IBM Software Group

IBM Rational Software Architect/Modeler

Arquitectura y Diseño de Aplicaciones UML 2.0

Rational. software

Ana López-Mancisidor - IBM Software Development Tools

Ana.lopez@es.ibm.com

ON DEMAND BUSINESS

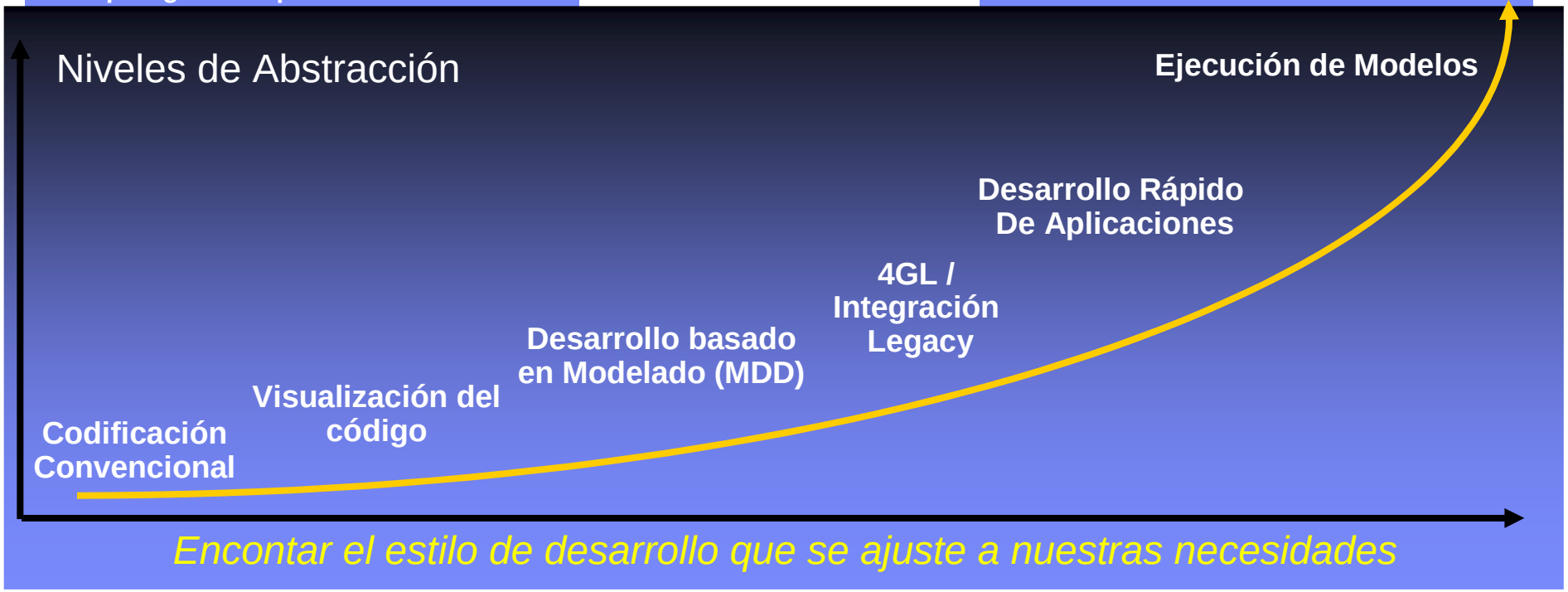
Evolución en el diseño y construcción de aplicaciones

Beneficios

- Mayor productividad
- Maximizar calidad, robustez, reusabilidad
- Maximizar el valor de código + modelos
- Soporte a diferentes tecnologías, y tipologías de aplicaciones

Capacidades

- Modelado, Codificación, pruebas, debug, despliegue
- Ingeniería directa e inversa del código y
- Integración legacy
- Desarrollo rápido de aplicaciones
- Ejecución de modelos



Encontar el estilo de desarrollo que se ajuste a nuestras necesidades

Evolución de las herramientas de modelado

- **Rational Rose – *Herramienta de modelado Visual UML***
  - Integrada con varios IDEs => Model-Driven Development (MDD)
  - Herramienta separada: buena para los architectos, no accesible para los desarrolladores

- **Rational XDE – *Entorno de Desarrollo Extendido***
  - Integrada con Eclipse 2.x, .NET => Model-Driven Development (MDD) más cercano a los desarrolladores
  - Perspectivas integradas para arquitectos y desarrolladores

- **Rational Software Architect/Modeler – *Integración de Análsis y Construcción***
  - Integrada con Eclipse 3.x
  - Soporte a UML 2.0
  - Integración y Trazabilidad en todo el ciclo de desarrollo software

Problemas a los que se enfrenta el Arquitecto

¿Cómo puedo detectar errores en la arquitectura?

¿Cómo puedo comunicar mi arquitectura al resto del equipo?

¿Cómo reforzar estándares de implementación?

Trazabilidad con el resto del desarrollo

Soluciones….

¿Cómo puedo detectar errores en la arquitectura?

Validaciones, Anti-patrones

¿Cómo puedo comunicar mi arquitectura al resto del equipo?

UML 2.0

Patrones, Transformaciones Reglas..

¿Cómo reforzar estándares de implementación?

Eclipse, Sincronización código..

Trazabilidad con el resto del desarrollo

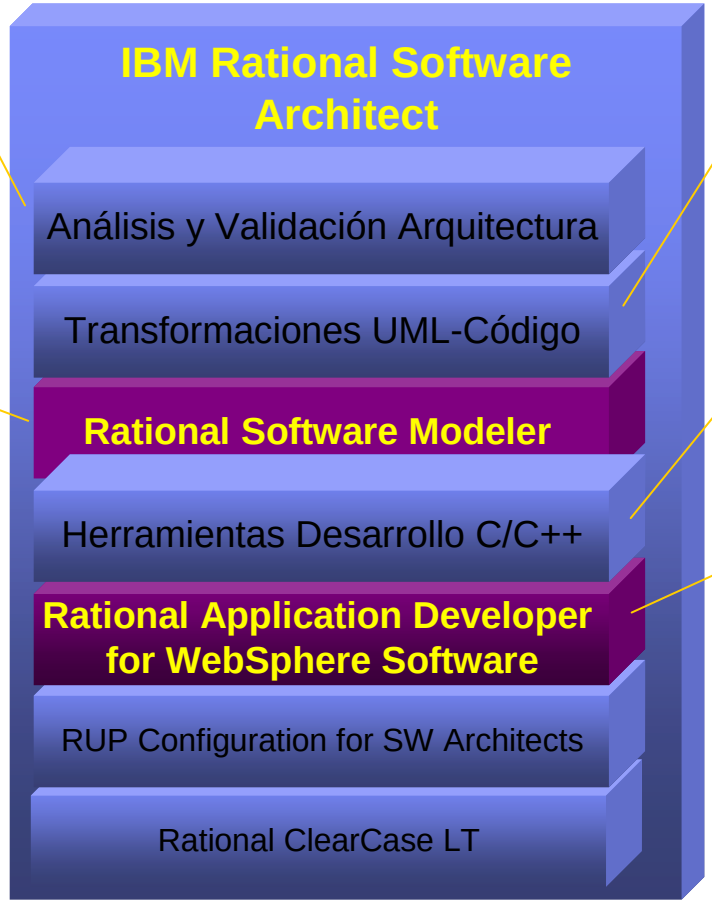
Funcionalidad de Rational Software Architect

"Análisis y Validación Arquitectura"

- Detección automática de patrones y anti-patrones
- Validación UML del modelo
- Reglas validación configurables

"Software Modeler"

- Soporte a UML 2.0
-
- Soporte OCL
- Patrones
- API estándar y pública
- Generación de informes
- Soporte a RAS (Reutilización Assets)



"Transformaciones"

- Generación y sincronización de código Java, C++
- Definición de reglas de transformación
- Visualización de la lógica del código

"Desarrollo C/C++"

- Editores C/C++
- Compilador y debugger
- Editores de código UML

"RAD v6"

- JSF, SDO, Struts
- Java GUI Editor
- Web diagram Editor
- Site designer
- Herramientas desarrollo Web Services
- Herramientas Diseño Base de Datos
- EGL
- Herramientas desarrollo EJBs
- Análisis Estático
- Análisis Dinámico
- Automatización Pruebas Componentes
- Herramientas desarrollo Portales

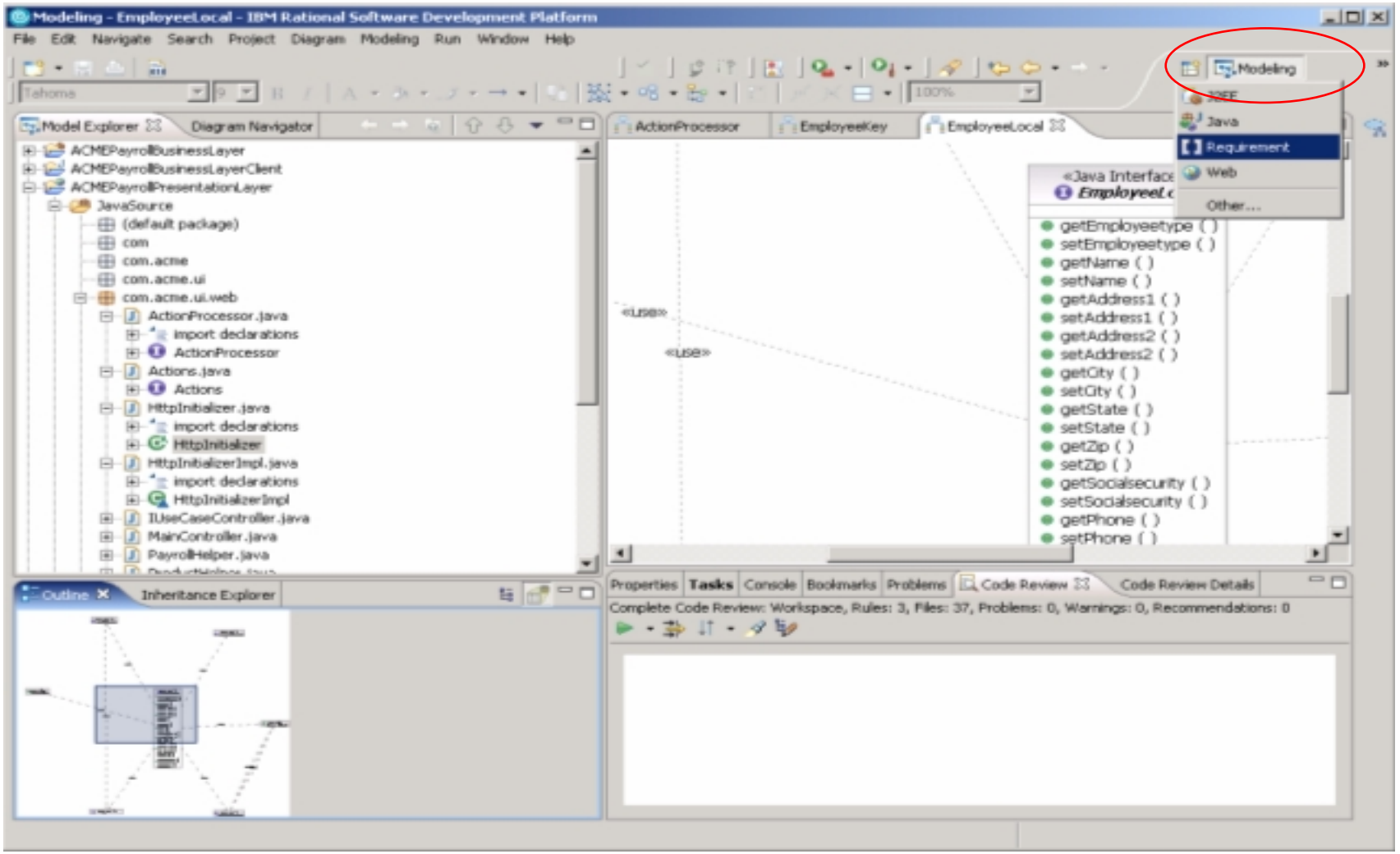
Demo Funcionalidad adicional de Rational Software Architect

- **Integración en Eclipse 3.0**
  - Nueva perspectiva de Modelado
- **Integración con otras herramientas del ciclo de vida:**
  - Gestión de Requisitos,
  - Construcción,
  - Pruebas,
  - Gestión de configuración,
  - Metodología ……
- **Soporte a UML 2.0**
- **Reutilización y Automatización con Patrones y Transformaciones**
  - Creación y aplicación de patrones
  - Generación y sincronización de código Java, C++
  - Visualización de la lógica del código
- **Validación Arquitectura**
  - Análisis y validación del modelo
  - Detección de anti-patrones y dependencias cíclicas

Demo Funcionalidad adicional de Rational Software Architect

- **Integración en Eclipse 3.0**
  - Nueva perspectiva de Modelado
- **Integración con otras herramientas del ciclo de vida:**
  - Gestión de Requisitos,
  - Construcción,
  - Pruebas,
  - Gestión de configuración,
  - Metodología
- **Soporte a UML 2.0**
- **Reutilización y Automatización con Patrones y Transformaciones**
  - Creación y aplicación de patrones
  - Generación y sincronización de código Java, C++
  - Visualización de la lógica del código
- **Validación Arquitectura**
  - Análisis y validación del modelo
  - Detección de anti-patrones y dependencias cíclicas

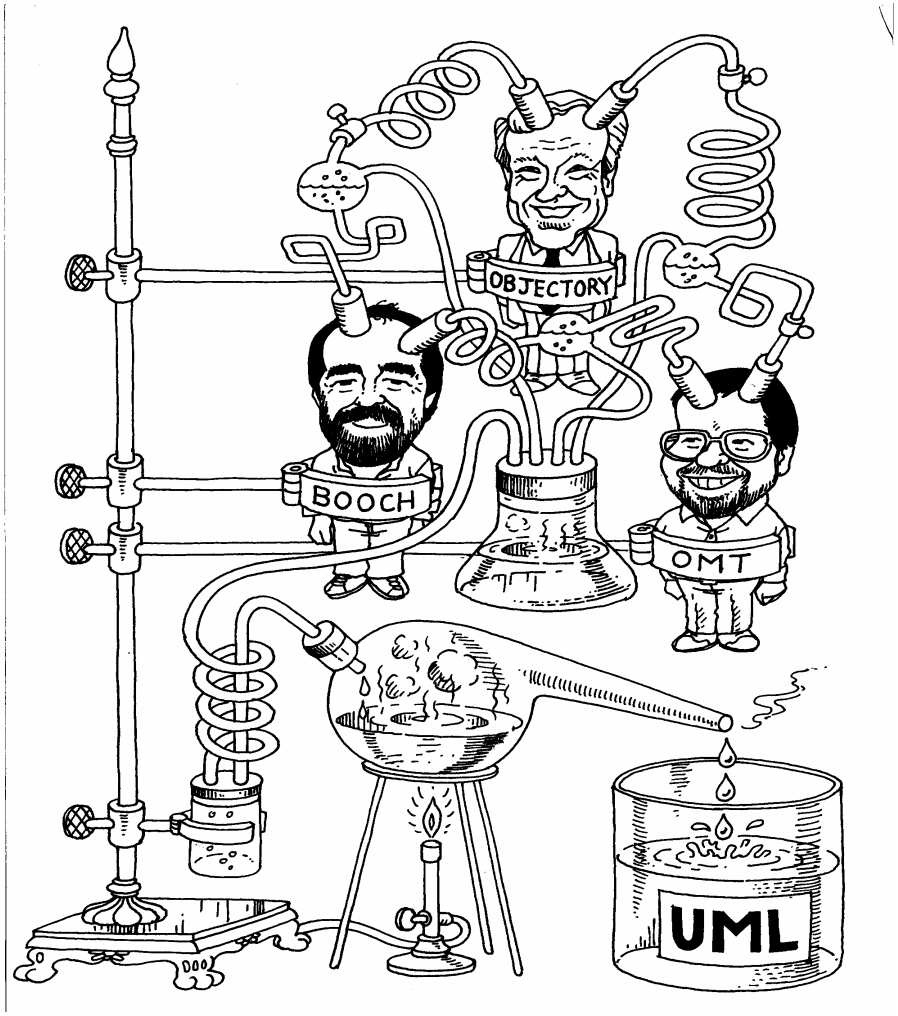
Integración en Eclipse 3.0 y Trazabilidad



Demo Funcionalidad adicional de Rational Software Architect

- **Integración en Eclipse 3.0**
  - Nueva perspectiva de Modelado
- **Integración con otras herramientas del ciclo de vida:**
  - Gestión de Requisitos,
  - Construcción,
  - Pruebas,
  - Gestión de configuración,
  - Metodología
- **Soporte a UML 2.0**
- **Reutilización y Automatización con Patrones y Transformaciones**
  - Creación y aplicación de patrones
  - Generación y sincronización de código Java, C++
  - Visualización de la lógica del código
- **Validación Arquitectura**
  - Análisis y validación del modelo
  - Detección de anti-patrones y dependencias cíclicas

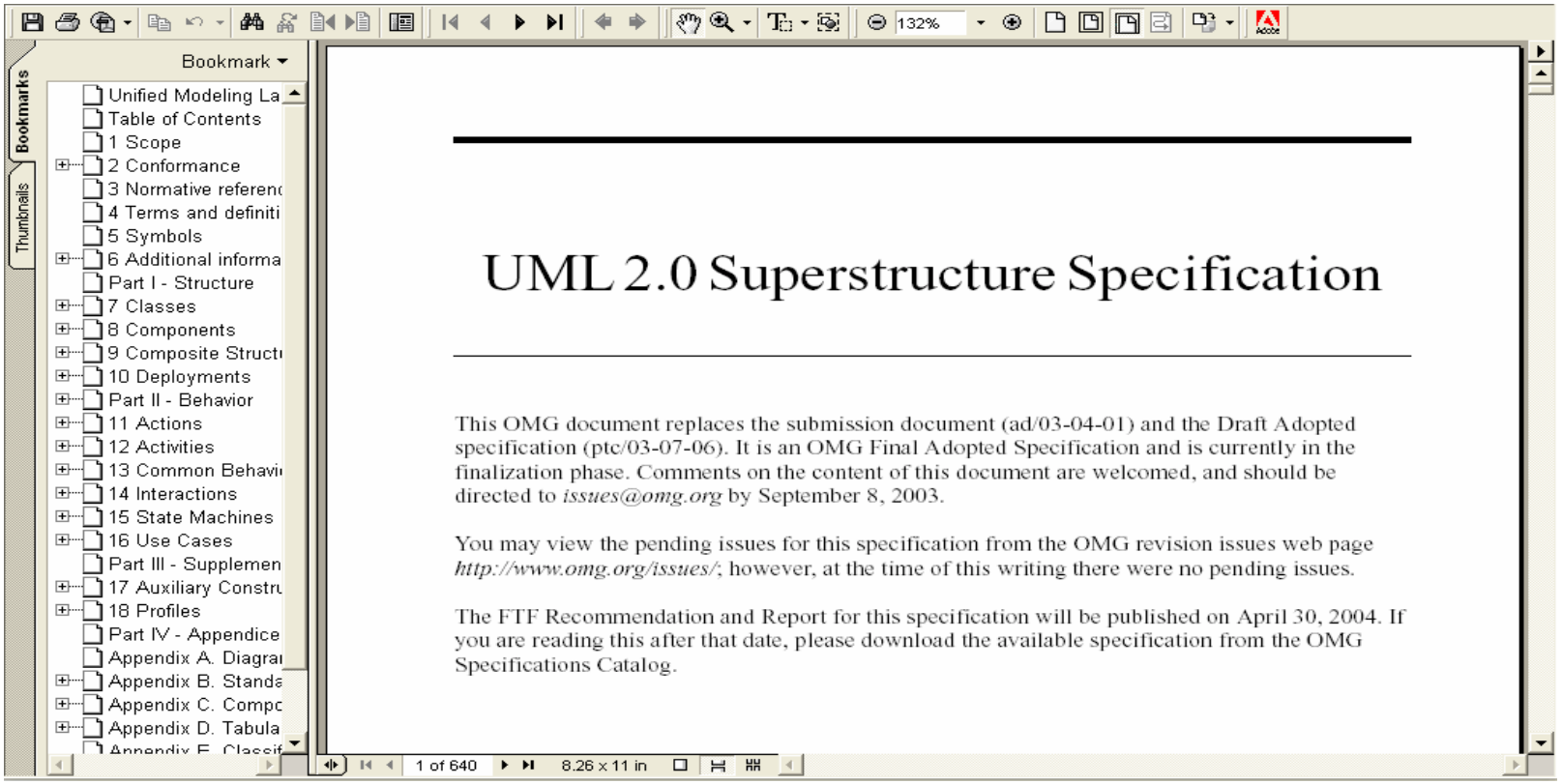
UML: El Lenguaje para el Modelado



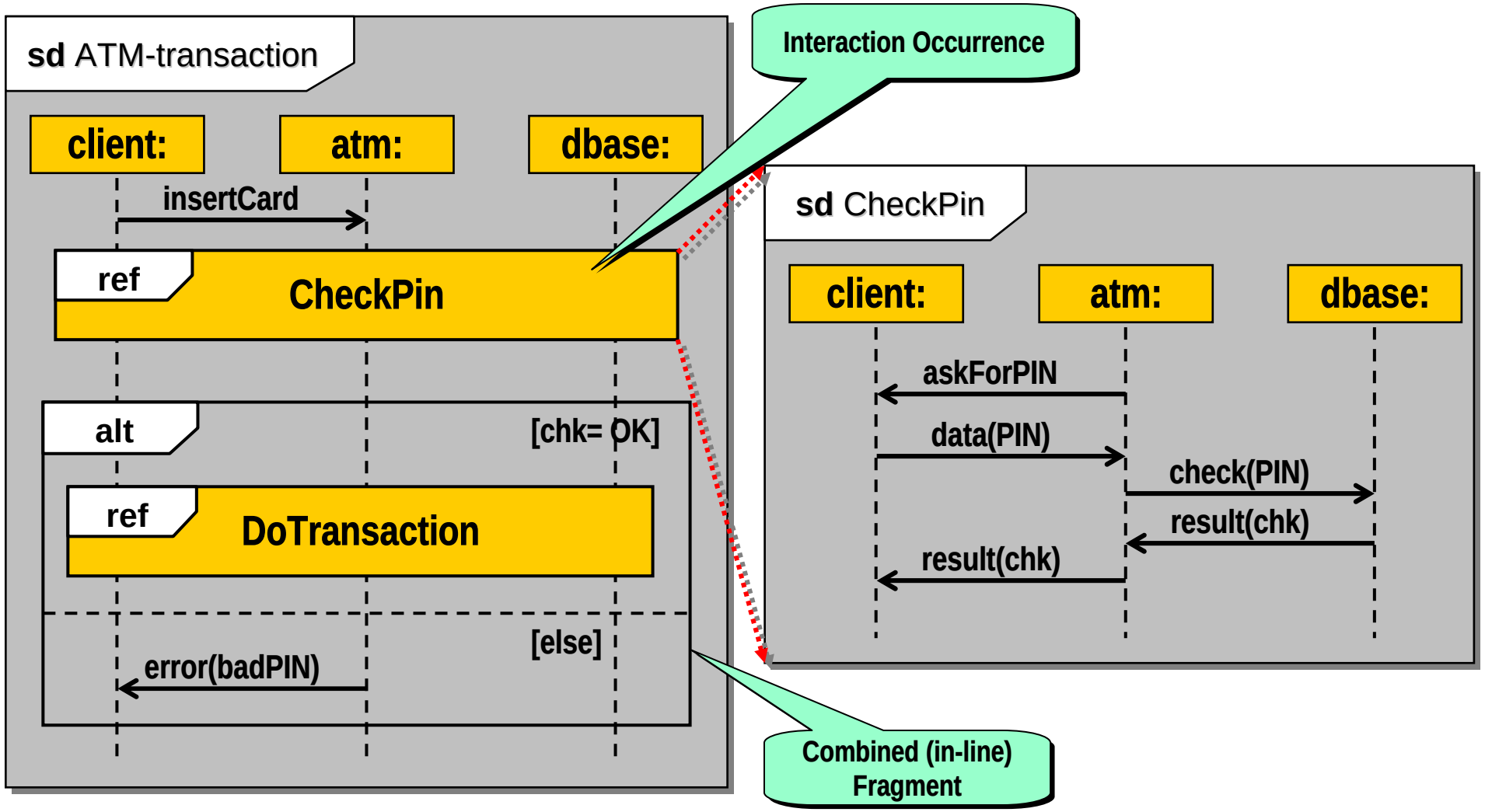
UML es el lenguaje estándar para la visualización, especificación, construcción y documentación de sistemas software.

UML 2.0 Specification

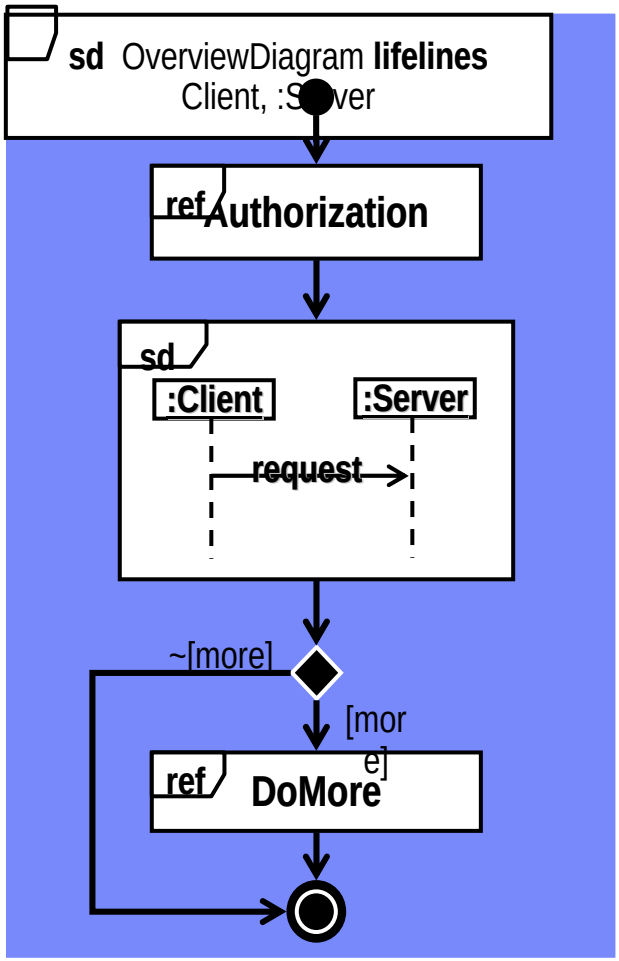
- Se puede descargar de http://www.omg.org/cgi-bin/doc?ptc/2003-08-02



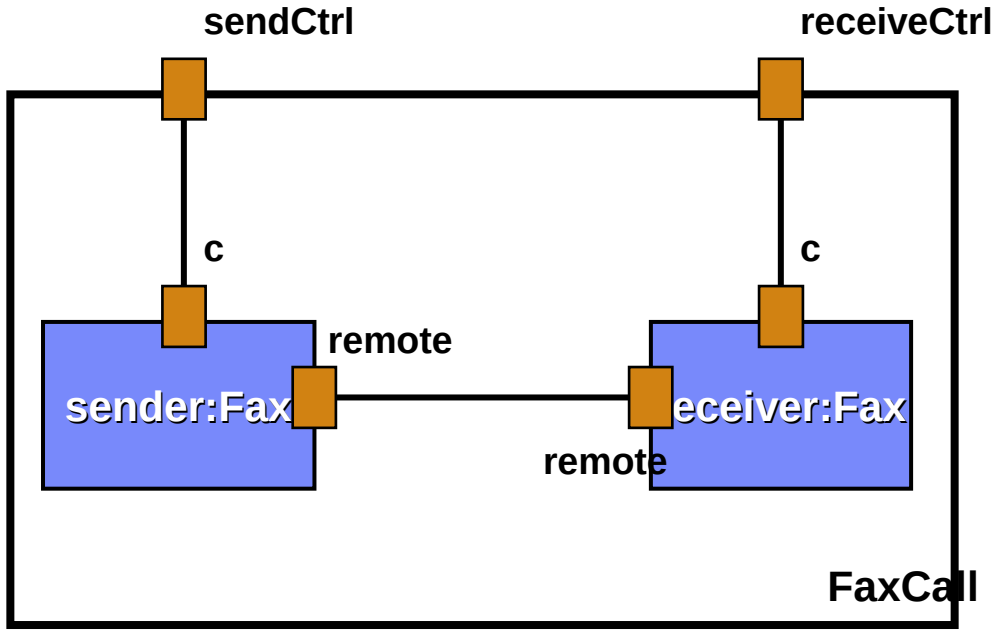
Nuevas notaciones en diagramas de secuencia



Nuevos diagramas



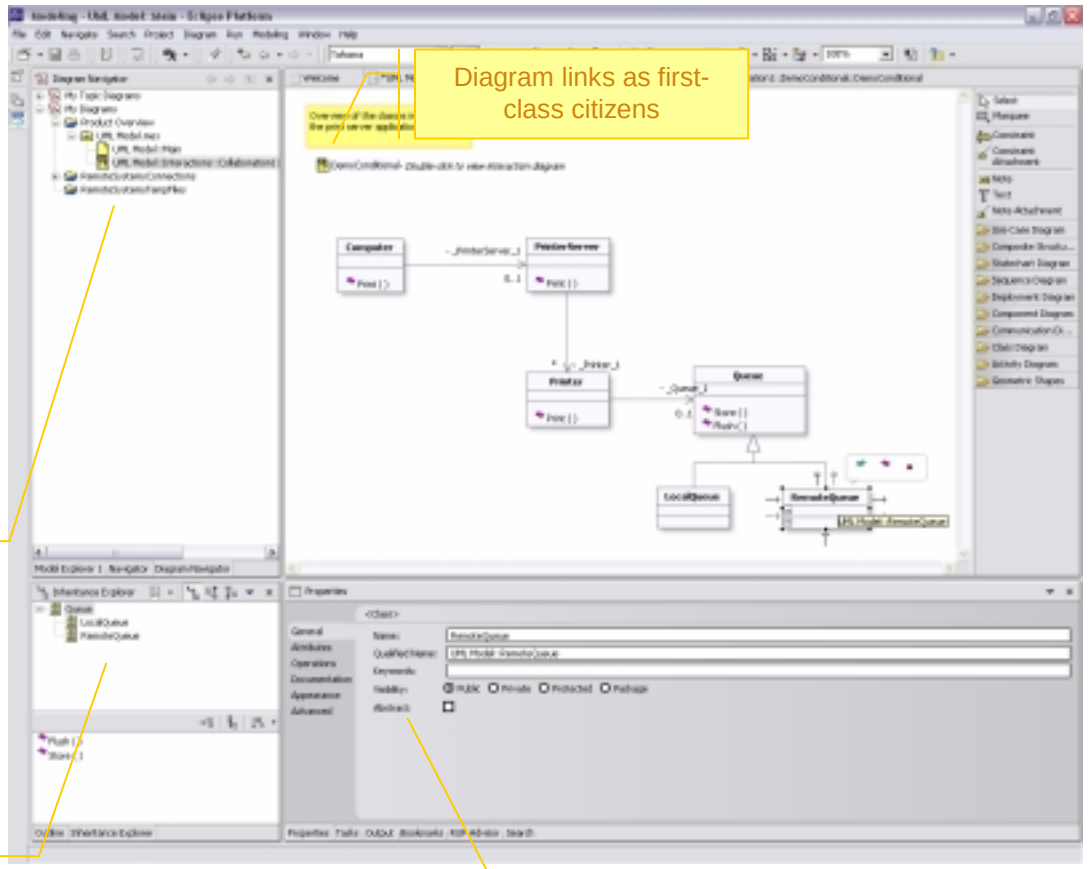
Interaction Overview Diagram



Composite Structure Diagram

Modelado UML 2.0

- Análisis y Modelado UML 2.0 dentro del mismo entorno de desarrollo Eclipse 3.0
- Asistentes en la edición de modelos
- Nuevas vistas configurables por el usuario

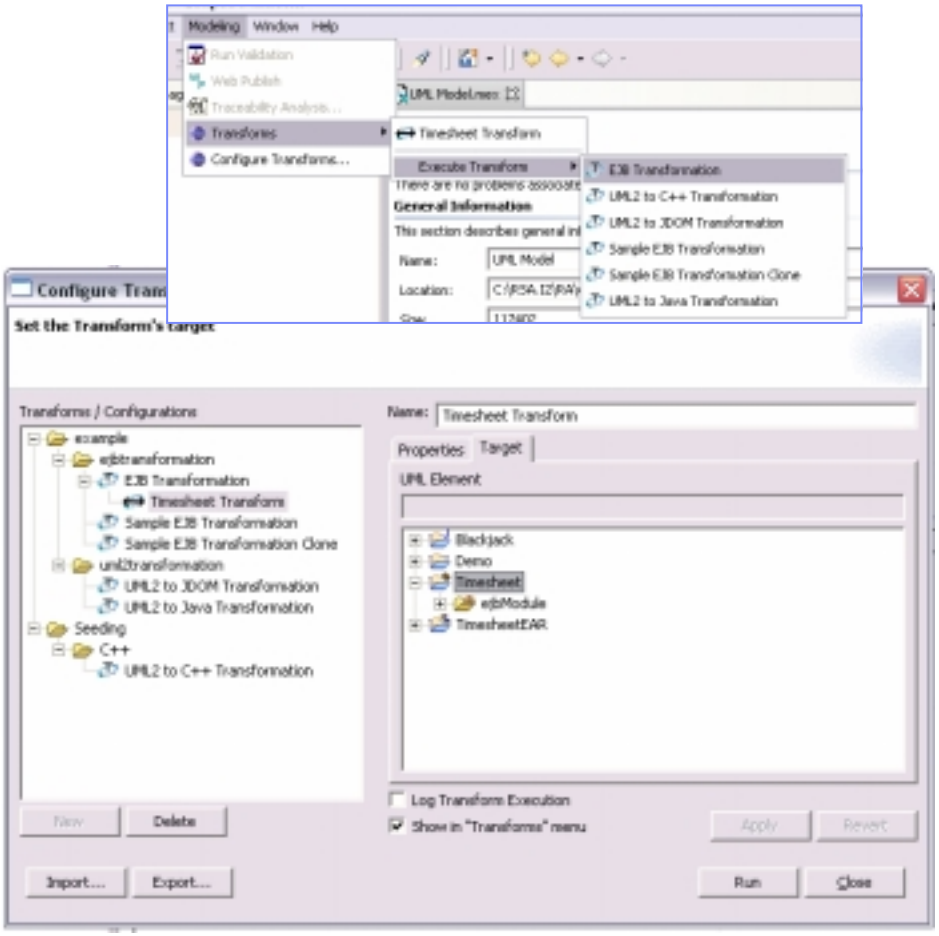


Demo Funcionalidad adicional de Rational Software Architect

- **Integración en Eclipse 3.0**
  - Nueva perspectiva de Modelado
- **Integración con otras herramientas del ciclo de vida:**
  - Gestión de Requisitos,
  - Construcción,
  - Pruebas,
  - Gestión de configuración,
  - Metodología
- **Soporte a UML 2.0**
- **Reutilización y Automatización con Patrones y Transformaciones**
  - Creación y aplicación de patrones
  - Generación y sincronización de código Java, C++
  - Visualización de la lógica del código
- **Validación Arquitectura**
  - Análisis y validación del modelo
  - Detección de anti-patrones y dependencias cíclicas

Transformaciones

- Posibilidad de realizar y configurar transformaciones:
  - Modelo a modelo
  - Modelo a código
- Transformaciones incluídas:
  - UML-a-Java/J2EE
  - UML-a-C++
  - Ejemplos de transformaciones simples de modelos
- Intercambio de transformaciones via RAS en IBM developerWorks

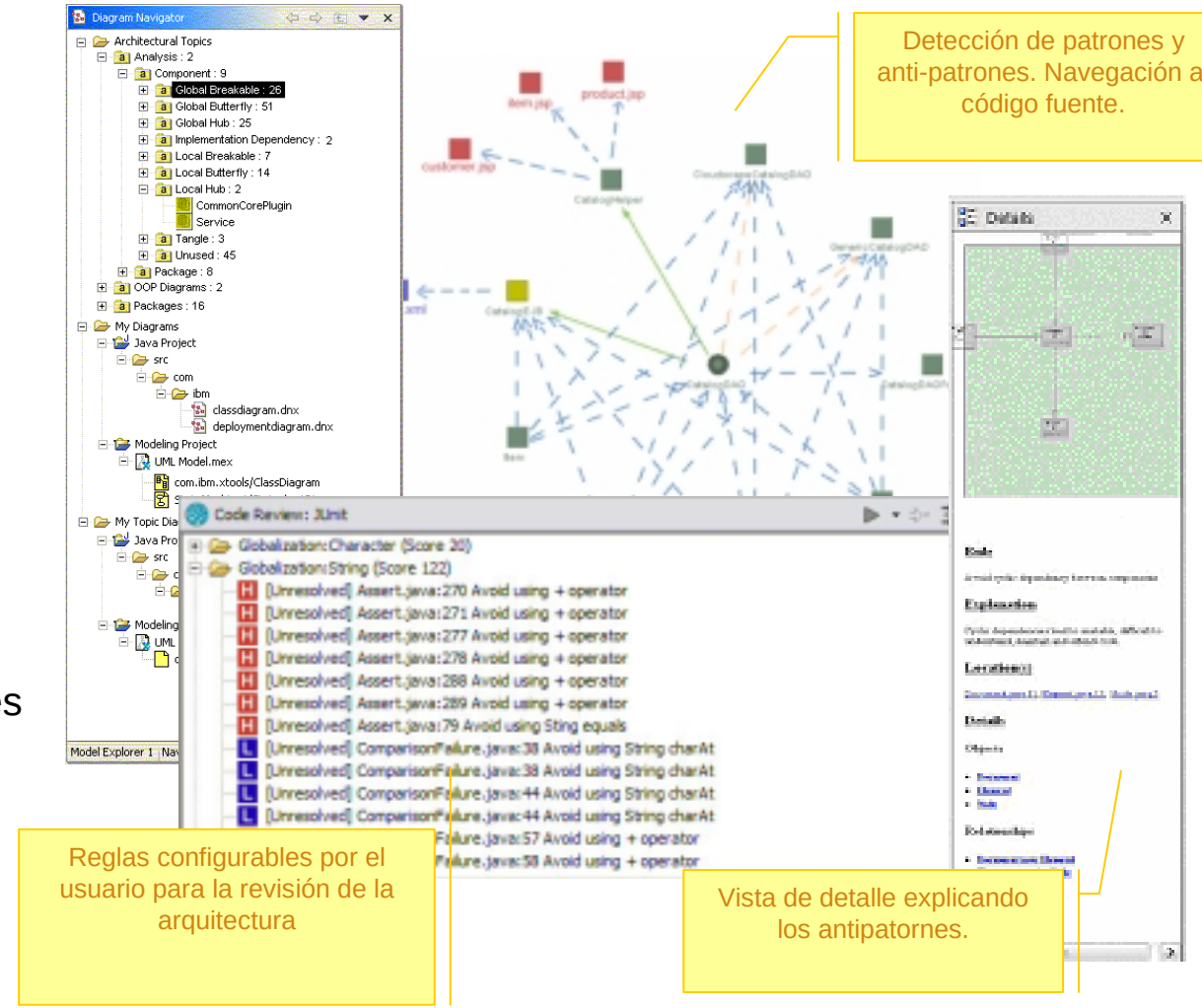


Demo Funcionalidad adicional de Rational Software Architect

- **Integración en Eclipse 3.0**
  - Nueva perspectiva de Modelado
- **Integración con otras herramientas del ciclo de vida:**
  - Gestión de Requisitos,
  - Construcción,
  - Pruebas,
  - Gestión de configuración,
  - Metodología
- **Soporte a UML 2.0**
- **Reutilización y Automatización con Patrones y Transformaciones**
  - Creación y aplicación de patrones
  - Generación y sincronización de código Java, C++
  - Visualización de la lógica del código
- **Validación Arquitectura**
  - Análisis y validación del modelo
  - Detección de anti-patrones y dependencias cíclicas

Control y Validación de la arquitectura

- **Validación del modelo**
  - Reglas UML 2.0
- **Análisis y Revisión de la Arquitectura**
  - Dependencias Circulares
  - Visualización patrones de diseño, OO
  - Detección de anti-patrones
  - Configuración de reglas específicas para aplicar controles y verificar estándares




Questions


Thank
You