UML 2.0

Telelogic Tau® Generation2

Antonio Rodríguez – Telelogic Iberica SL
antonio.rodriguez@telelogic.com

Estado actual de la Industria: ¿Un punto de ruptura?

Solo en los últimos 15 años, los defectos de software han:

- Destruido el Ariane5 durante su lanzamiento
- Retrasado un año la apertura del aeropuerto de Denver, el más caro en la historia de EEUU
- Estrellado una sonda de la NASA contra la superficie de Marte
- Asesinado 4 militares americanos al estrellarse un helicóptero
- Inducido a una fragata de la marina americana a destruir un avión civil
- Parado el sistema de control de ambulancias de Londres, produciendo más de 30 muertos

© July/August 2002, Technology Review, Inc.

2

Algunas razones …

- Los requisitos están incompletos o son malentendidos
- Fallos en la trazabilidad de los requisitos
- Poca reutilización de componentes
- Proceso manual muy intensivo en trabajo manual
- Falta de comunicación y colaboración
- Se necesitan desarrolladores altamente especializados en cada etapa
- Procesos de desarrollo inflexibles y herramientas muy rígidas
- Demasiadas herramientas a utilizar e integrar

3 Telelogic

Limitaciones de UML 1.x

- Limitado soporte en el desarrollo de sistemas
  - No soporta arquitecturas complejas muy especificas
  - No esta claramente definido como soportar el desarrollo basado en componentes
- No satisface todos los aspectos de desarrollo de software
  - Impreciso e incompleto
  - No permite la especificación completa del comportamiento
- Ha creado un nuevo problema: "roundtrip engineering"
  - UML 1.x no genera el 100% del código
- UML ha fallado en su penetración en el mercado

"Entre el 7 y 10% de los desarrolladores utilizan UML"

Thomas Murphy, META Group

4 Telelogic

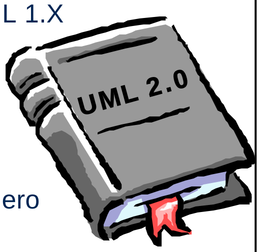
La evolución de UML

- UML 2.0
  - Diseñado para corregir las limitaciones de UML 1.0
  - Mejorada la visualización de requisitos
    - Describir interacciones complejas
  - Mejorado el soporte para sistemas complejos
    - Definición de componentes
    - Especificación de Arquitecturas
    - Especificación de Interfaces
  - INCOSE y OMG iniciativa
    - “UML for Systems Engineering”

5

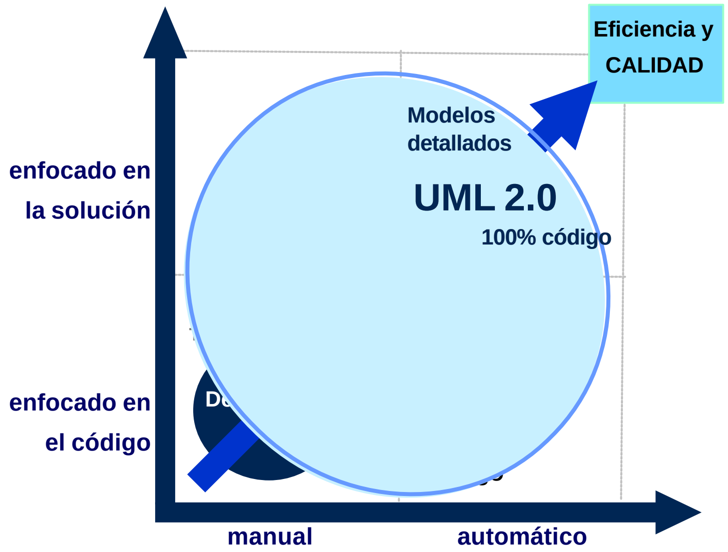
UML 2.0

- UML esta sufriendo actualmente una importante revisión
  - basada en la experiencia de los usuarios y fabricantes de herramientas
  - usabilidad y escalabilidad
- Nuevas tecnologías han emergido desde que UML 1.X fue estandarizado
  - desarrollo basado en componentes
  - modelos ejecutables
- UML 2.0 esta previsto que sea aprobado para Enero de 2003



6

Evolución de UML
Desarrollo visual a través de modelos



7 © Telelogic AB

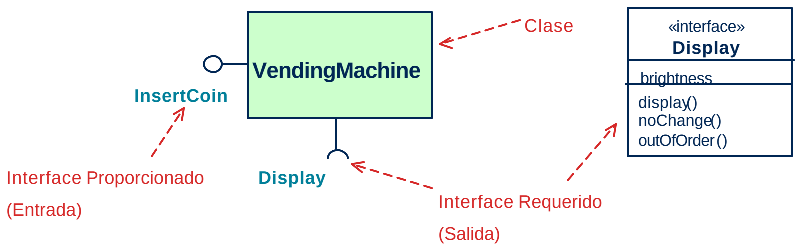
Telelogic - Liderando la definición de UML 2.0

- Co-director de 3 subcomités
- Promotor y miembro significativo de “U2 Partners consortium”

8 © Telelogic AB

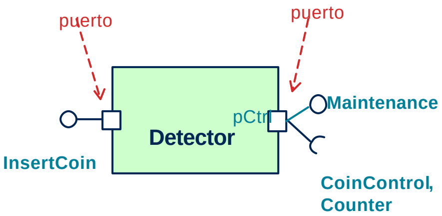
Desarrollo basado en componentes

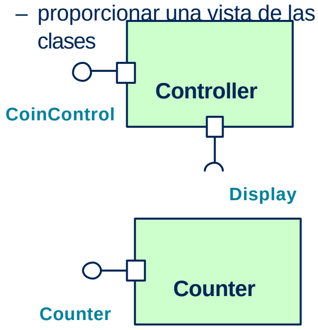
- Diseño de interfaces
  - interfaces permiten definir las clases como entidades independientes
- Encapsulación
  - la clase se trata como una ”caja negra”
  - se necesita conocer como utilizar los interfaces



Especificación de la comunicación entre clases

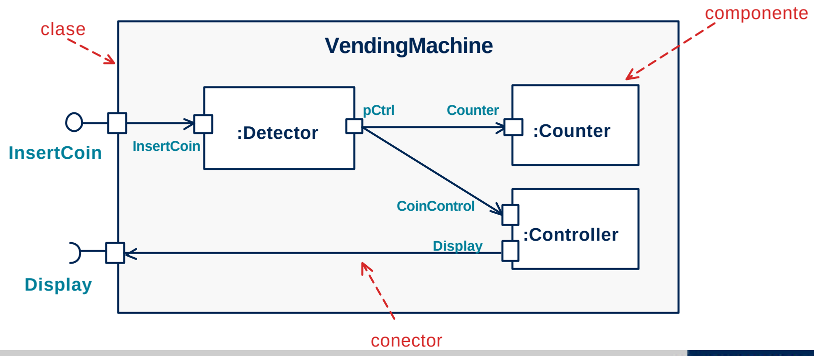
- Solo aquellas clases con interfaces coincidentes pueden comunicarse entre si.
- Un puerto tiene varias funciones:
  - actuar como punto de interacción entre clases
  - proporcionar una vista de las clases





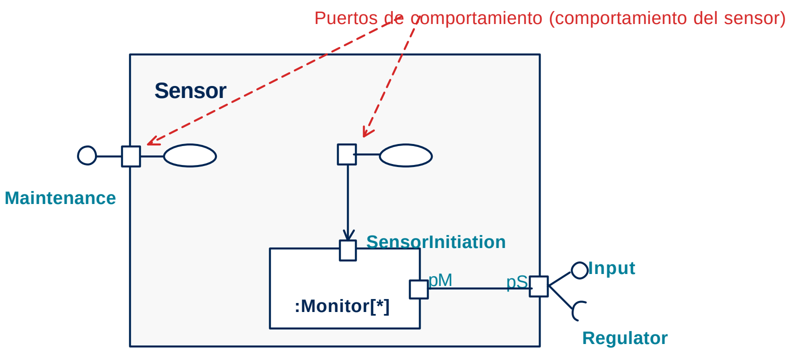
Integración de clases

- Una clase puede usarse como parte de la estructura interna de otra clase
  - descomposición jerárquica
  - vista "caja blanca"
- Conectores se usan como asociaciones contextuales
  - representan los caminos de comunicación



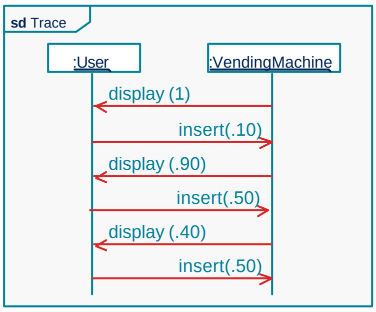
Estructura interna y comportamiento

- Comportamiento puede mezclarse con la estructura interna
- Permite la comunicación entre el contenedor y sus componentes

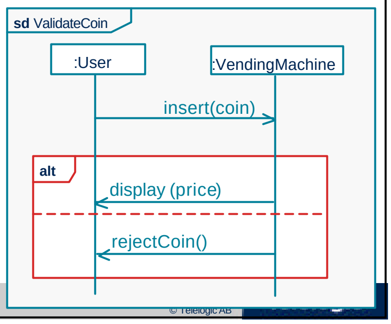


Secuencias

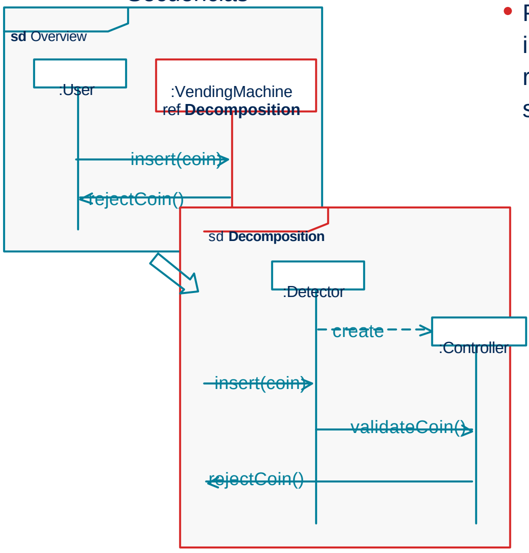
- Propósito:
  - trazar
  - captura de requisitos
  - casos y escenarios de test



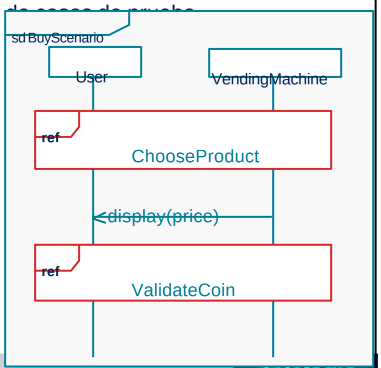
- Expresar variaciones
  - paralelismo
  - alternativas
  - iteraciones
  - opciones
- Reducción dramática del numero de diagramas de secuencias



Descomposición de Secuencias

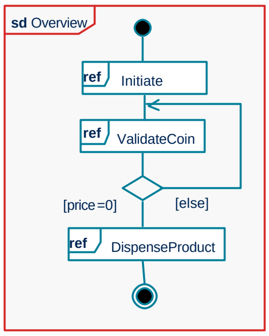


- Para evitar repeticiones innecesarias, es posible hacer referencia a diagramas de secuencia existentes
  - permitiendo la rapida creación de escenarios. Por ejemplo escenarios de pruebas a partir

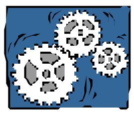


Organizar secuencias

- Es posible organizar los diagramas de secuencia como flujos para indicar su comportamiento
  - visión general
  - combinar secuencias para generar escenarios



15 © Telelogic AB Telelogic



UML ejecutable

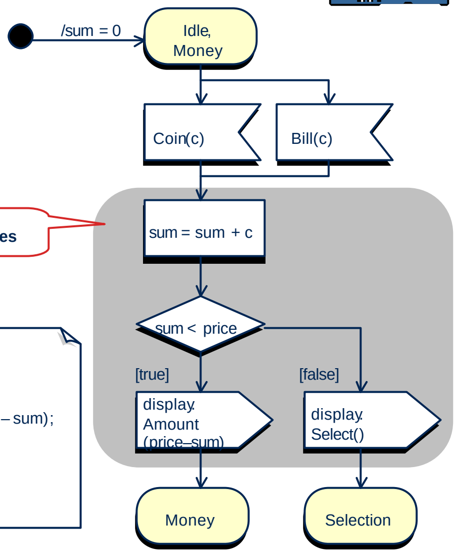
- Utilización de *"action semantics for UML"* para permitir la ejecución de modelos
  - definición del significado preciso de acciones como asignaciones, decisiones o bucles
- Chequear el sistema mediante la verificación del mismo en la primeras fases

16 © Telelogic AB Telelogic

Expresar funcionalidad

- Funcionalidad puede ser expresada a través de:
  - interacciones
  - maquinas de estado
  - actividades
- Permite a las herramientas extender el soporte al desarrollo
  - verificación de modelos (simulación)
  - validación
  - pruebas

Acciones




```
sum=sum + c;
if (sum < price)
  display.Amount(price – sum);
else
  display.Select();
```

```
sum = sum + c;
switch (sum < price) {
 case true:
  display.Amount(price – sum);
  break;
 case false:
  display.Select();
  break;
}
```

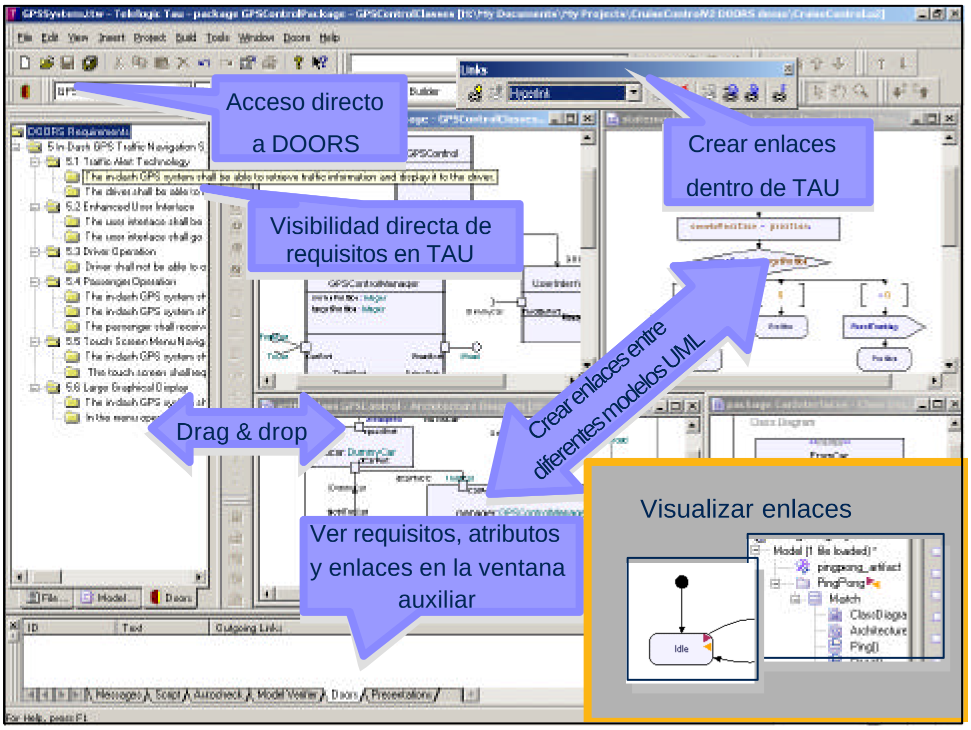
17 © Telelogic AB Telelogic

Desarrollo de aplicaciones con UML 2.0

Un ejemplo:

Desarrollo de un sistema de control de mísiles utilizando TAU G2 y DOORS

18 © Telelogic AB Telelogic



Más información

- Antonio Rodríguez Perales: antonio.rodriguez@telelogic.com
- Pagina web Telelogic: http://www.telelogic.com
- Pagina web específica de Telelogic Tau G2: http://www.taug2.com
- U2 Partners: http://www.u2-partners.org
- Libro “Requirements Engineering”
  http://www.telelogic.com/news/publications/reading/index.cfm

36 Telelogic