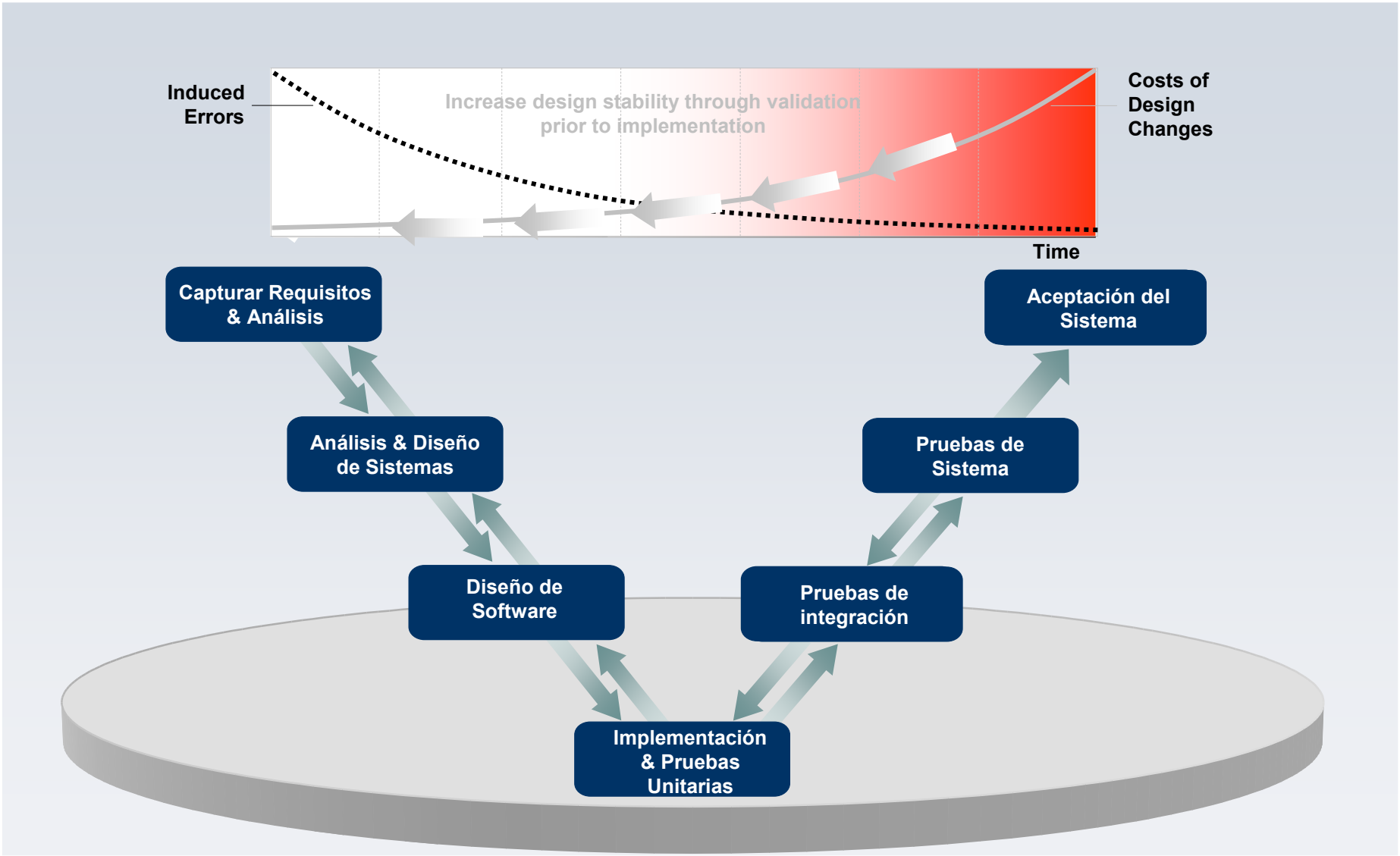
Durante la fase de testing

$7,600/defecto

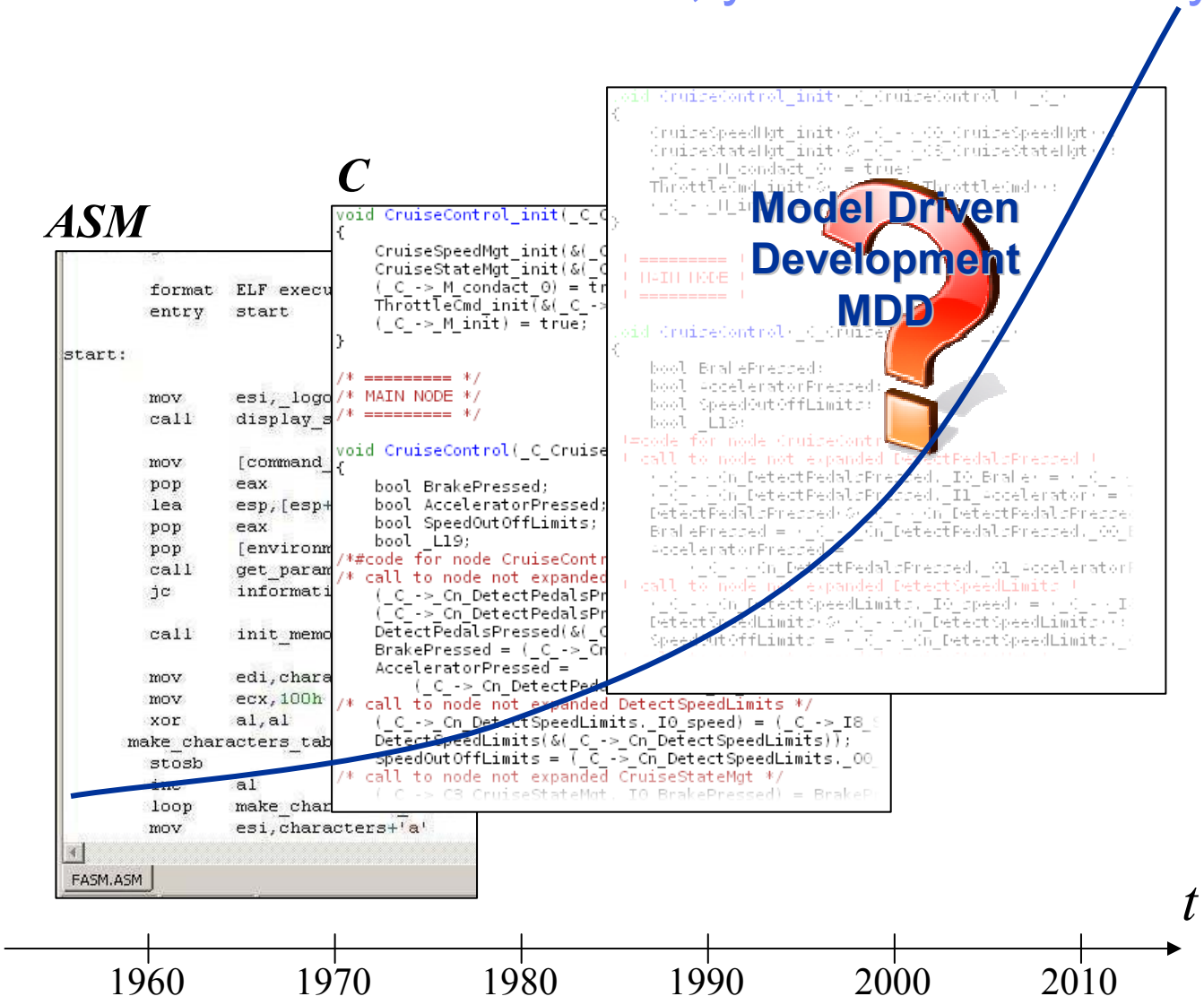
En producción

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

Cuanto antes pruebe antes encuentro defectos…



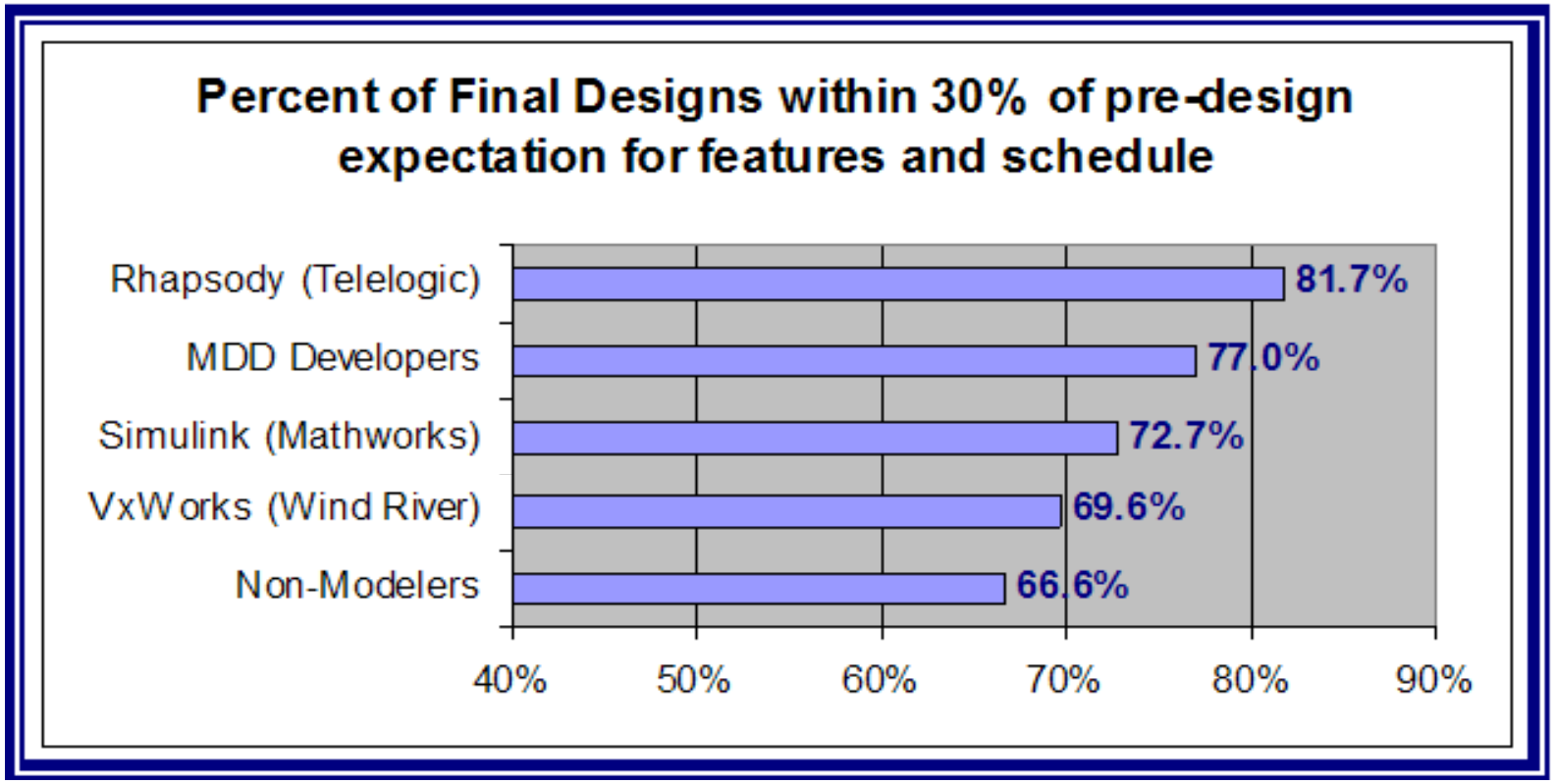
Los sistemas evolucionan, y cómo se construyen también…



Model Driven Development (MDD):

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

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

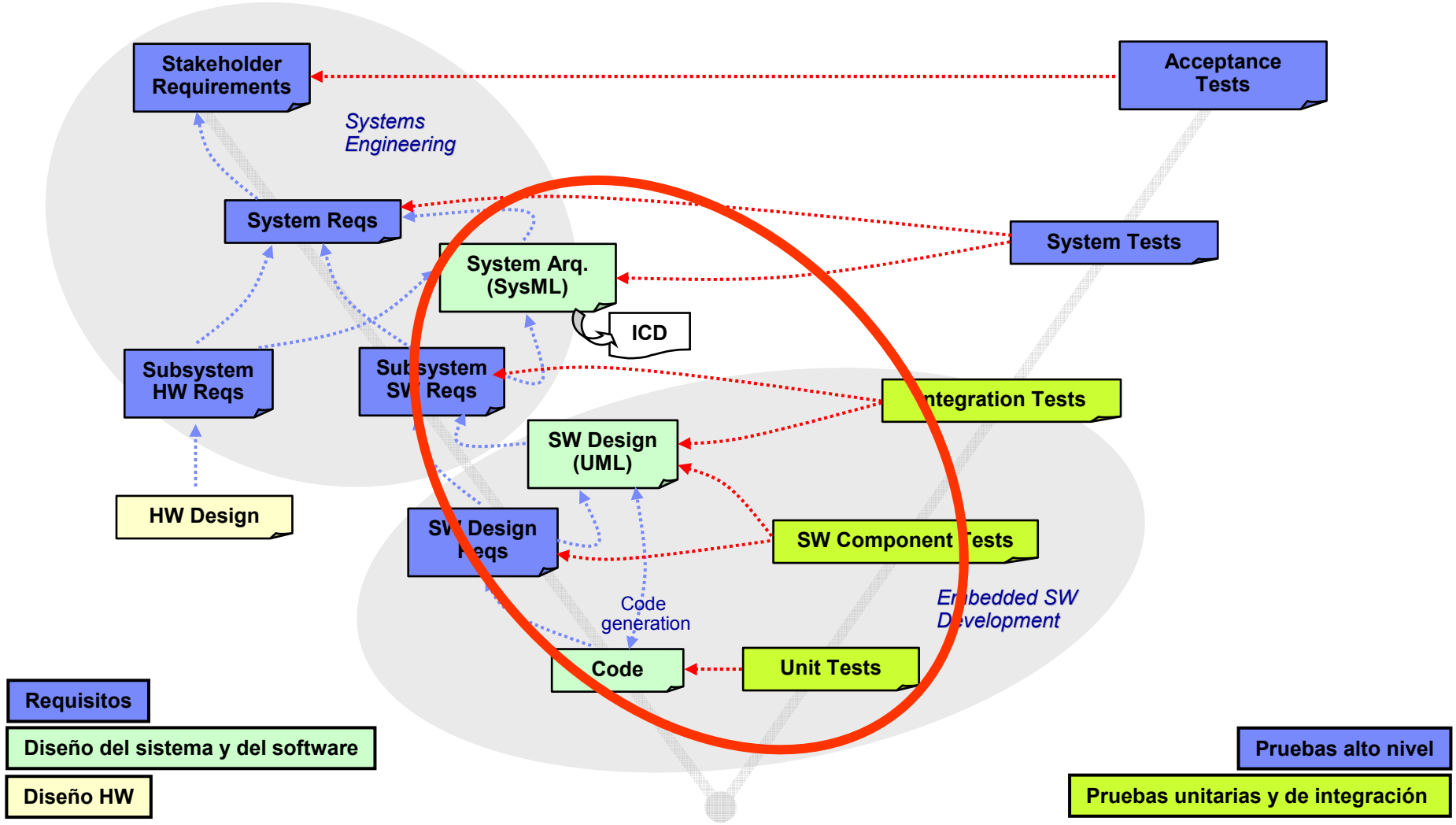


Embedded Market Forecasters
*Documented in "**What Do You Do When the Horse You're Riding Drops Dead? Why Model Driven Design is Emerging as a Preferred Best Practice", March 2007***

IBM Telelogic Rhapsody®

- **Entorno MDD para el diseño y desarrollo de sistemas**
- **Trazabilidad a los requisitos asegura que el diseño cumple los requisitos del cliente**
- **Incrementa la productividad del desarrollador. Simulación del sistema mediante ejecución de modelos. La verificación temprana y continua ayuda a detectar errores en el momento de introducirse.**
- **Mejora la comunicación. Visualización de los requisitos, la estructura y el comportamiento del sistema a través de UML/SysML**
- **Mejora la colaboración entre equipos de desarrollo distribuidos**
- **Generación automática de código a partir de los modelos**
- **Trabajar desde el código o desde el modelo**
- **Incrementa la productividad del equipo de pruebas: Pruebas dirigidas por el modelo *(Model Driven Testing - MDT)***
- **Generación automática de documentación**

Proceso de desarrollo genérico construcción y desarrollo de sistemas



IBM Software Group

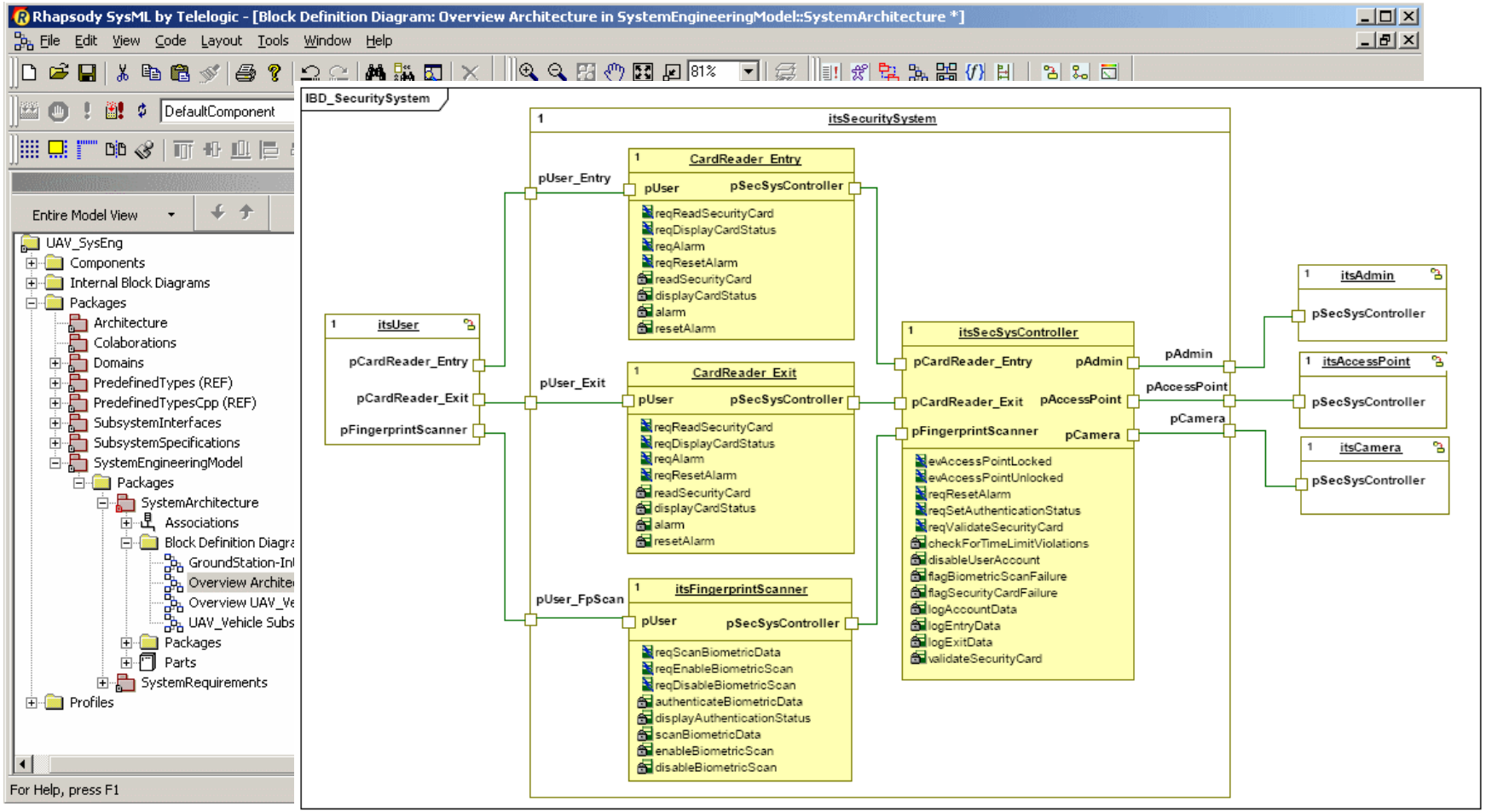
Arquitectura del sistema

Rational. software

Go to IBM

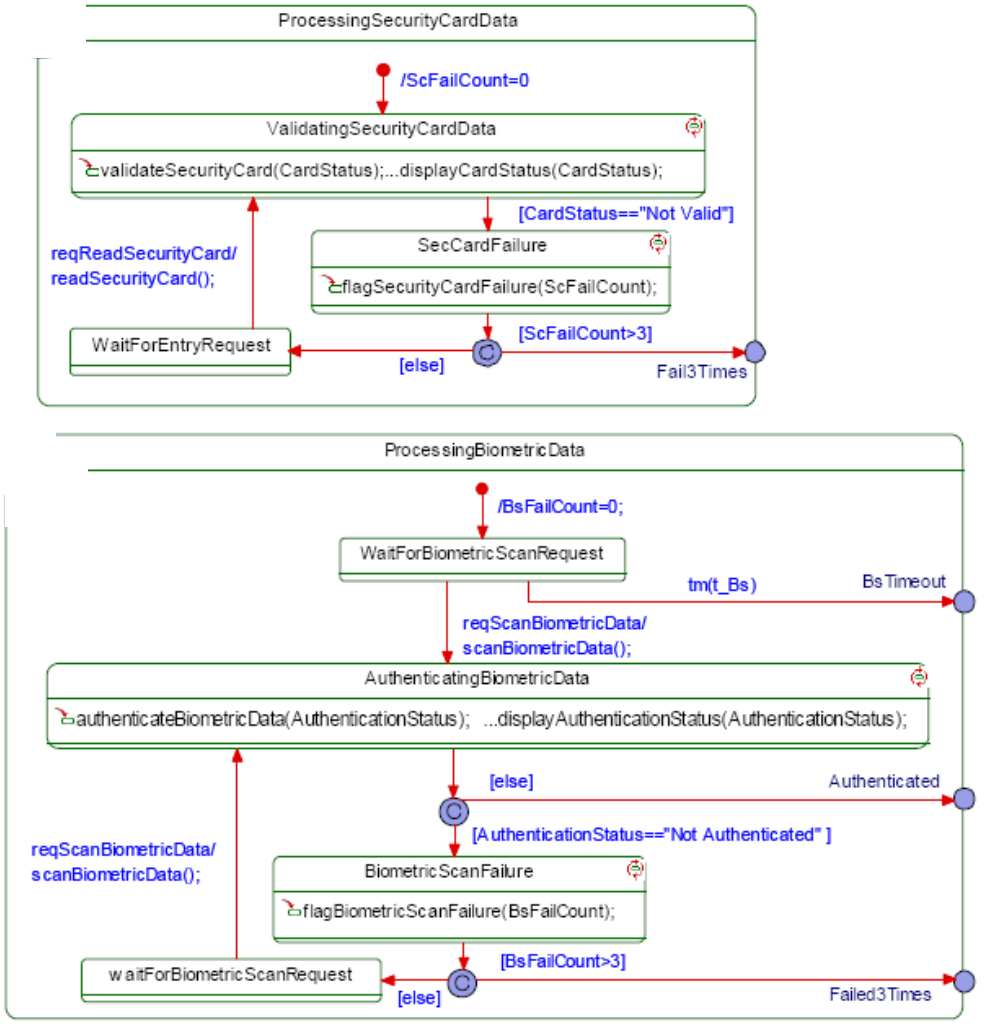
Diseño de la arquitectura del sistema

- **Estructura: Block Definition Diagrams e Internal Block Diagrams**



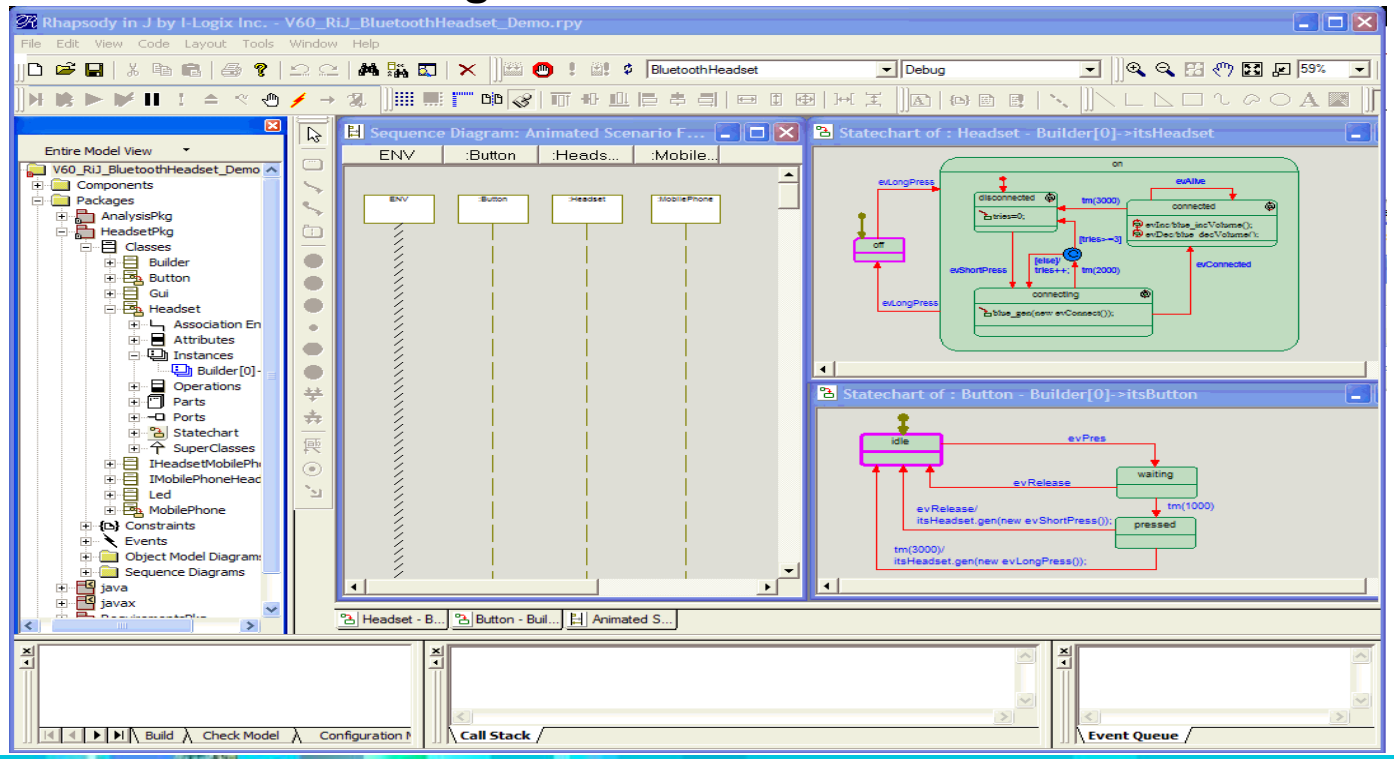
Diseño de la arquitectura del sistema

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



Verificación y validación temprana

- **La ejecución del modelo ayuda a detectar errores en las fases tempranas**
- **Promueve el desarrollo ágil, permite validar los requisitos en fases tempranas de diseño**
- **Visualización de la ejecución: Flujo de mensajes en diagrama de secuencia, y estado del sistema en diagrama de estados**



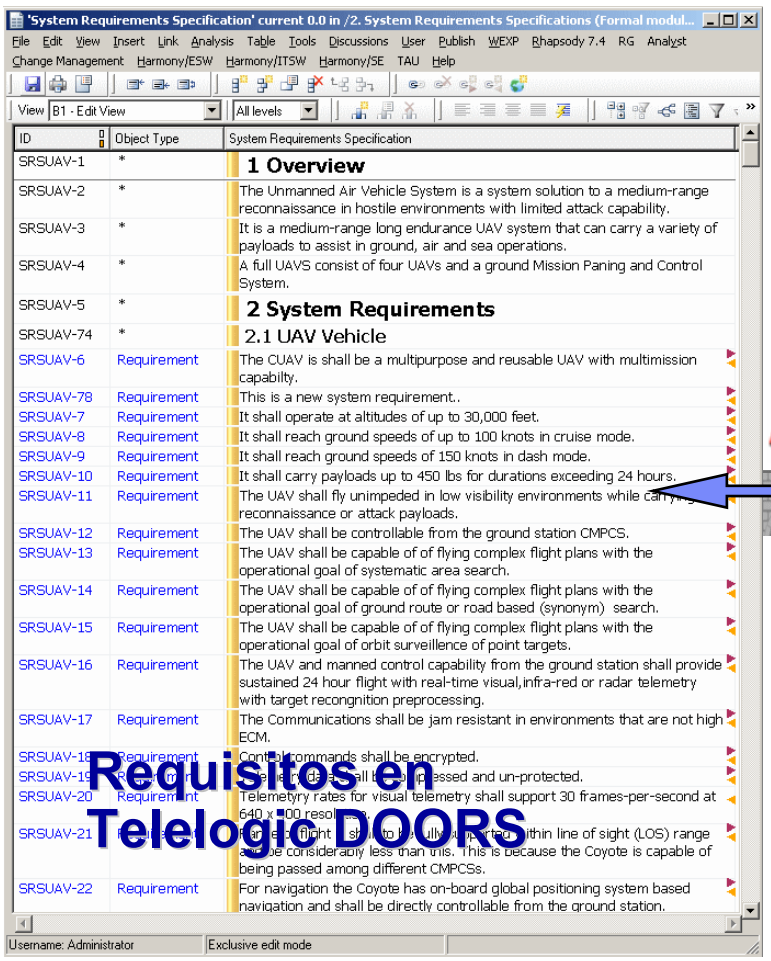
IBM Software Group

Trazabilidad requisitos del sistema y arquitectura

Trazabilidad de los requisitos de sistema al modelo

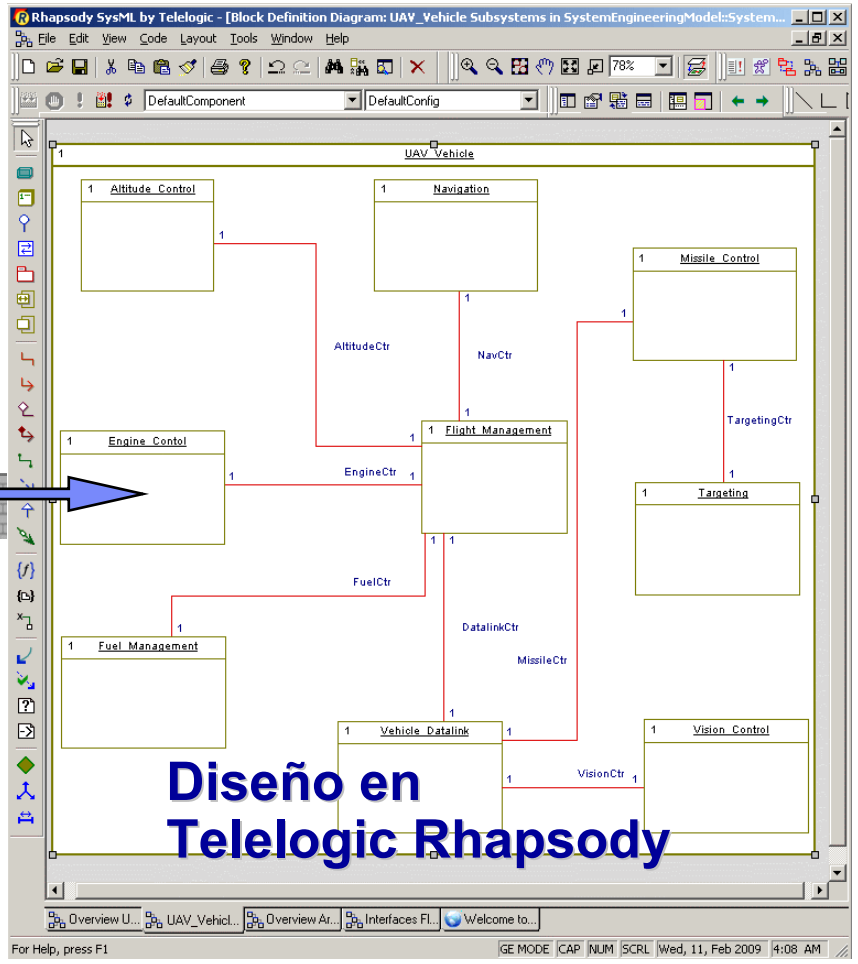
Asegurar que la arquitectura cumple los requisitos del sistema…

- **¿Cómo se establece la trazabilidad entre requisitos y elementos del diseño?**



¿Cómo?

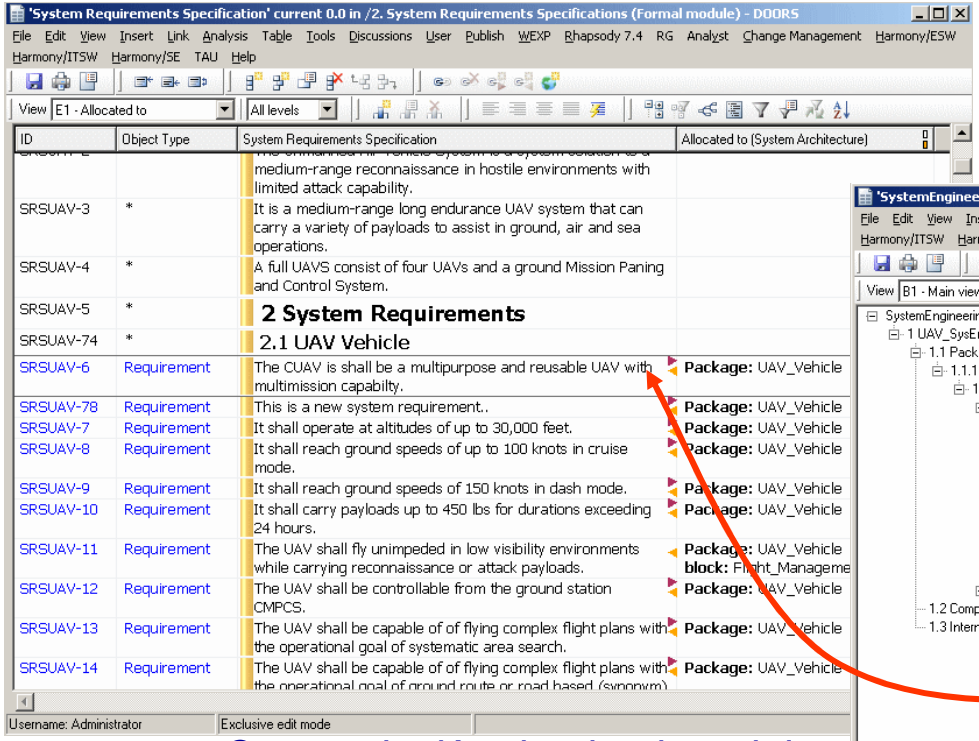
Gateway



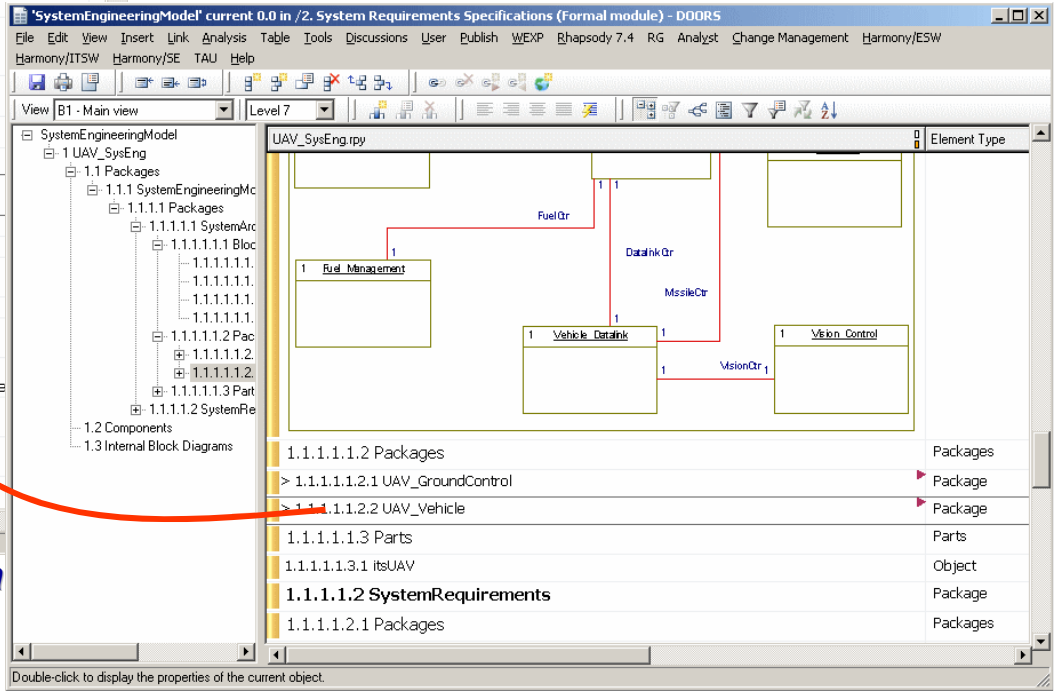
Trazabilidad de los requisitos de sistema al modelo

Asegurar que la arquitectura cumple los requisitos del sistema…

- **Las dependencias pueden establecerse desde DOORS ….**



Módulo DOORS con una copia de la información del modelo



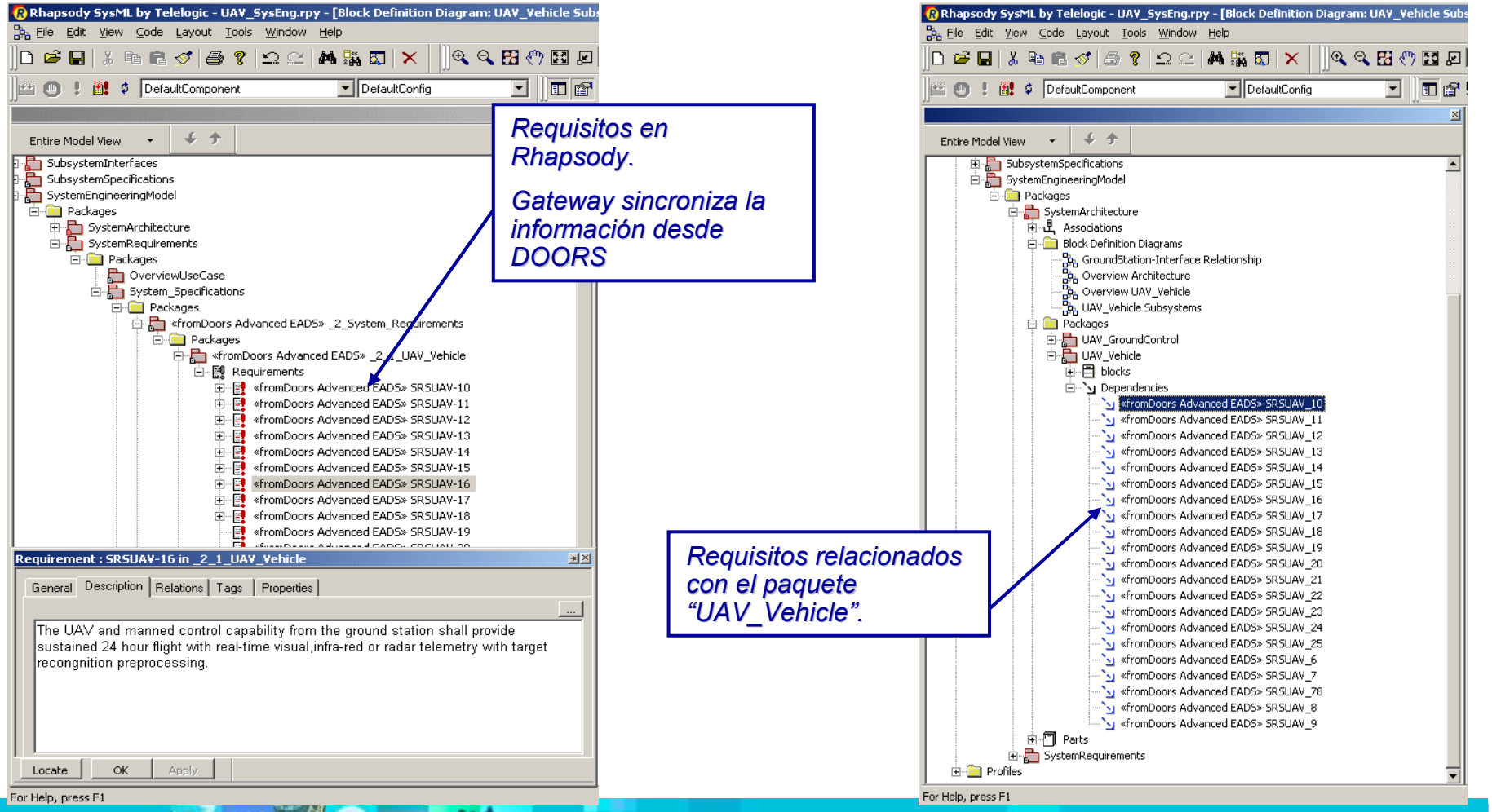
Crear relación desde el modelo a los requisitos

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…

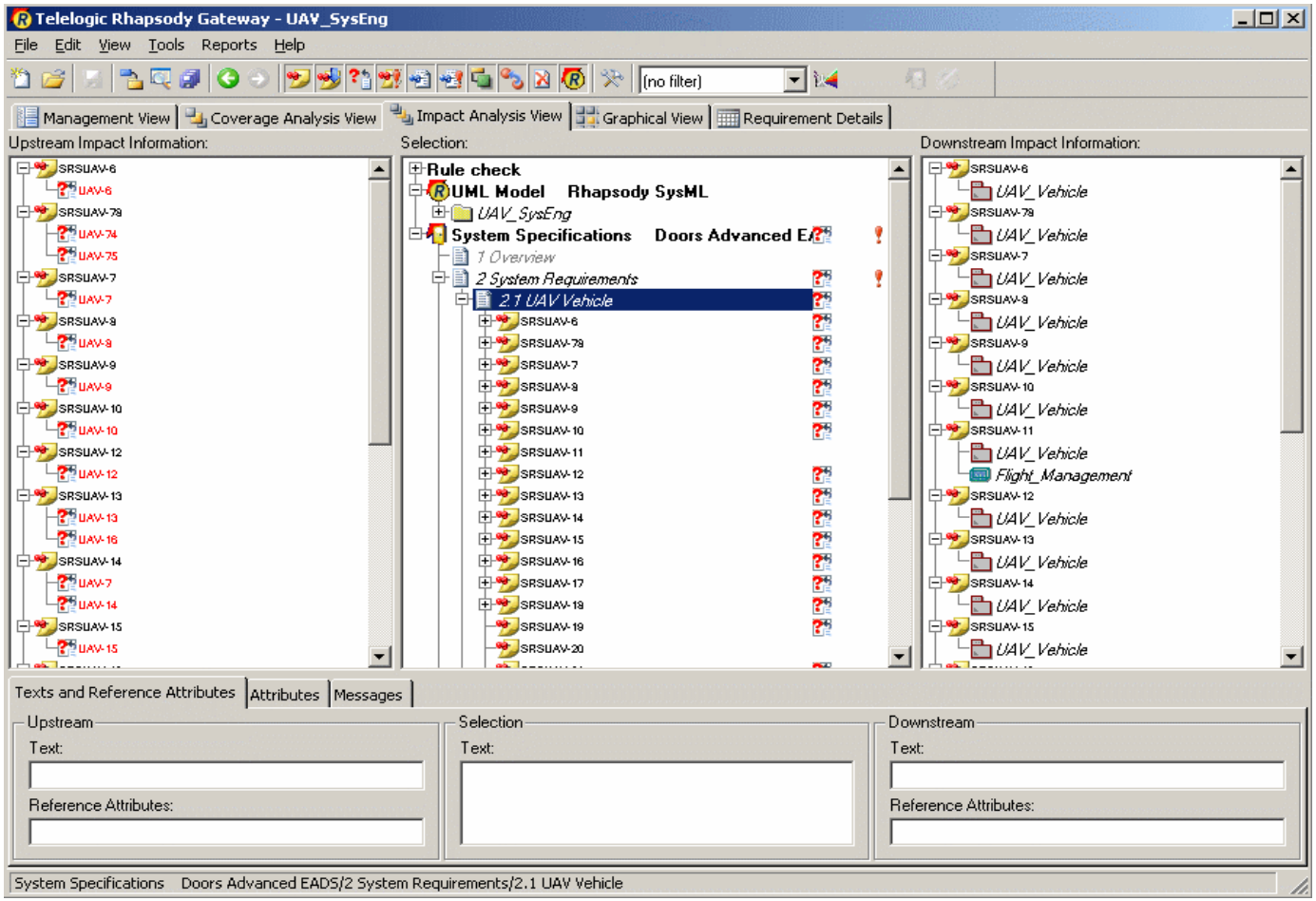
- **…o desde Rhapsody**



Trazabilidad de los requisitos de sistema al modelo

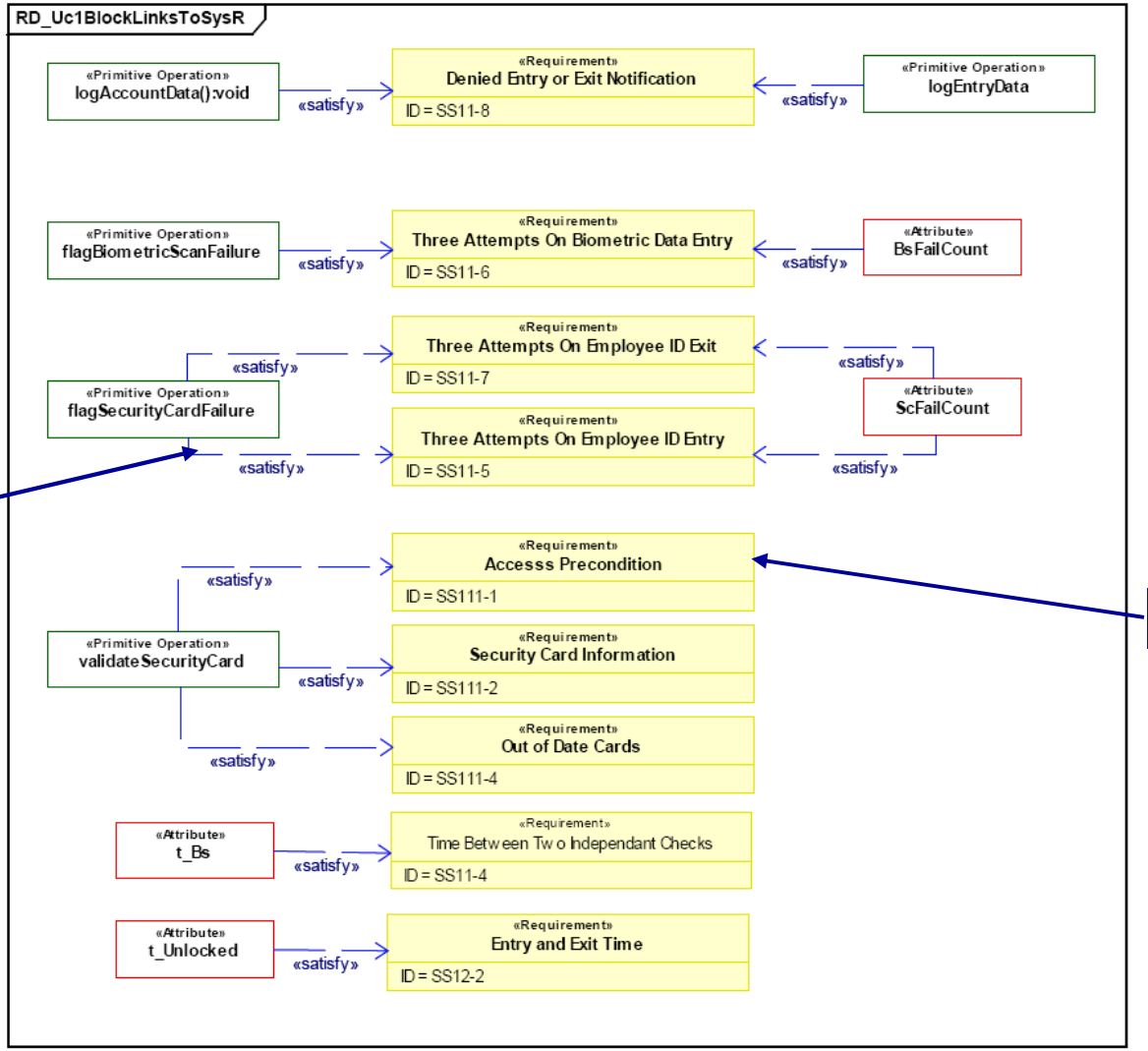
Asegurar que la arquitectura cumple los requisitos del sistema…

- **Impacto y cobertura desde Gateway**



Visualización de los requisitos y su trazabilidad

Diagramas de Requisitos



Trazabilidad, relación de satisfacción

Requisito

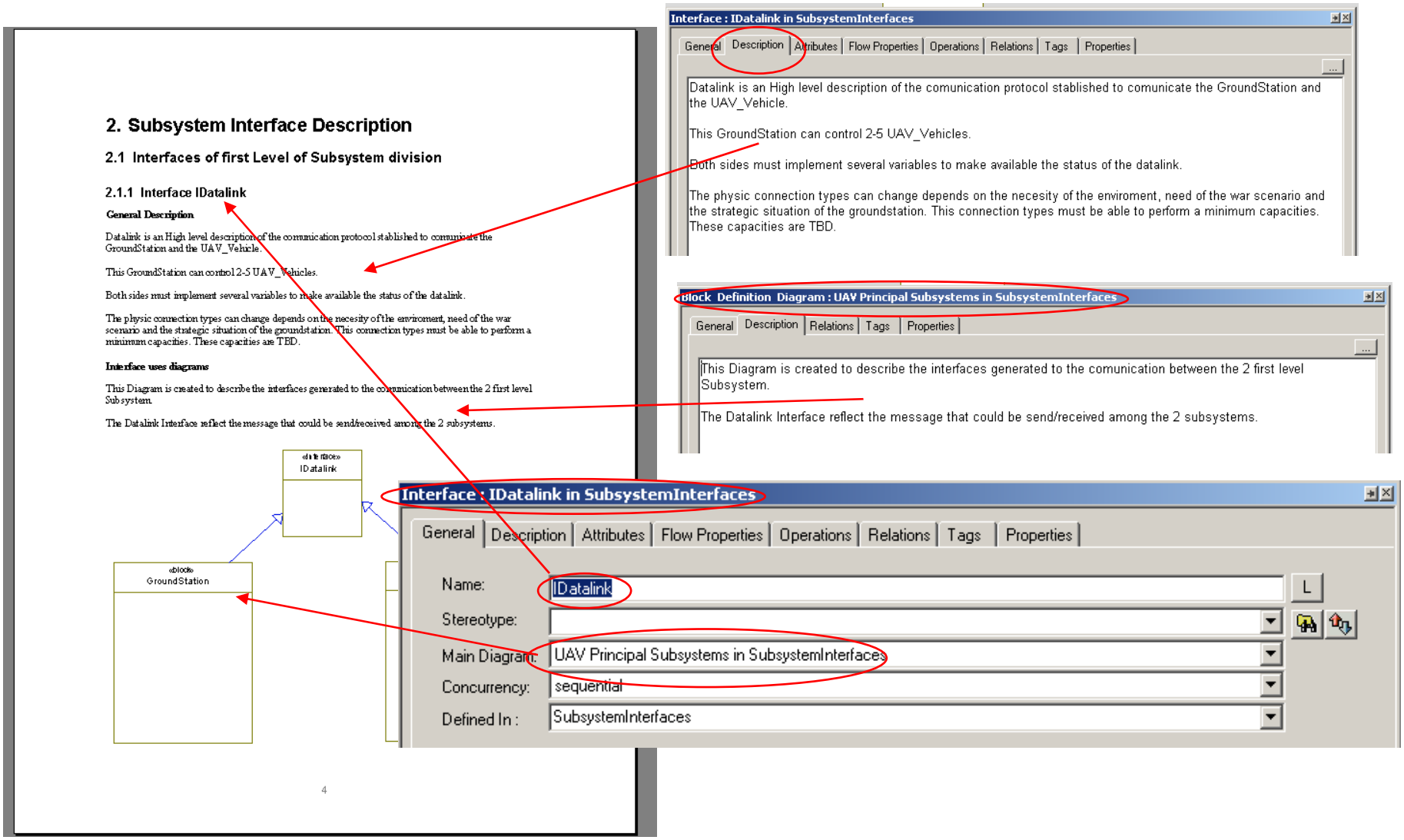
IBM Software Group

Generación automática de documentación: ICDs, …

Generación automática de documentación desde el modelo

- **Personalización de informes**
- **Diferentes formatos: HTML, Microsoft® Word®, PowerPoint®, Rich Text Format (RTF), etc. desde un modelo Rhapsody**
  - Crea enlaces para una navegación rápida del informe
- **Amplia variedad de plantillas predefinidas**
- **Fácil de utilizar**

Generación de documentación: diagramas, descripción textual..



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	9600..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..2e32	32 bits
OrgPort	int	0..2e32	32 bits
DestAddress	int	0..2e32	32 bits
DestPort	int	0..2e32	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

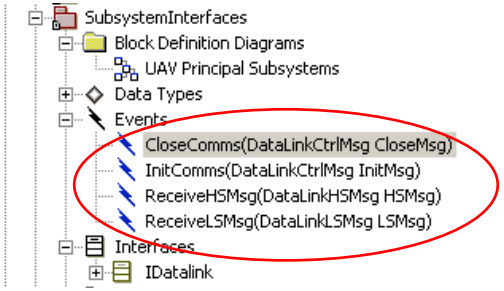
Name	Type	Range	Length
OrgAddress	int	0..2e32	32 bits
OrgPort	int	0..2e32	32 bits
DestAddress	int	0..2e32	32 bits
DestPort	int	0..2e32	32 bits
Type	LinkType	See LinkType	4 bits
DLOperation	DataLinkOpType	See DataLinkOpType	3 bits

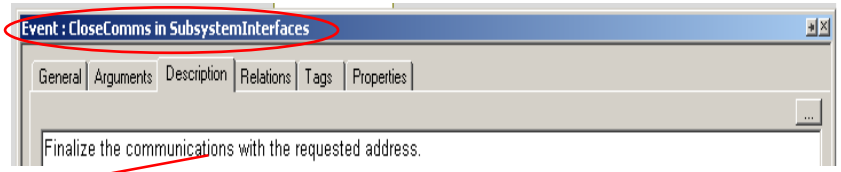
2.1.1.2.3 Message:ReceiveHSMsg

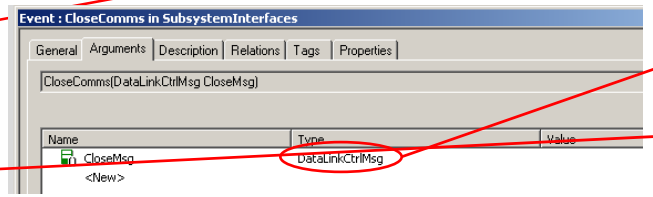
Description:
A Message of High Speed and High priority is sended from an origin address-port pair to a destination address-port pair. These pairs represent the 2 subsystems that is connected.
Fields in this message

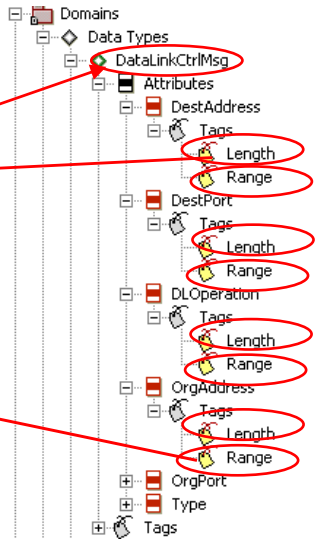
Name	Type	Range	Length
OrgAddress	int	0..2e32	32 bits
OrgPort	int	0..2e32	32 bits
DestAddress	int	0..2e32	32 bits
DestPort	int	0..2e32	32 bits

5









IBM Software Group

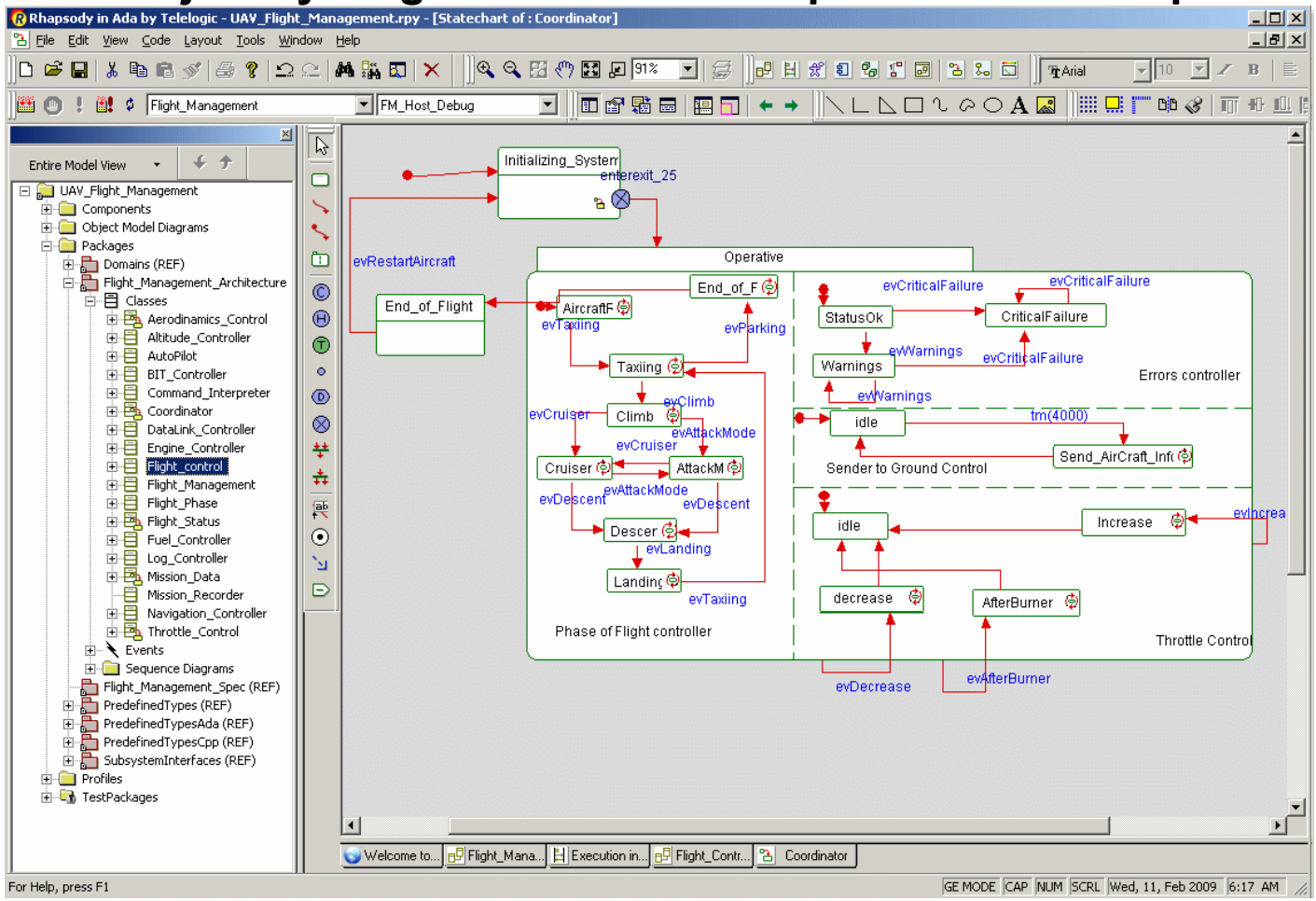
Diseño y desarrollo de software

Rational. software

Go to IBM

Diseño del software

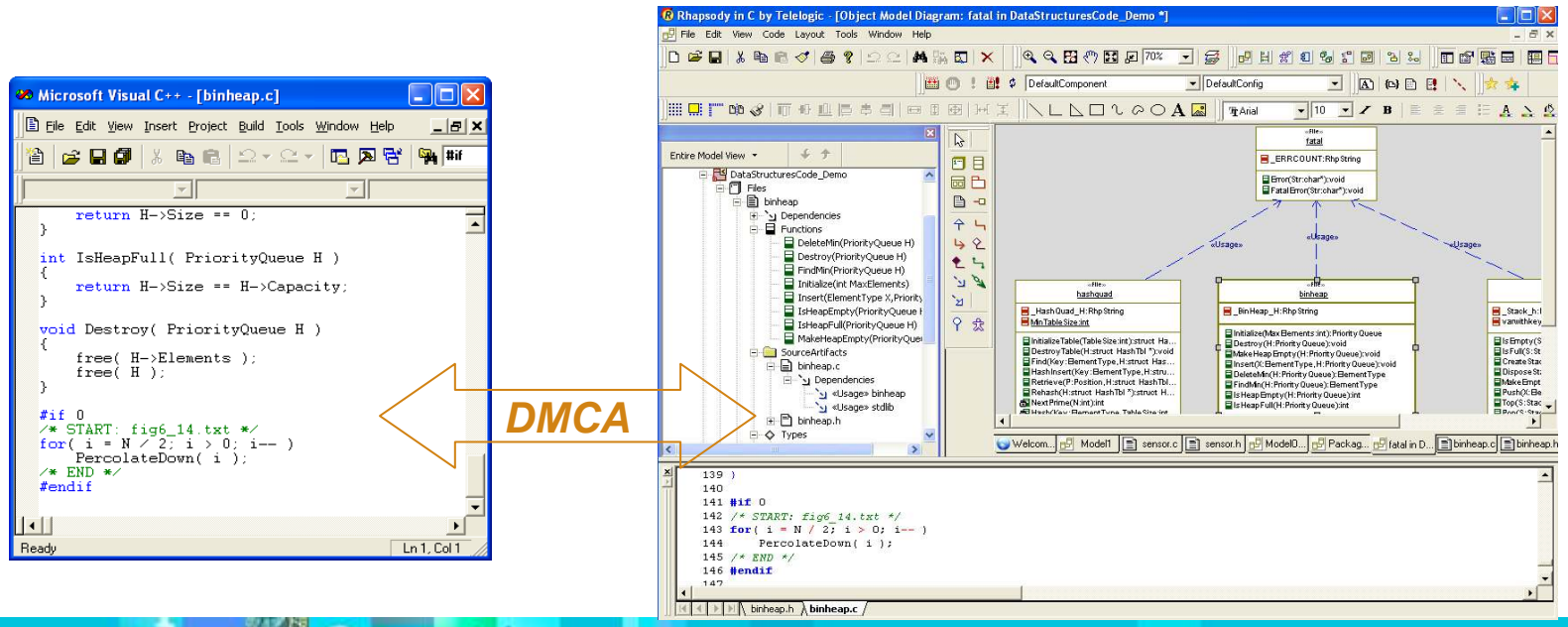
- **Diagrama de objetos y diagrama de estados que define el comportamiento**



Trabaje como usted quiera

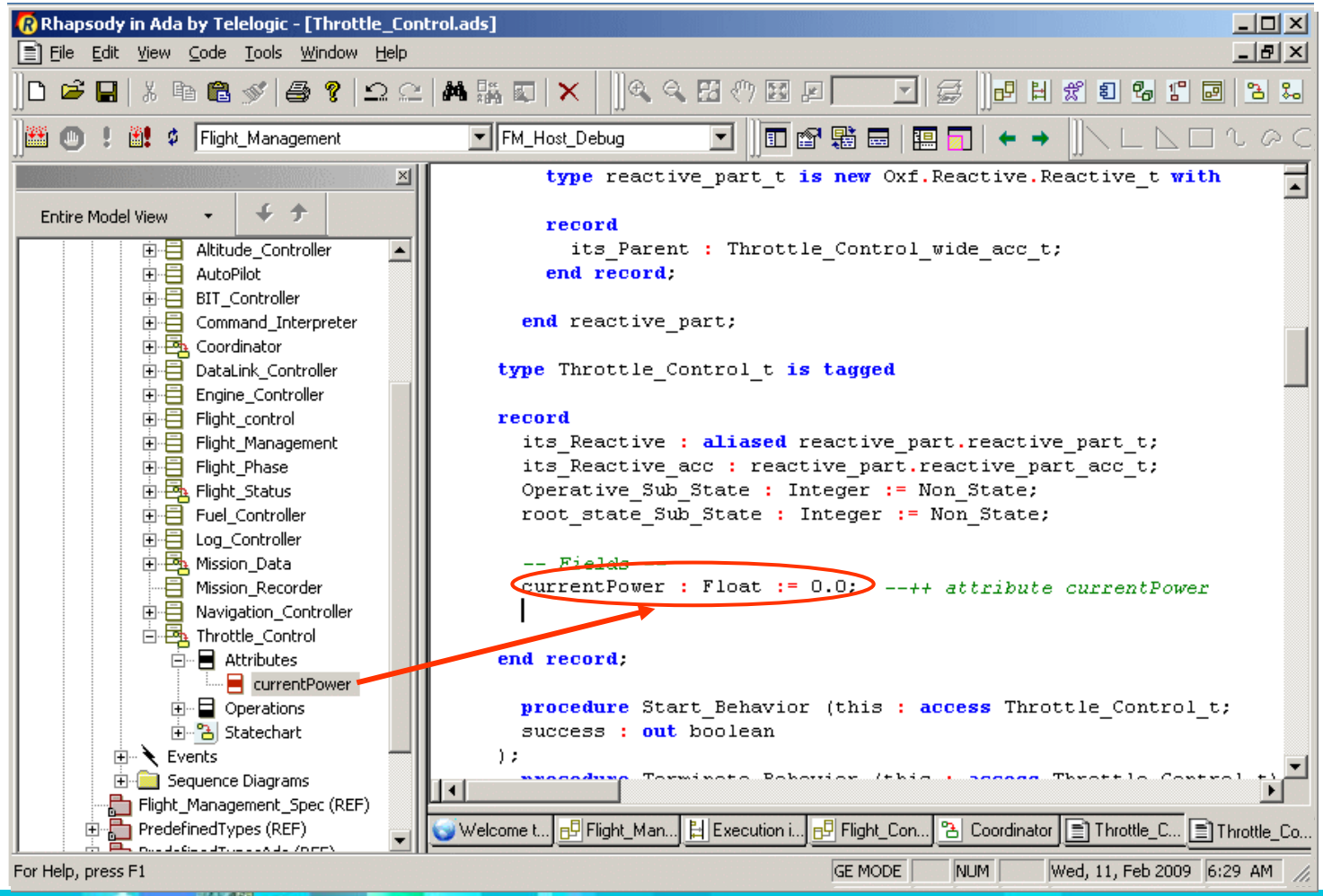
Dynamic Model Code Associativity - DMCA

- **A nivel del modelo o el código**
  - Reduce la curva de aprendizaje
  - Incremento de la productividad
- **Diseño y código se mantienen sincronizados: "dynamic model code associativity" (DMCA)**
  - Cambie el modelo y el código cambiará automáticamente
  - Cambie el código y el modelo cambiará automáticamente



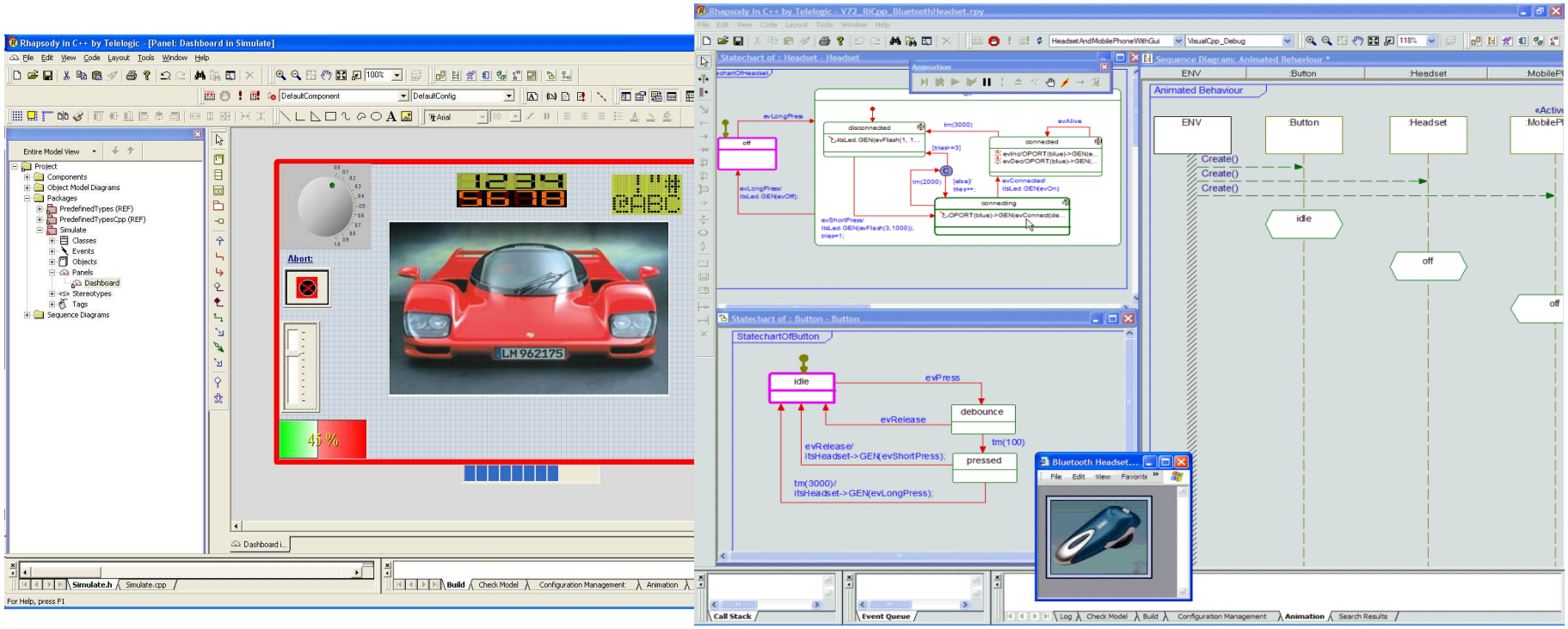
Si no quiere, no es necesario codificar…

- **Añadir detalle al modelo a través de formularios**



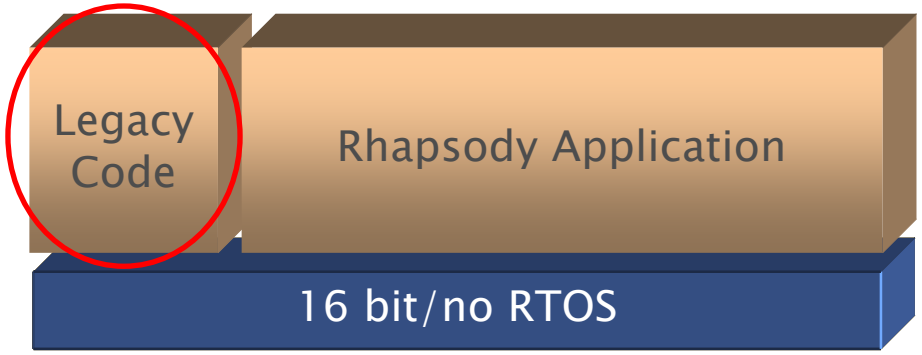
Paneles gráficos para la simulación y animación

- **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**



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 en el modelo o integrado (ingeniería inversa)**

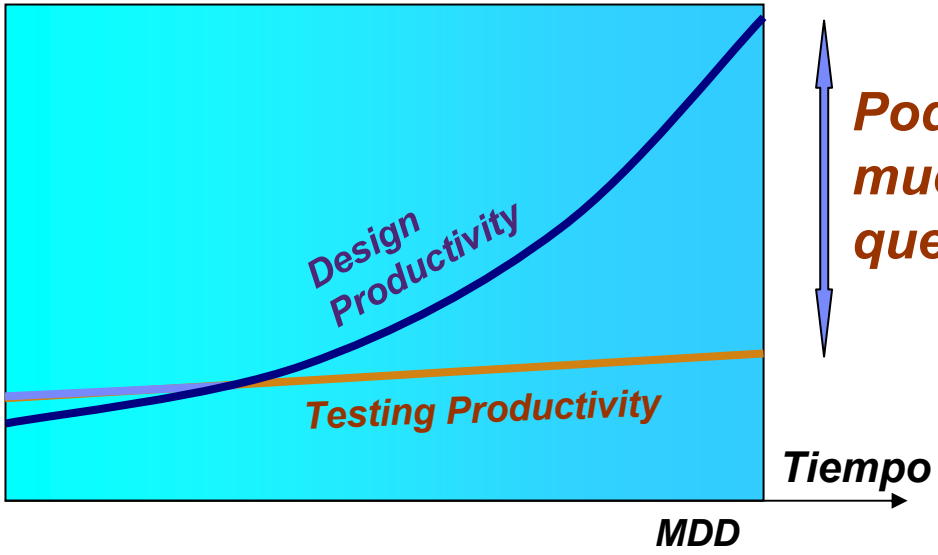


IBM Software Group

Pruebas dirigidas por modelos (Model Driven Testing – MDT)

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!!!**



Podemos diseñar mucho más rápido que probar!

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

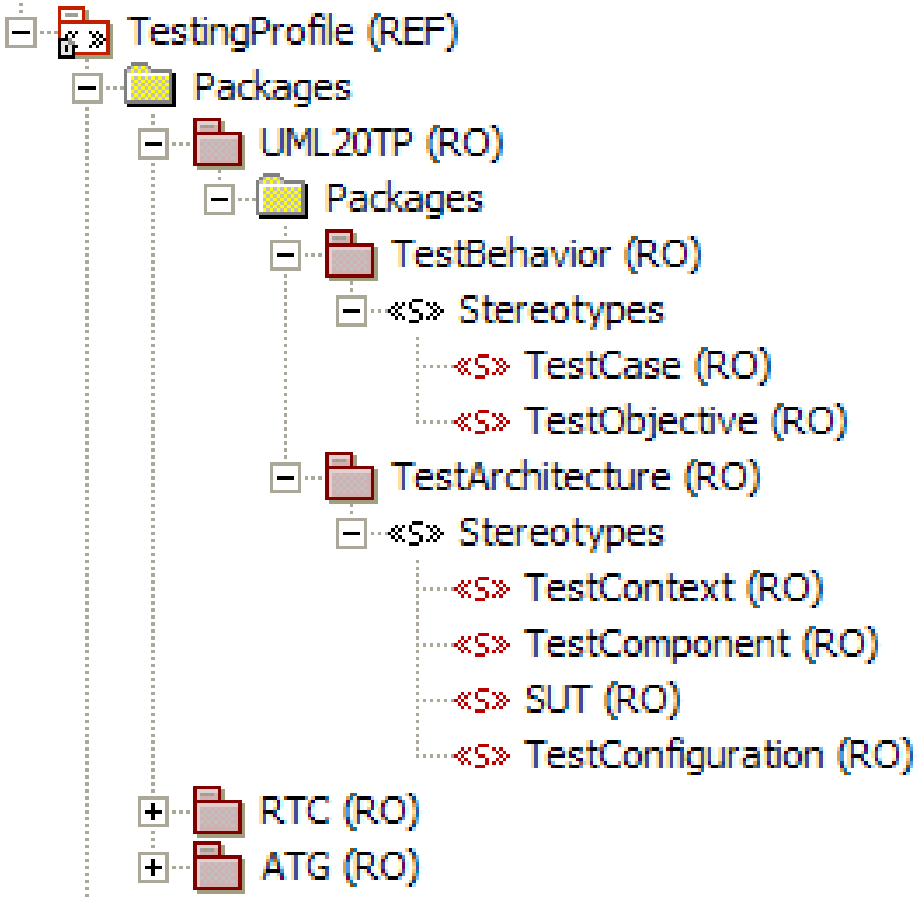
Model Driven Testing

Pruebas dirigidas por modelos (Model Driven Testing - MDT)

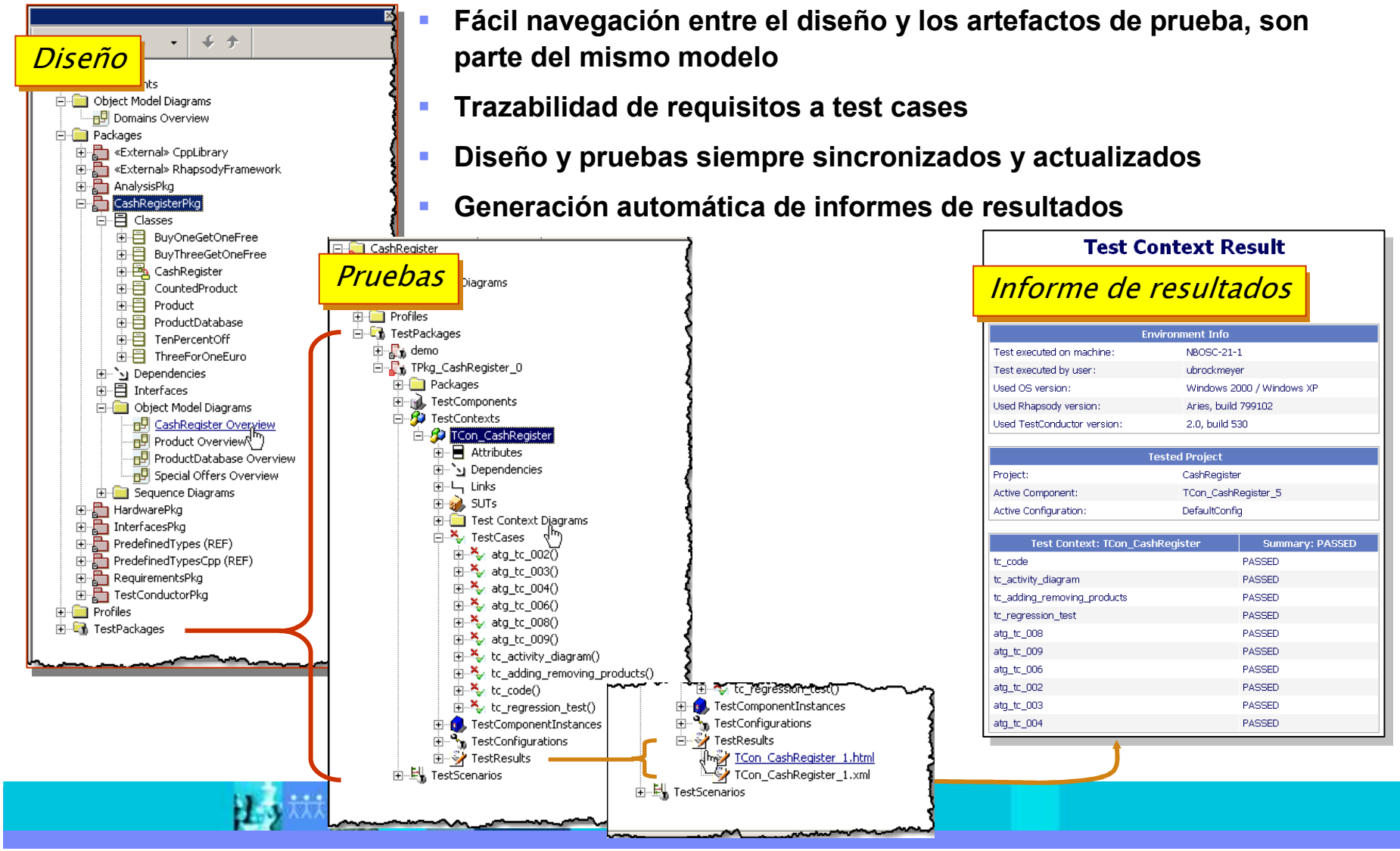
- Wikipedia, la enciclopedia libre en la World Wide Web (www), se refiere a *model driven testing* como "*software testing where test cases are derived in whole or in part from a model that describes some (if not all) aspects of the system under test (SUT)*"
- UML es el lenguaje para modelado, mientras 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



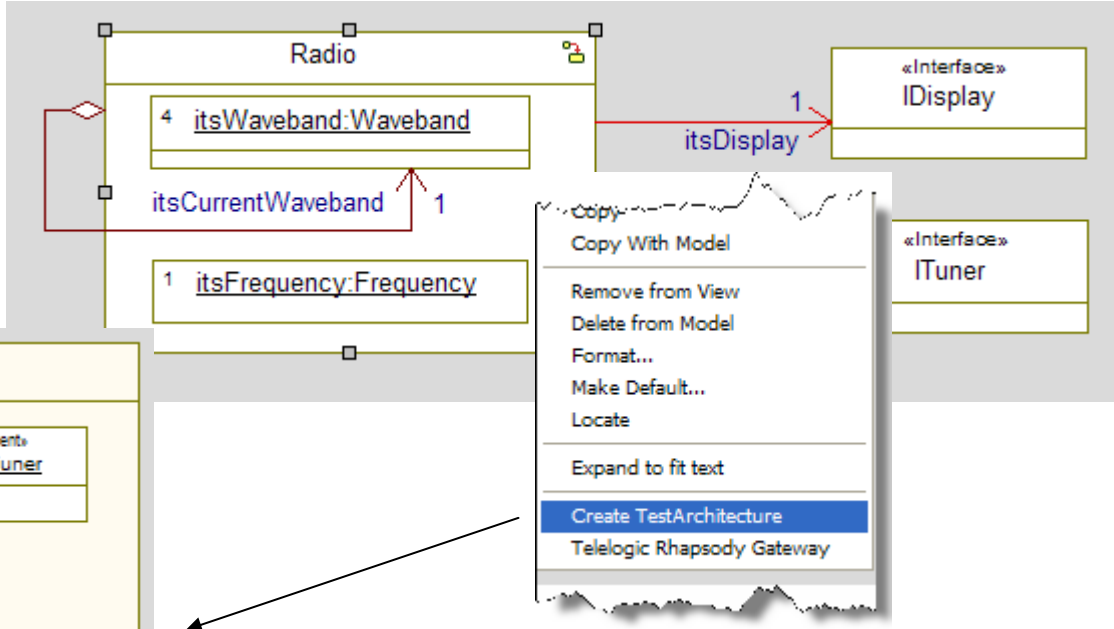
Procesos de diseño y pruebas completamente integrados

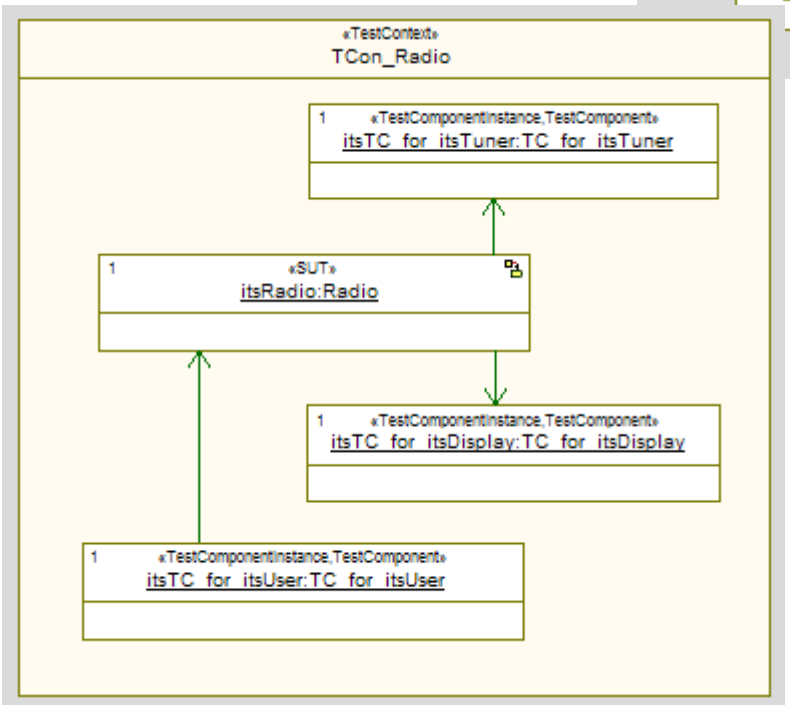


- **Fácil navegación entre el diseño y los artefactos de prueba, son parte del mismo modelo**
- **Trazabilidad de requisitos a test cases**
- **Diseño y pruebas siempre sincronizados y actualizados**
- **Generación automática de informes de resultados**

Proceso de pruebas: Crear arquitectura de pruebas

- **Crear Test Architecture**
  - Creación automática
  - Actualización automática, o
  - Actualización manual



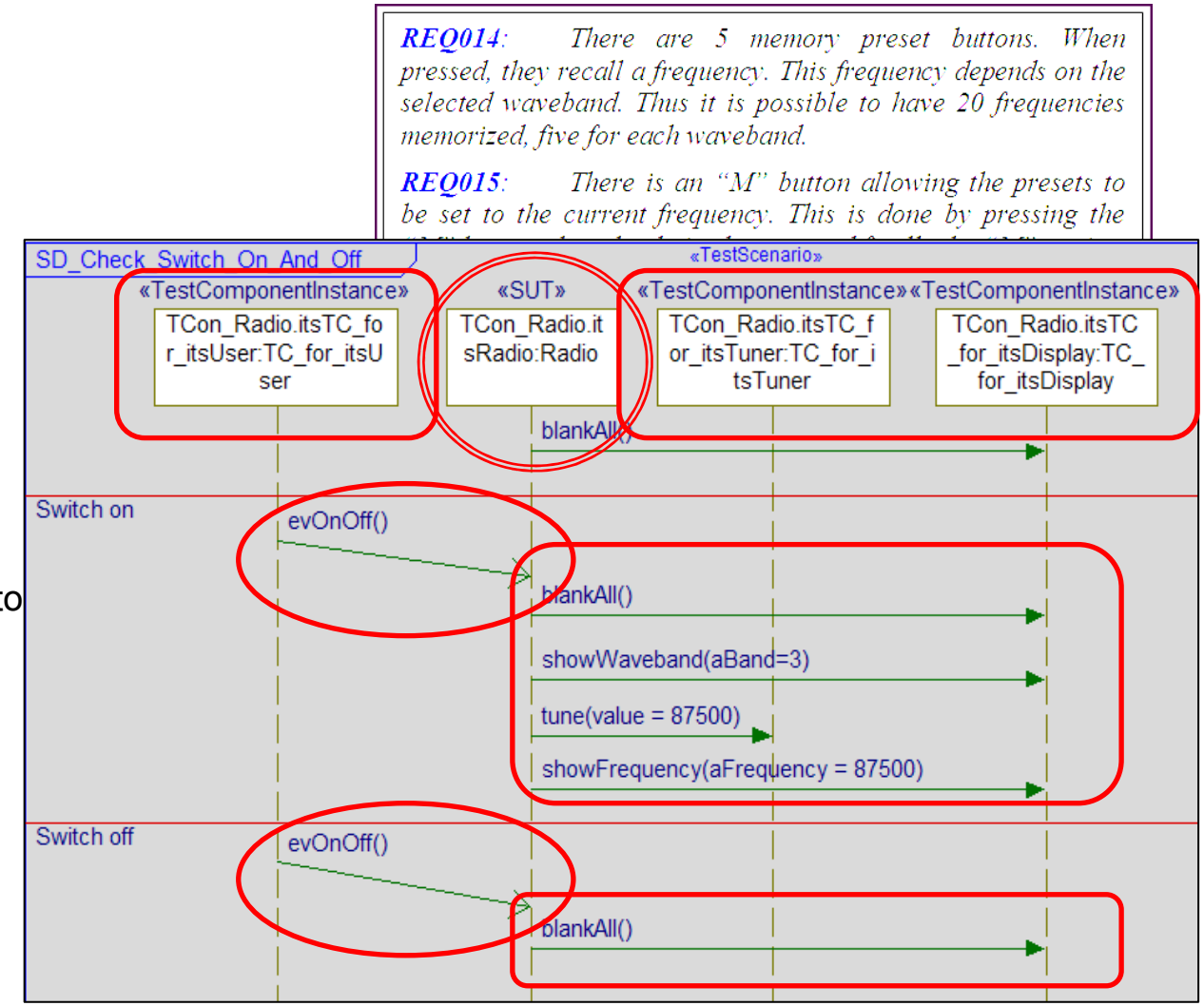


- **Graphical Test Architecture**
  - Es sólo arquitectura, todavía no hay casos de prueba

Proceso de pruebas: creación de casos de prueba

- Crear Test Architecture

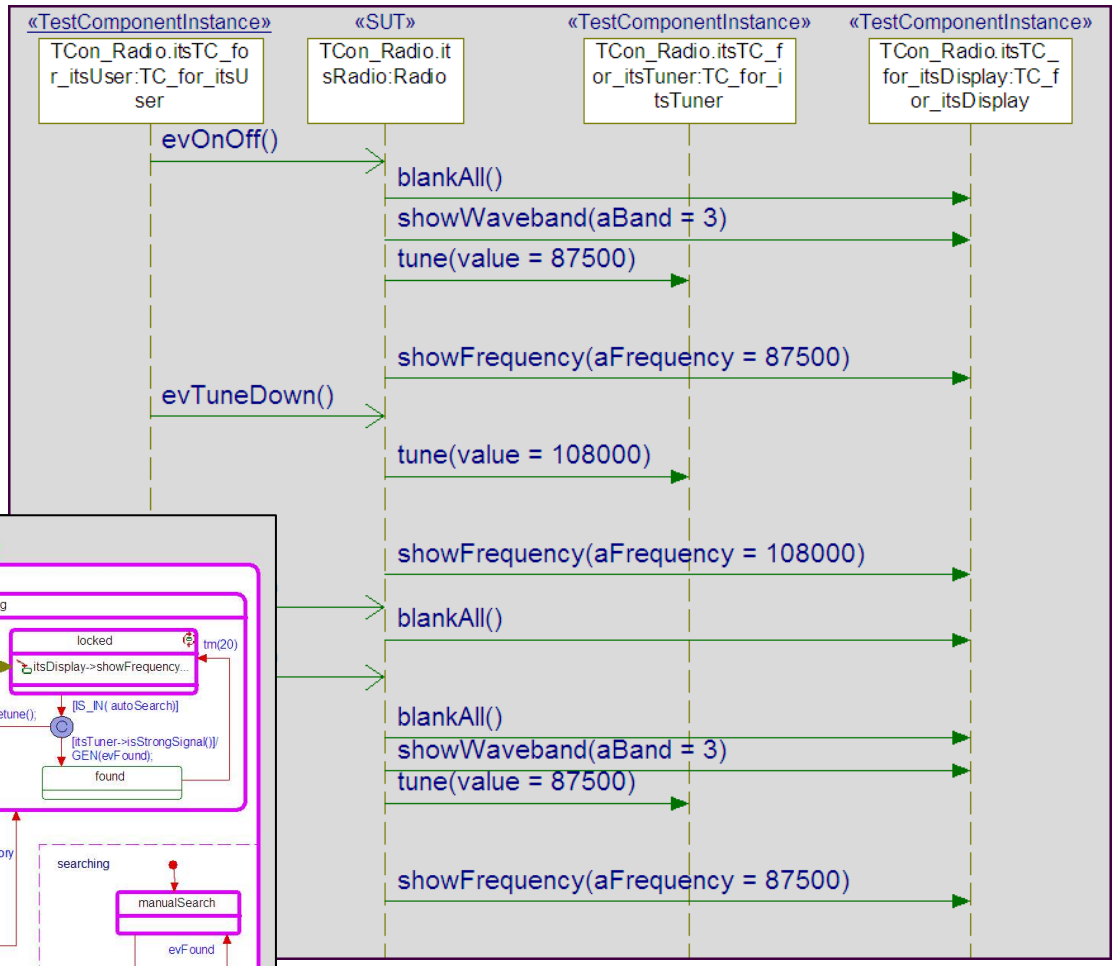
- **Casos de prueba como diag de secuencia**
  - Manualmente
    - En base a
      - Requisitos Textuales
      - Experiencia previa o en el dominio
    - Se define el comportamiento esperado del elemento a probar o SuT (System under Test)

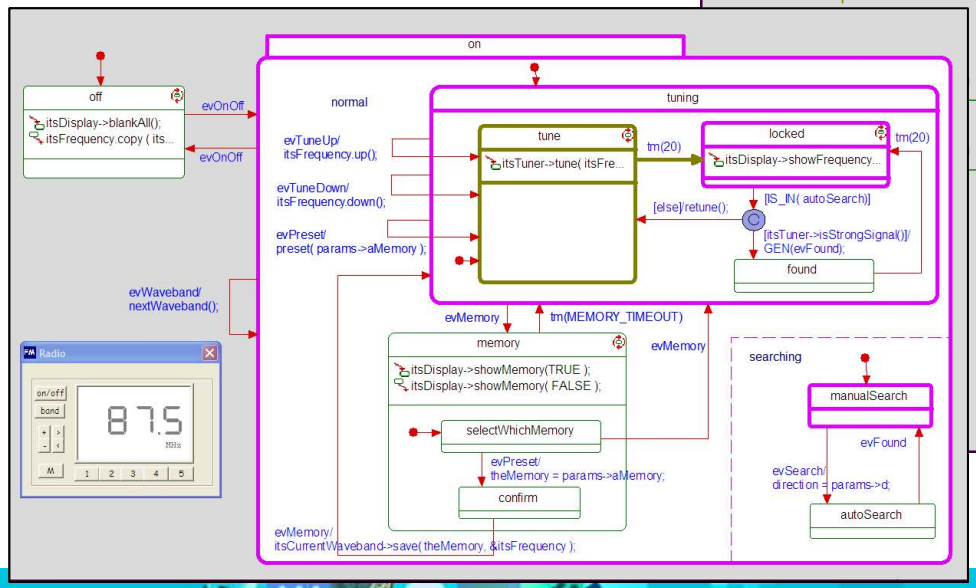


Proceso de pruebas: creación de casos de prueba

- **Crear Test Architecture**

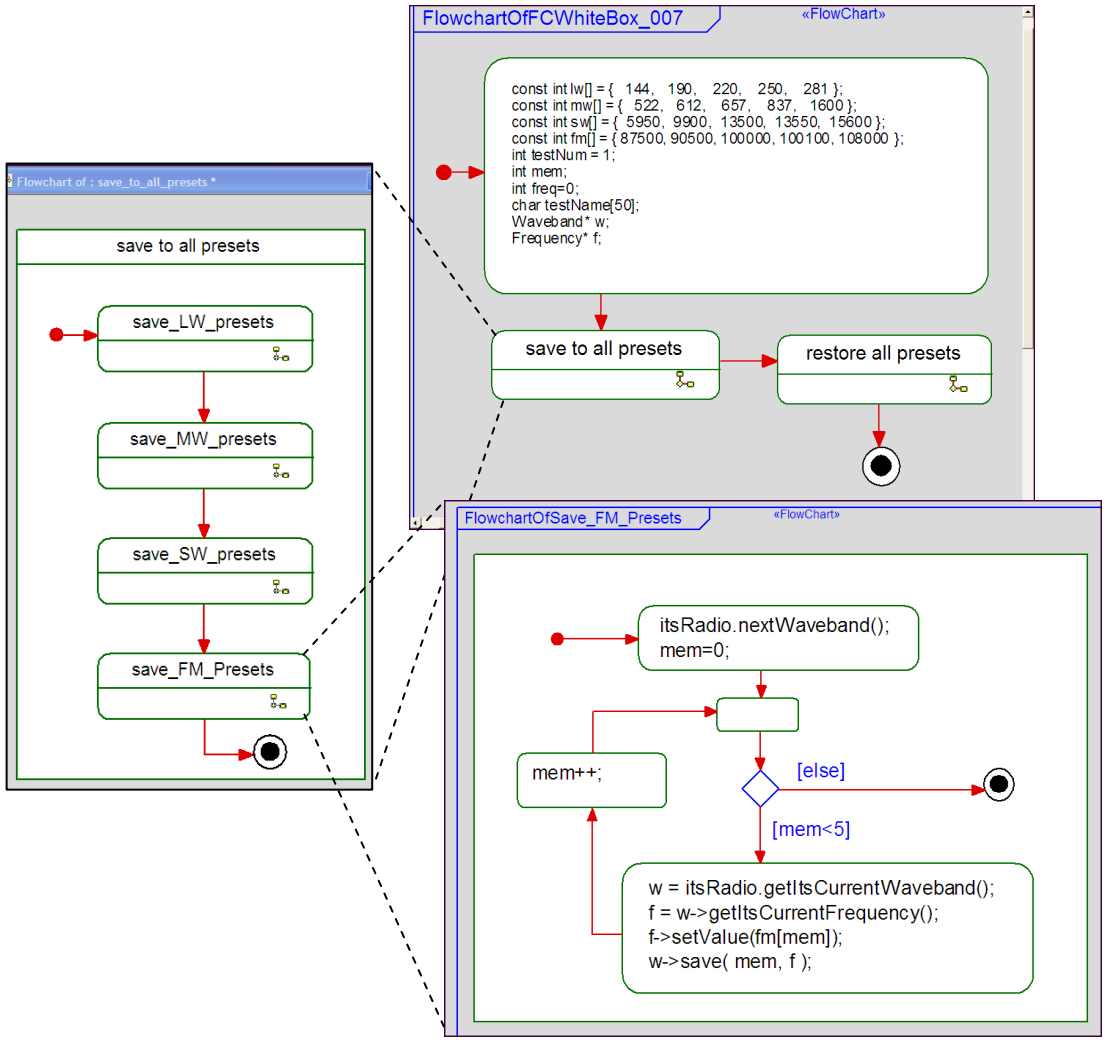
- **Casos de prueba como diag de secuencia**
  - Manualmente
  - Grabación de simulaciones (diagramas de secuencia animados)





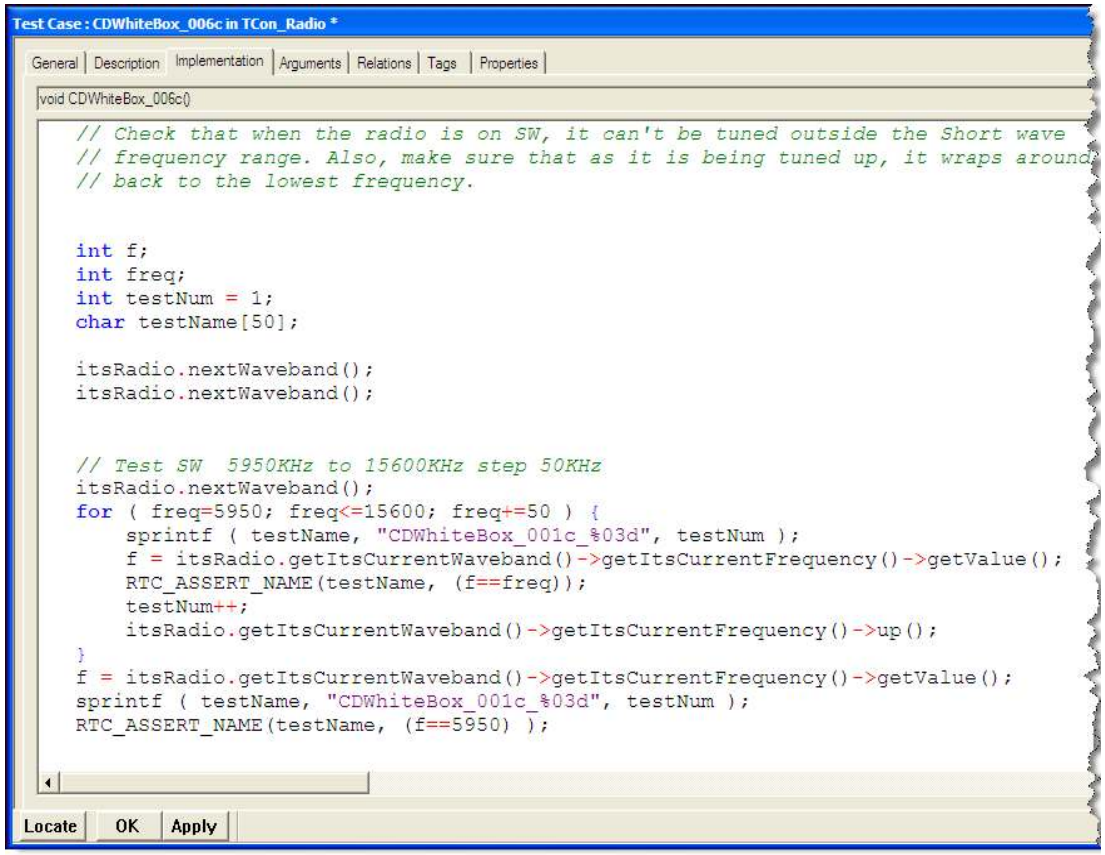
Proceso de pruebas: creación de casos de prueba

- **Crear a Test Architecture**
- **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



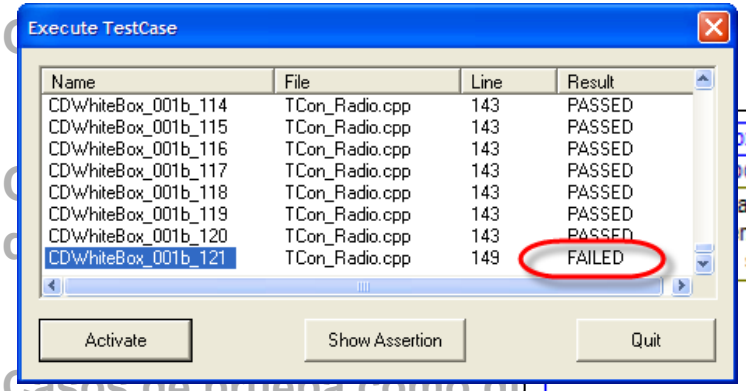
Proceso de pruebas: creacion de casos de prueba

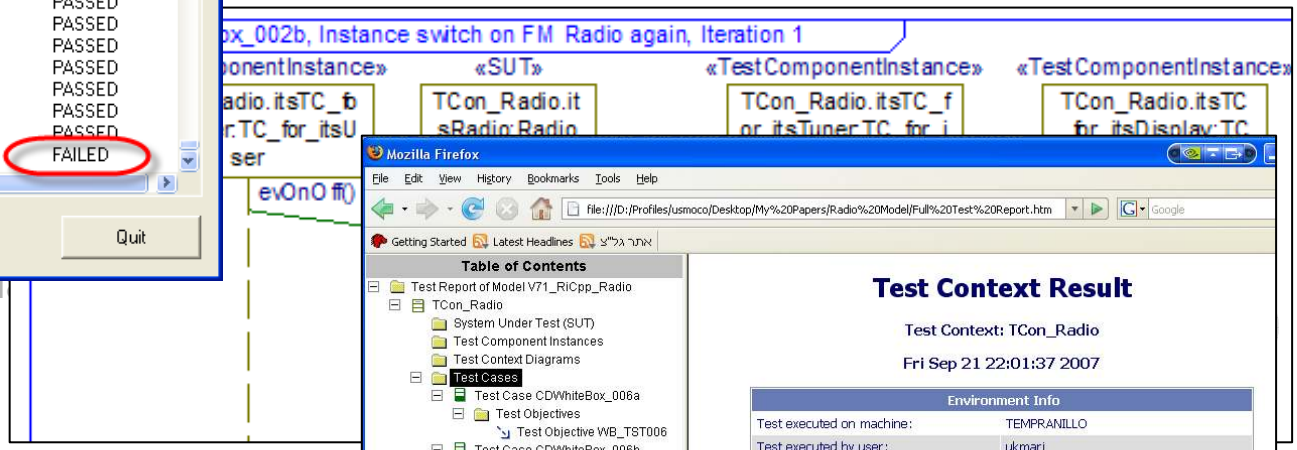
- **Crear Test Architecture**
- **Casos de prueba como diag de secuencia**
  - Manualmente
  - Grabación de simulaciones
- **Casos de prueba como diagramas de flujo**
  - “Programación gráfica”
  - Intuitivos, potentes y fáciles de usar
- **Casos de prueba como código**
  - El comportamiento del caso de prueba se puede introducir directamente codificando



Proceso de pruebas: ejecucion y resultados

- C
- C
- Casos de prueba como di
  de flujo
- Casos de prueba como código
- **Ejecución e informe de resultados**
  - Comportamientos inesperados se resaltan





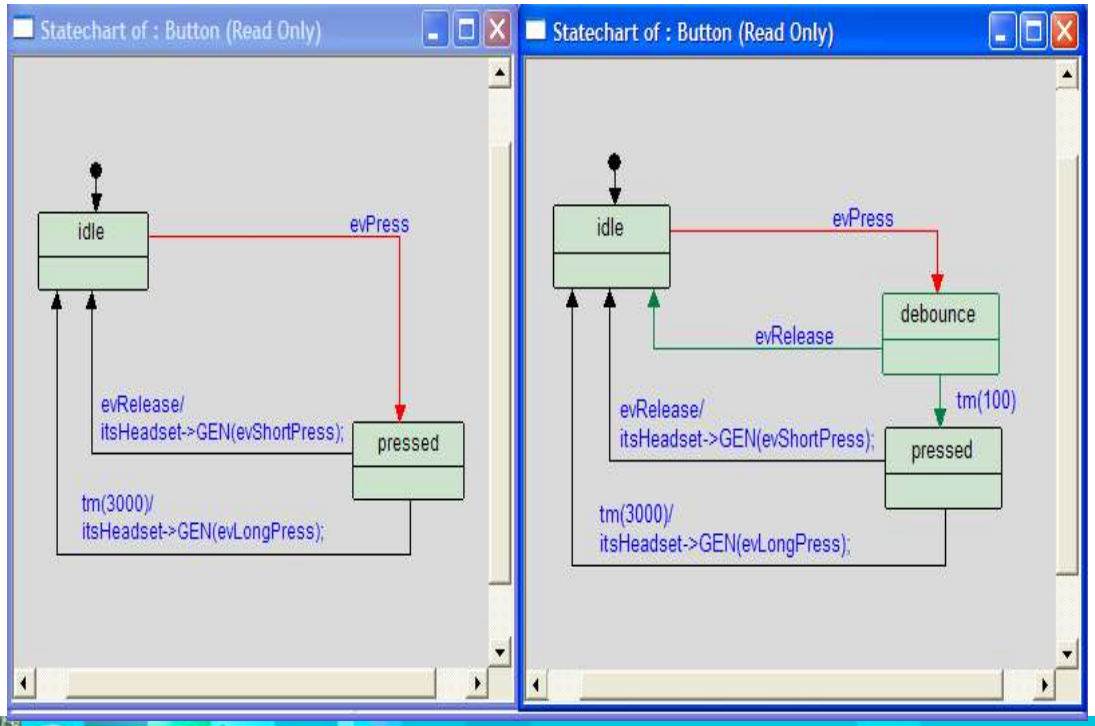
IBM Software Group

Colaboración, equipos distribuidos

Go to IBM

Colaboración

- **Soporte a equipos de desarrollos de cualquier tamaño**
- **Gestión del desarrollo en paralelo con comparador gráfico de diferencias entre modelos (también merge)**
- **Integración con herramientas de configuración de software como Rational ClearCase®, Rational Team Concert y Telelogic Synergy™**



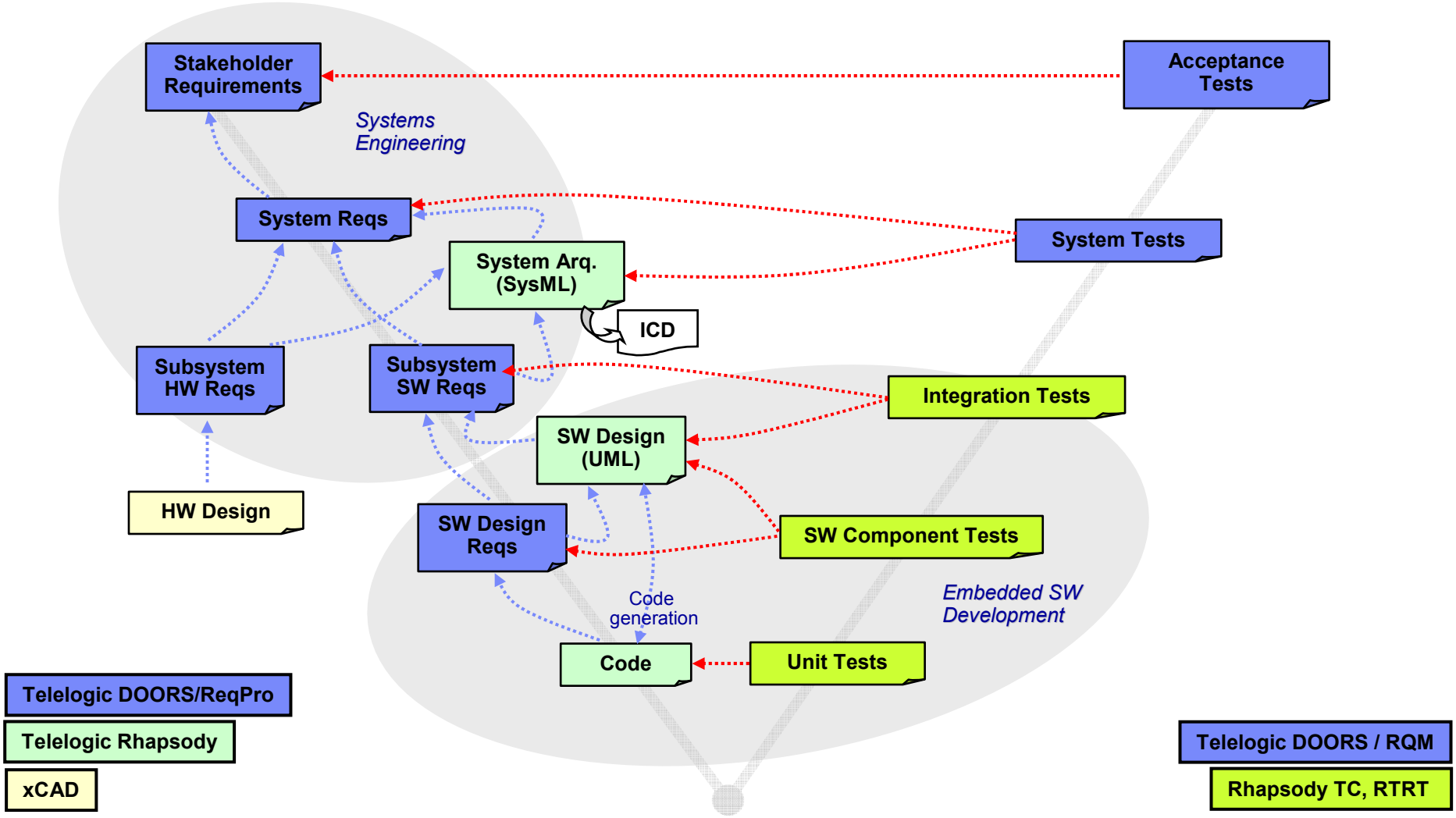
Resumen

Go to IBM

Resumen - IBM Telelogic Rhapsody ®

- **IBM Telelogic Rhapsody® es el entorno de diseño y desarrollo de sistemas que permite llevar a la práctica de forma productiva y eficiente los procesos de diseño y desarrollo de sistemas.**
- **Desarrollo dirigido por modelos (MDD). Mayor productividad desarrolladores**
- **Pruebas dirigidas por modelos (MDT). Mayor productividad ing. Pruebas**
- **Trazabilidad a requisitos, diseño cumple los requisitos.**
- **Verificación temprana mediante simulación, detección de errores antes de las pruebas, ahorro de costes.**
- **Mejora la comunicación entre disciplinas. Uso de lenguajes estándar SysML y UML**
- **Facilita la colaboración a equipos distribuidos**
- **Trabaja como quieras: en el modelo o en el código (DMCA). Disminuye el tiempo de aprendizaje**

Proceso, herramientas…



… y Telelogic Harmony

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

- **Harmony/SE**
  - Systems Engineering
- **Harmony/ESW**
  - Embedded Software
- **Harmony/IT**
  - IT Software

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