Modelado de Sistemas Embebidos

Andrés Djordjalian <andres@indicart.com.ar>
Seminario de Sistemas Embebidos
Facultad de Ingeniería de la U.B.A.

Modelado de Sistemas Embebidos

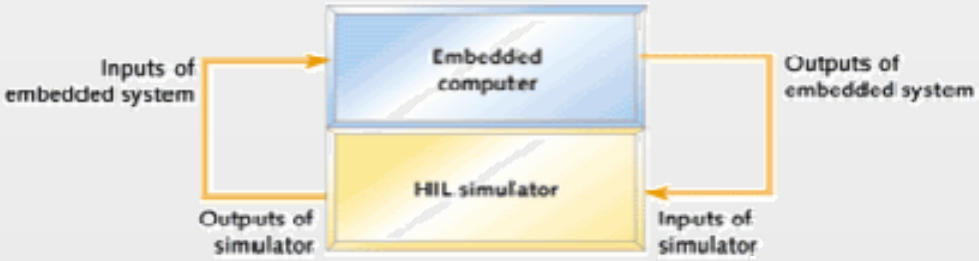
- **Temario**
  1. ¿Qué es el modelado?
  2. Modelos de computación
  3. Tendencias
  4. UML
  5. Desarrollo basado en modelos
  6. Herramientas
- **La presentación tiene actividades**
  - Para eso, vayan formando grupos
    - de 3 o 4 personas.
    - Vean quién vive más cerca de FIUBA
      - Esa persona va a hacer las anotaciones y ser el o la **vocero** del grupo, cuando discutamos entre todos las conclusiones.
        - » Los demás pueden explicarle y hacer comentarios cuando discutamos, pero sin sacarle el laburo.

¿Qué es un Modelo?

- **Es la descripción del funcionamiento o la estructura de un sistema, o de alguna de sus partes, en un nivel alto de abstracción**
  - Pueden emplearse en los requerimientos, en la definición de la arquitectura o en el diseño detallado
- **Lo forman uno o más artifacts**
  - Que pueden estar en pizarrón (*whiteboard*), papel, o en un archivo de computadora.
- **Los modelos generalmente están expresados en lenguajes cercanos al problema**
  - Frecuentemente son **gráficos** o **matemáticos**
- **Ejemplos: escribir un código Matlab, diagramar en Simulink o en un diagrama de estados.**

El Modelado Sirve Para:

- **Organizar y comunicar ideas eficientemente.**
  - …al pensar un diseño, hacerlo en equipo, y documentar
- **Encontrar defectos temprano**
  - …si es que se puede ejecutar o chequear formalmente
    - A veces se dice *simular* en lugar de *ejecutar*
    - Las técnicas de chequeo formal intentan *demostrar* que es correcto, como se demuestra un teorema
- **Implementar un sistema embebido**
  - …si contamos con herramientas de MDD
    - MDD=*Model-Driven Development*
      - (vamos a verlo más adelante)
- **Representar el *entorno* de un sistema embebido, para verificarlo**
  - Ej., simulación de “hardware in the loop” (HIL, ver figura)



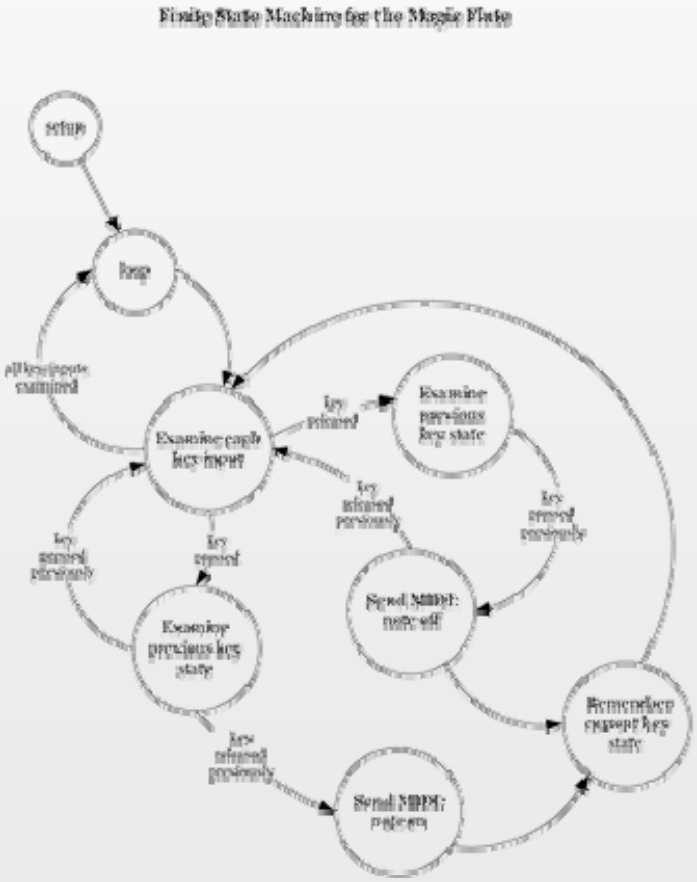
Fuente: embedded.com

Lenguajes de Modelado

- **Son lenguajes artificiales para construir modelos**
  - Ejemplos: El lenguaje del MATLAB o la notación típica de una máquina de estados finitos
- **Propósitos:**
  - Evitar malentendidos
  - Habilitar el uso de herramientas y su interoperabilidad
  - Facilitar modos eficientes de expresar ideas
- **Muchos de estos lenguajes son gráficos**
- **Algunos son de propósito general y otros son *domain-specific***
  - …o sea, especiales para determinados problemas
- **Los hay abiertos, otros son propietarios**
- **Frecuentemente emplean modelos de computación.**

Modelos de Computación

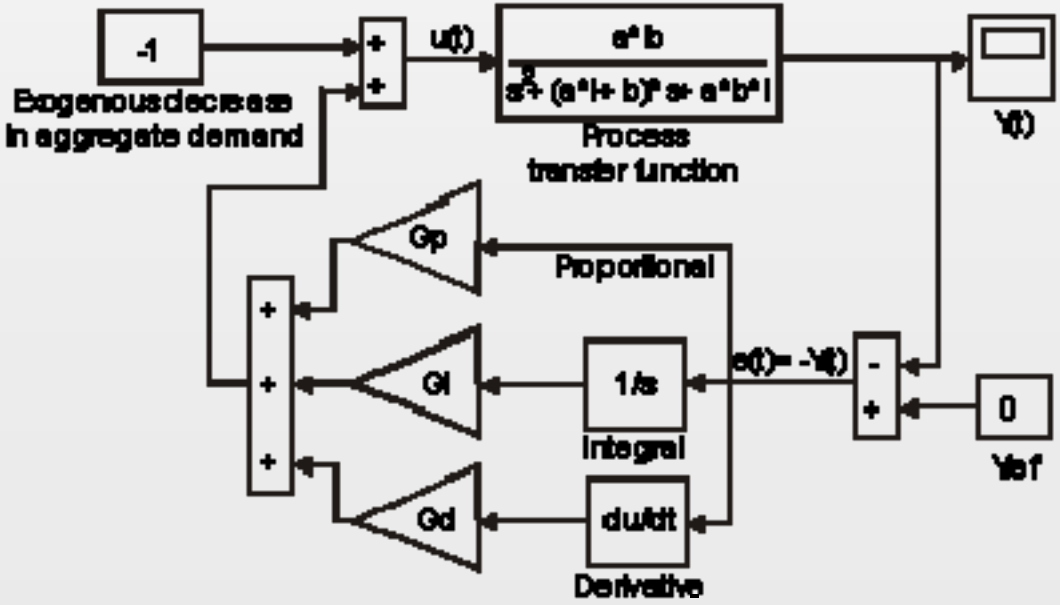
- **Son definiciones abstractas de máquinas capaces de computar**
- **Ejemplo: una máquina de estados finitos (FSM)**



- **Algunos lenguajes de modelado incluyen, por ejemplo diagramas de estado**
  - Ej: los StateCharts de UML, o la extensión Stateflow de Simulink
  - A veces les incorporan elementos nuevos y/o una sintaxis particular
    - Más adelante vamos a verlo para el caso de los StateCharts

Modelo de Flujo de Datos

- **Representación gráfica de cómo se mueven los datos entre los distintos procesos o componentes**
  - Como la que se usa en DSP
- **También se le dice Data-Flow Diagram (DFD)**
- **Puede ser en tiempo discreto o continuo**



Redes de Petri

- **Sirven para representar sistemas con concurrencia y necesidades de sincronización**
  - Ej., porque compiten por recursos

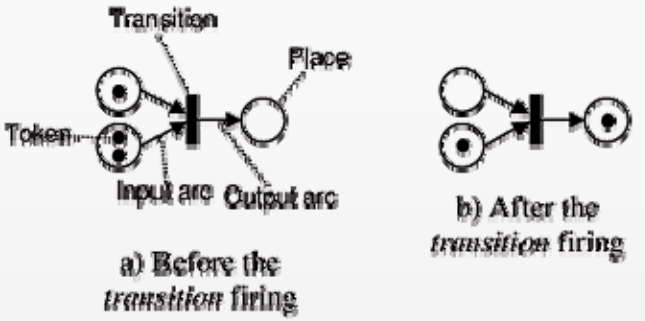
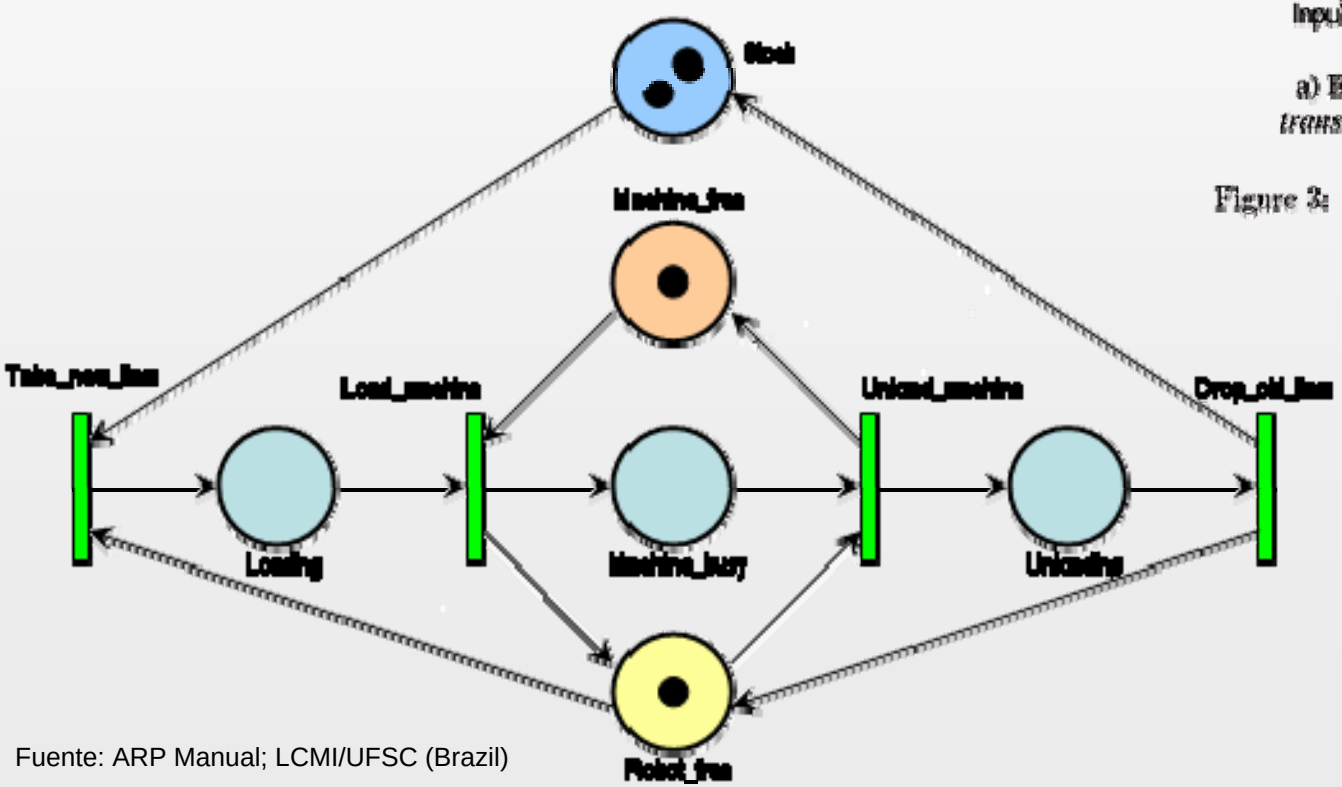
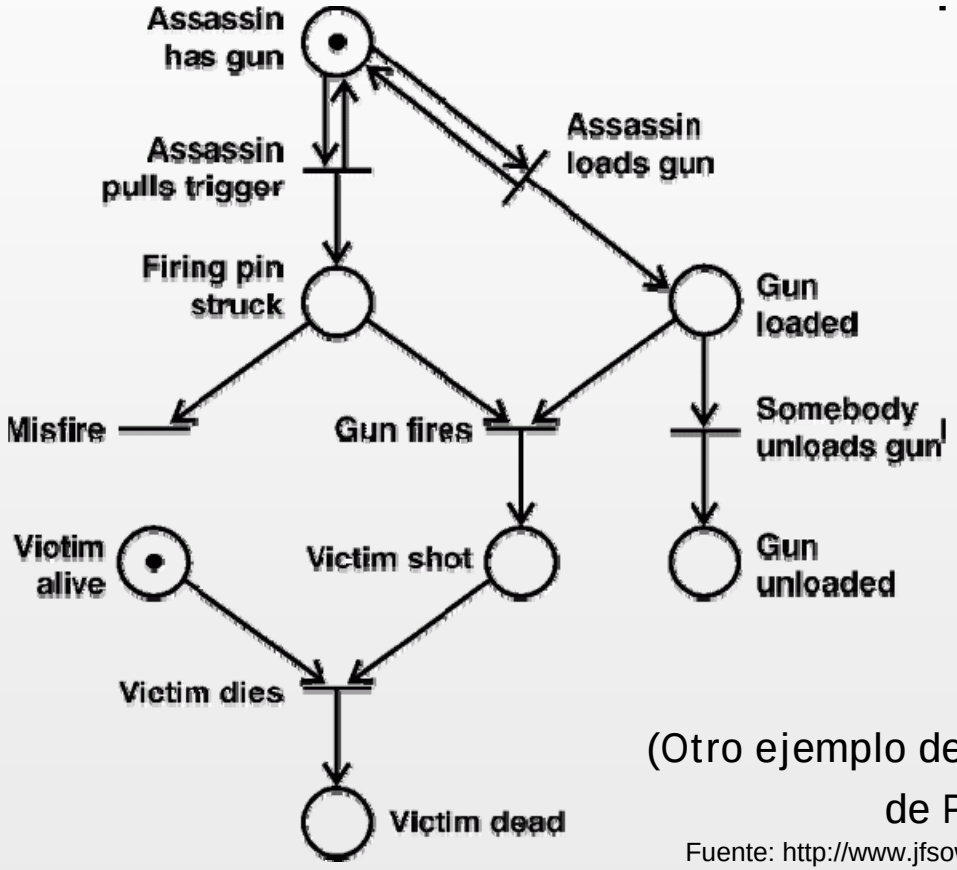


Figure 3: Example of Petri net *trasition firing*



Fuente: ARP Manual; LCMI/UFSC (Brazil)

Redes de Petri



(Otro ejemplo de Red de Petri)

Fuente: http://www.jfsowa.com

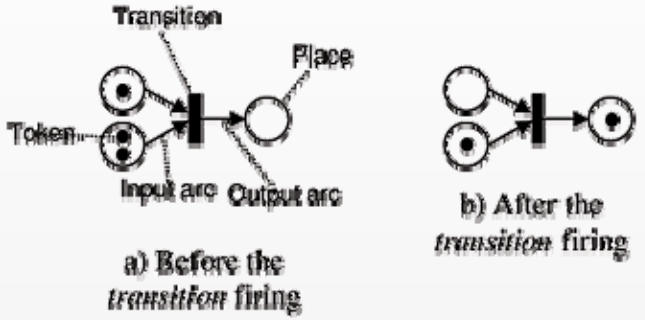


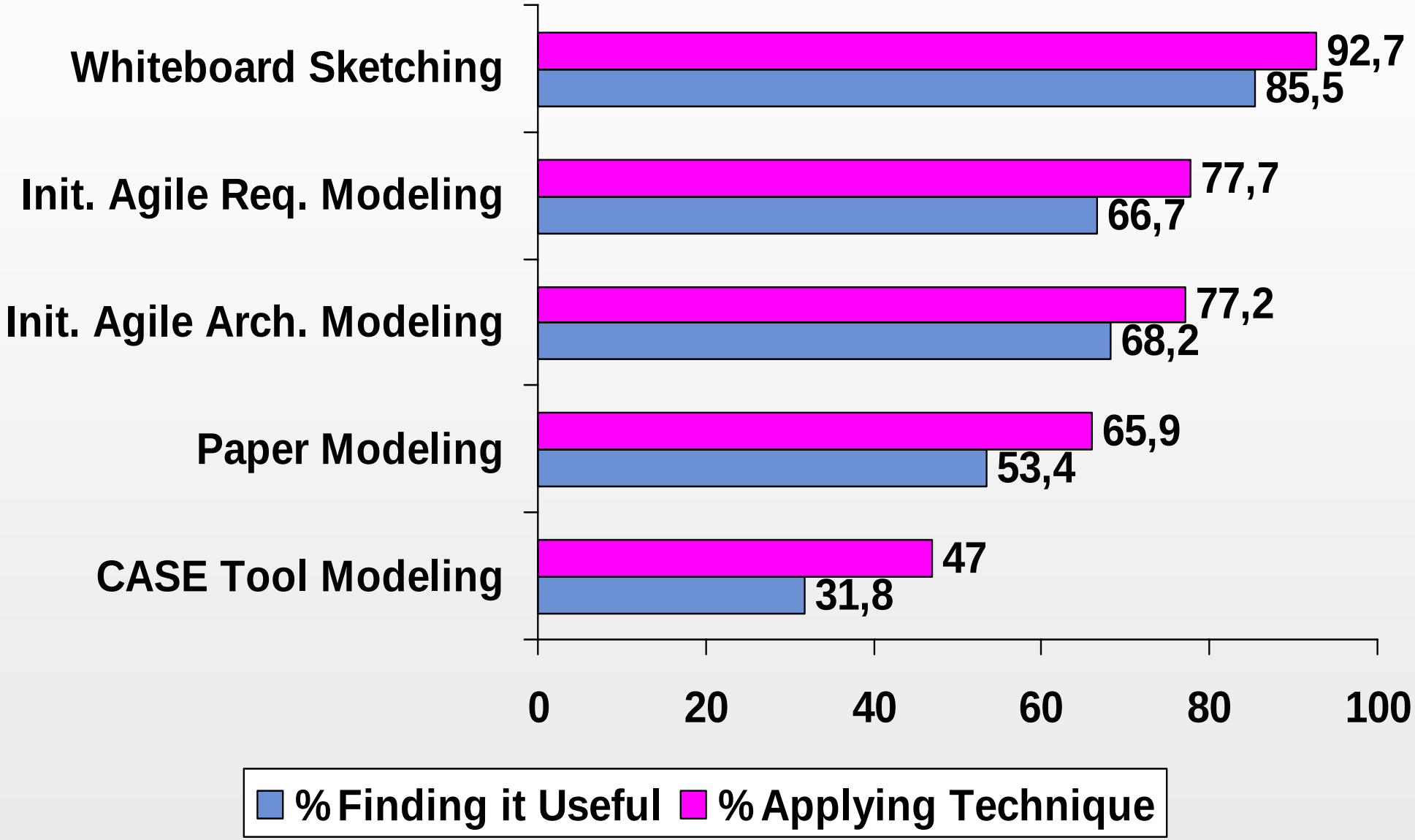
Figure 3: Example of Petri net *trasition firing*

Utilización de Lenguajes de Modelado en la Industria

- **2006 State of the Embedded Market Survey: Encuesta a 1217 suscriptos a publicaciones sobre embebidos y visitantes a conferencias.**
- **Preguntas:** "My current embedded project uses..." y "My next embedded project is likely to use..."

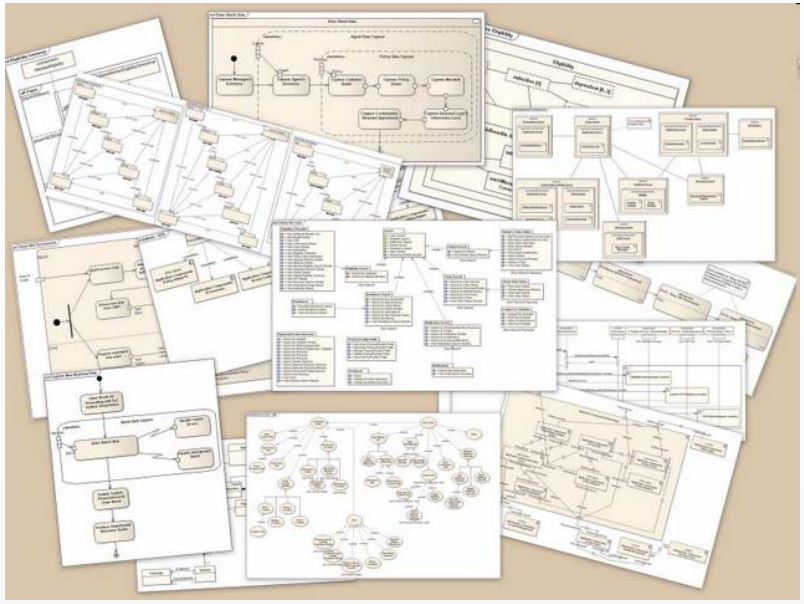
	2006 %	2005 %
Current Project		
SystemC or other "hardware C" language	19	16
UML	15	17
Other hardware/software codesign or coverficiation tool	15	11
SimuLink or other modeling language	12	12
None of the above	53	58
Next Project		
UML	23	26
SystemC or other "hardware C" language	21	21
SimuLink or other modeling language	17	16
Other hardware/software codesign or coverficiation tool	18	17
None of the above	44	44

Survey Says: Agilists are Modeling

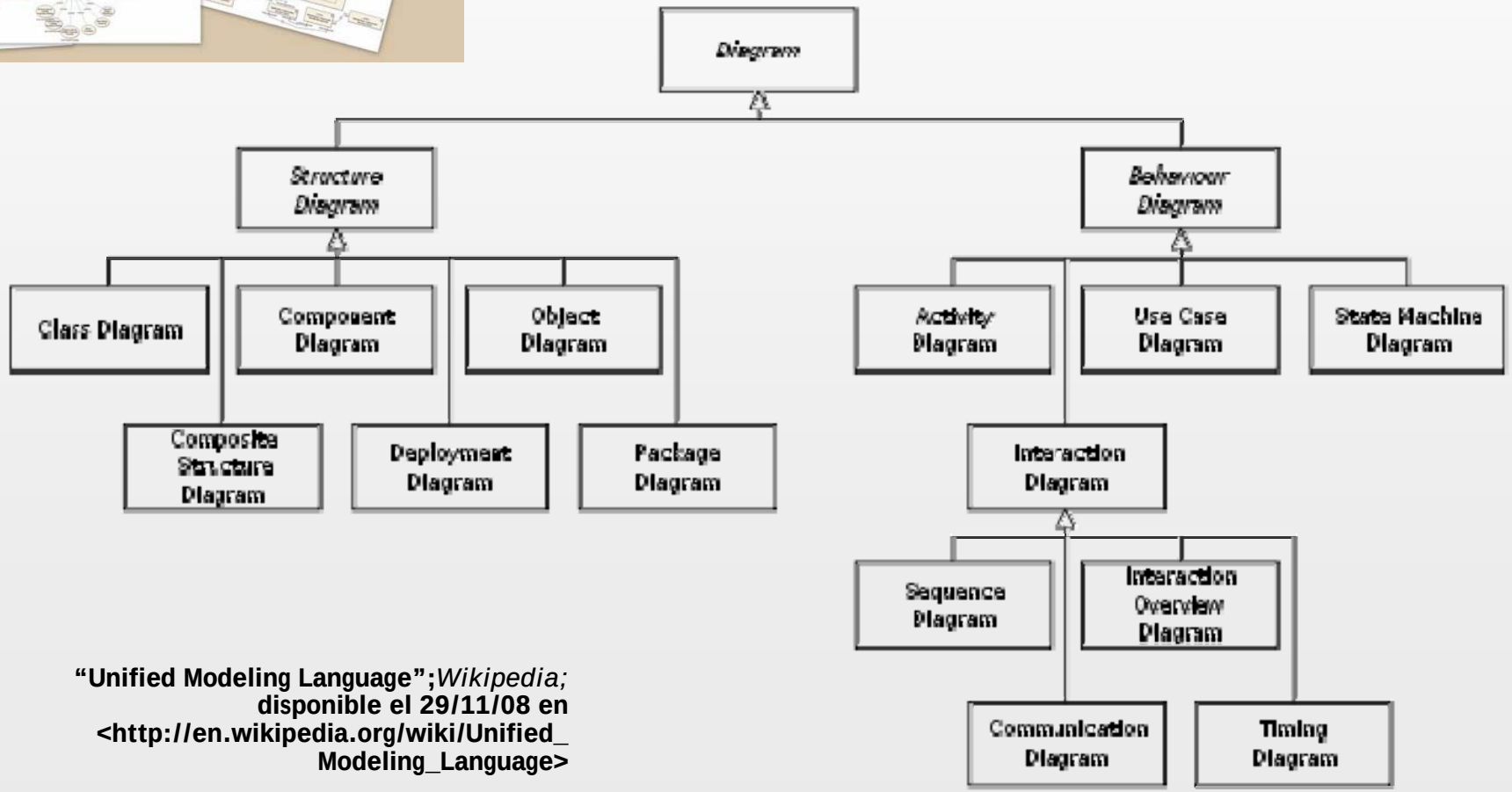


Unified Modeling Language (UML)

- **Lenguaje de modelado estándar en la industria del software**
  - Desarrollado por Rumbaugh and Booch, dos especialistas en orientación a objetos.
  - Lanzado en 1996. La versión actual es la 2 (2005).
    - La subversión más reciente es la 2.2.
  - Emplea muchos símbolos que ya se usaban comúnmente desde antes.
  - Está basado en orientación a objetos pero también puede aprovecharse con otros paradigmas.
- **Dada su influencia, hay lenguajes de modelado ajenos al software, y domain-specific, que se basan en UML**
  - Ej.: Systems Modeling Language (SysML)
- **Tiene mecanismos de extensión**
  - Además, usarlo flexiblemente es una práctica frecuente, en lugar de respetar estrictamente la norma.
  - Con estos mecanismos se creó un profile de UML ejecutable (i.e., que se puede simular) llamado **Exectuable UML** o **xUML**.



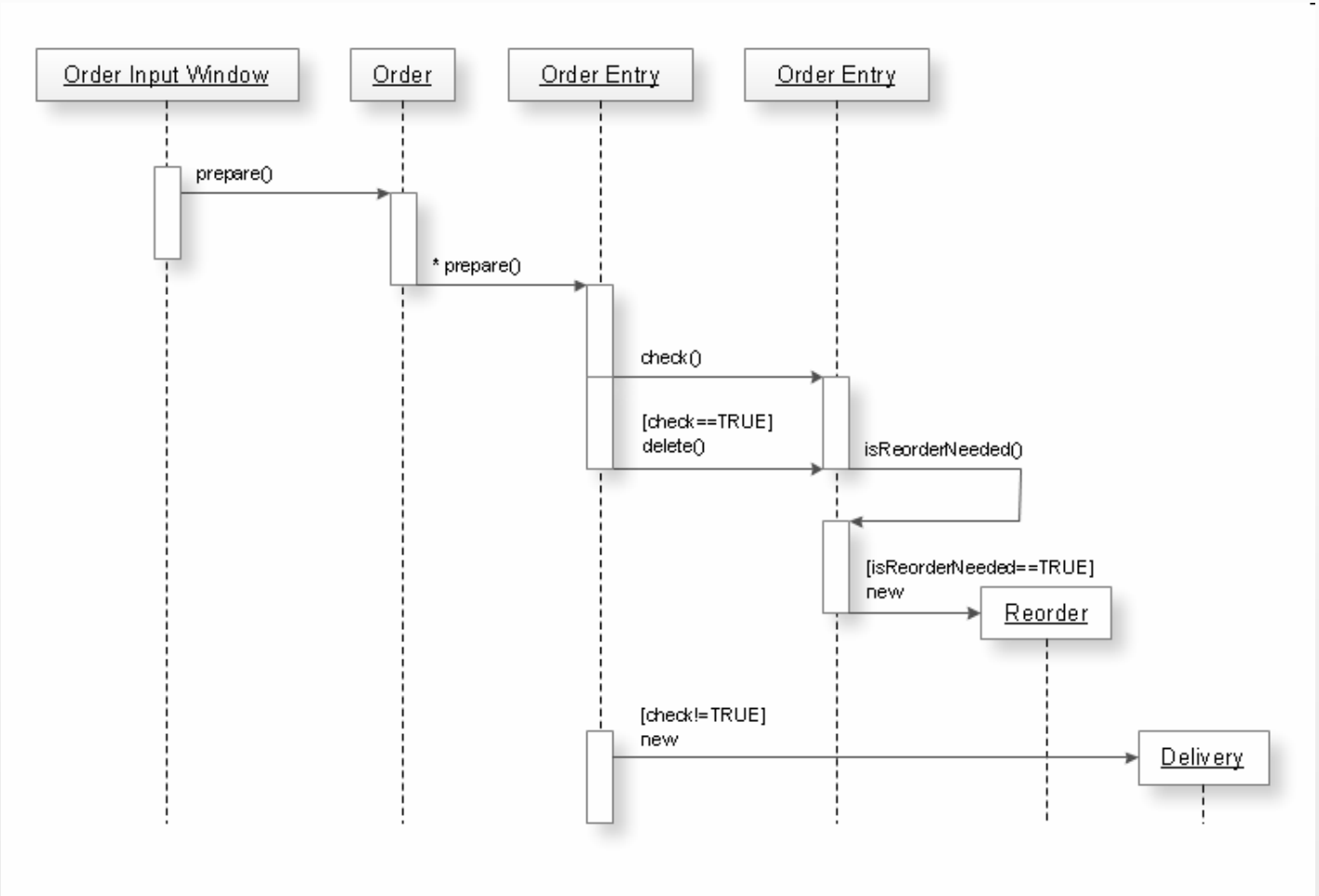
Diagramas del UML 2.0



"Unified Modeling Language"; *Wikipedia;* **disponible el 29/11/08 en <http://en.wikipedia.org/wiki/Unified_Modeling_Language>**

Diagrama de Secuencia

- **Muestran las comunicaciones entre varios objetos, a lo largo del tiempo**
- **La línea punteada se hace rectángulo cuando el objeto está "activo"**

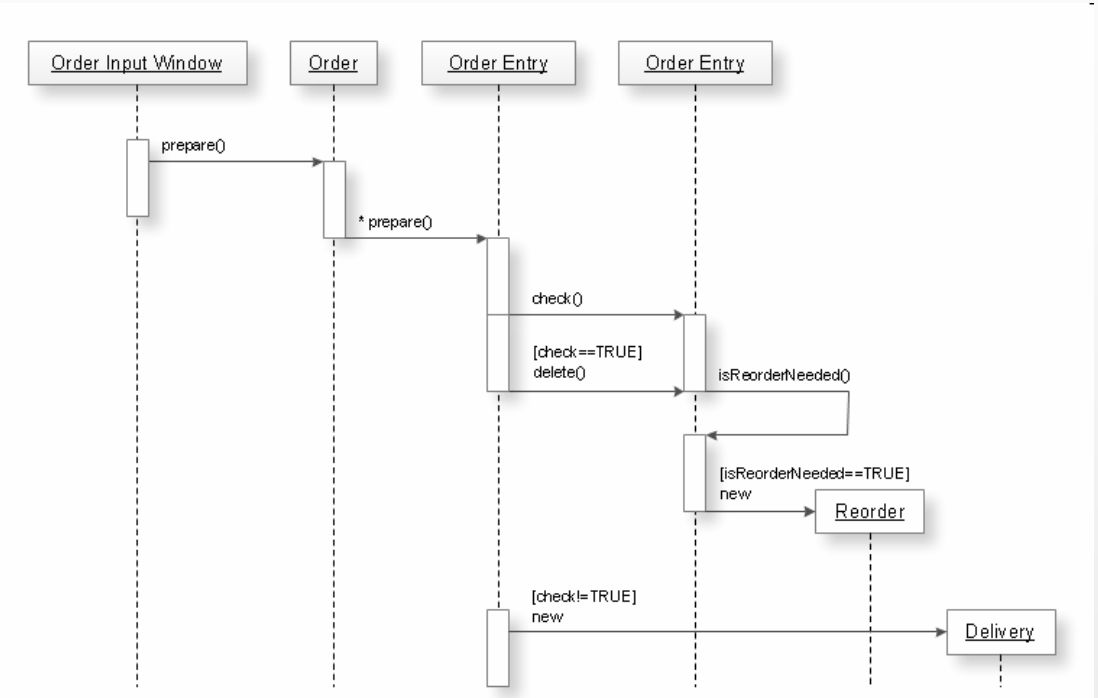


Fuente: http://www.conceptdraw.com/en/products/cd5/ap_uml_tool.php

Actividad

Diagrama de Secuencia

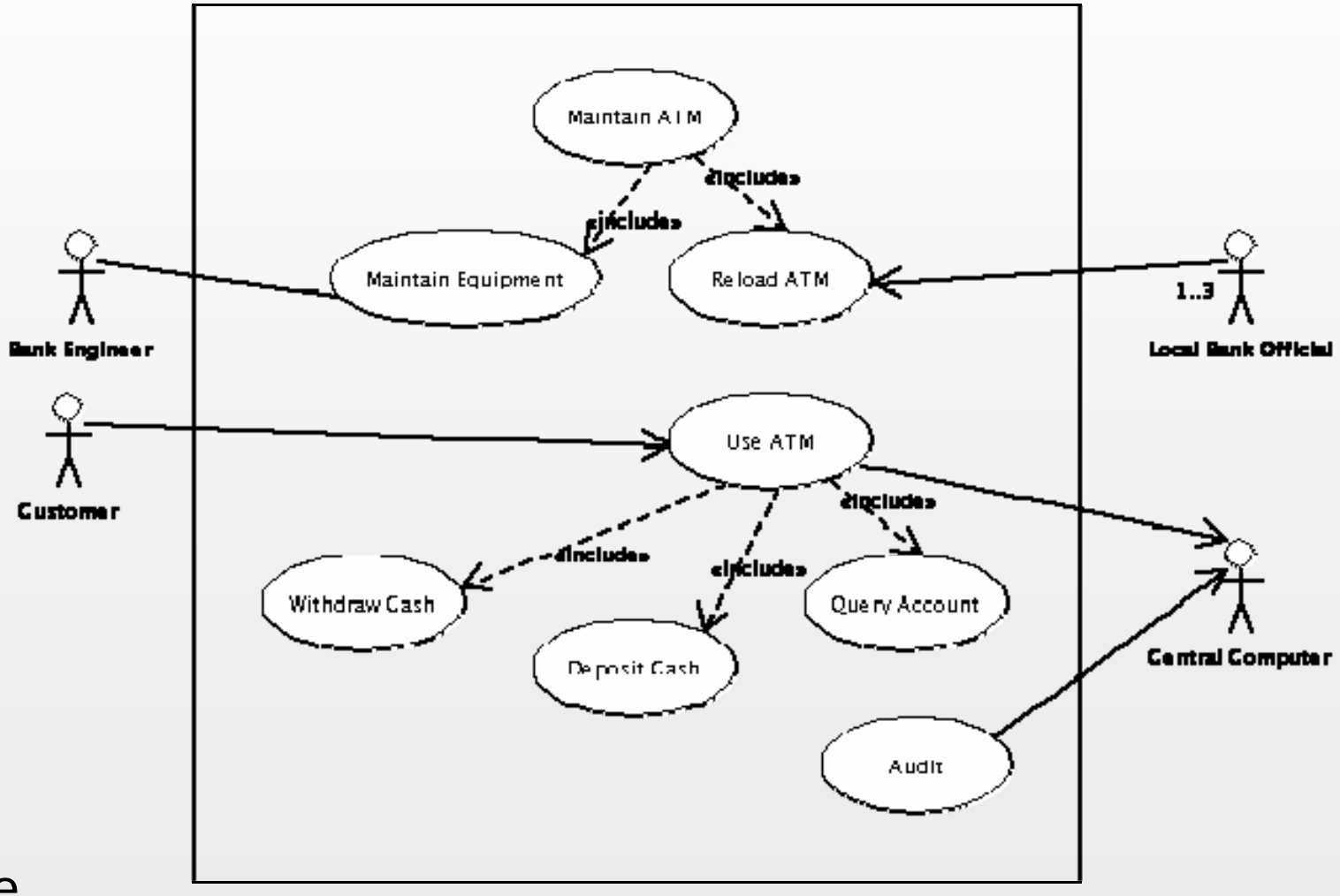
- **Describir, mediante un diagrama de secuencia, la comunicación entre un microcontrolador y un periférico, en donde el primero le manda un byte ‘A’ al segundo, seguido de un byte ‘B’**
- **La sincronización se realiza mediante dos señales (conexiones) unidireccionales:**
  - **`‘req’`** (request)
    - De la MCU al periférico
  - **`‘ack’`** (acknowledge)
    - Del periférico a la MCU
- **La conexión restante es `‘data’` (de 8 bits)**



Fuente: http://www.conceptdraw.com/en/products/cd5/ap_uml_tool.php

Diagrama de Casos de Uso

- **En inglés:** *Use case diagram*
- **Muestra a los usuarios del sistema (o sea,** *actores****), sus objetivos al utilizarlo (o sea,** *casos de uso****), y las relaciones entre los casos de uso**
  - Un actor no necesariamente es una persona

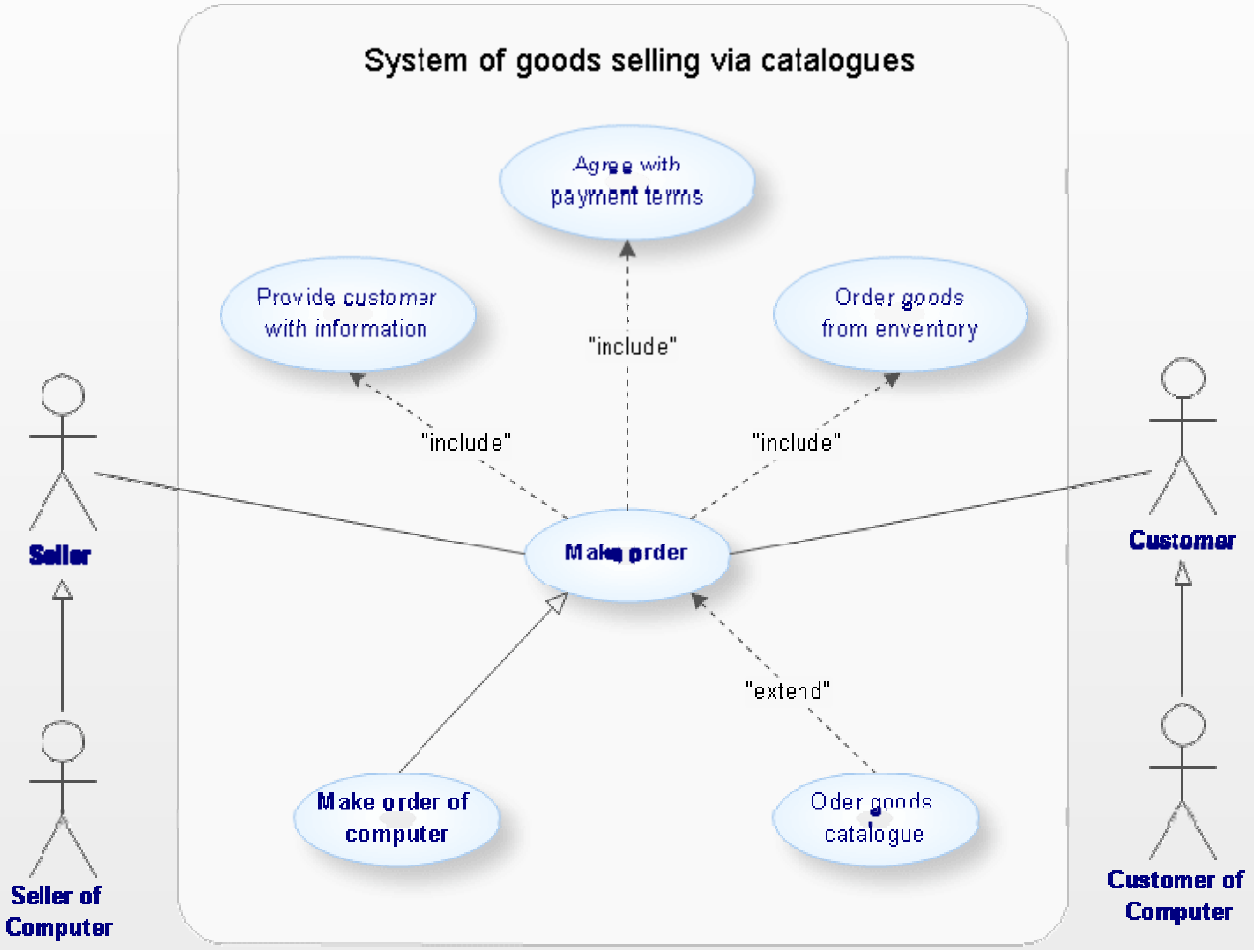


ArgoUML User Manual, **A. Ramirez et al. ; disponible el 29/11/08 en <http://argouml-stats.tigris.org/documentation/printablehtml/manual/argomanual.html>**

Actividad

Diagrama de Casos de Uso

- **Relaciones:**
  - A <<include>> B
    - cuando el caso de uso A incluye al B
  - A <<extend>> B
    - cuando A es un adicional que se usa en casos particulares de B
  - A, línea con flecha sin llenar, B
    - cuando A es un tipo de B
  - Ver ejemplos en la figura



Fuente: http://www.conceptdraw.com/en/products/cd5/ap_uml_tool.php

StateCharts

- **Son diagramas de estados, mejorados**

FSM

\+ Jerarquía

Concurrencia

StateCharts

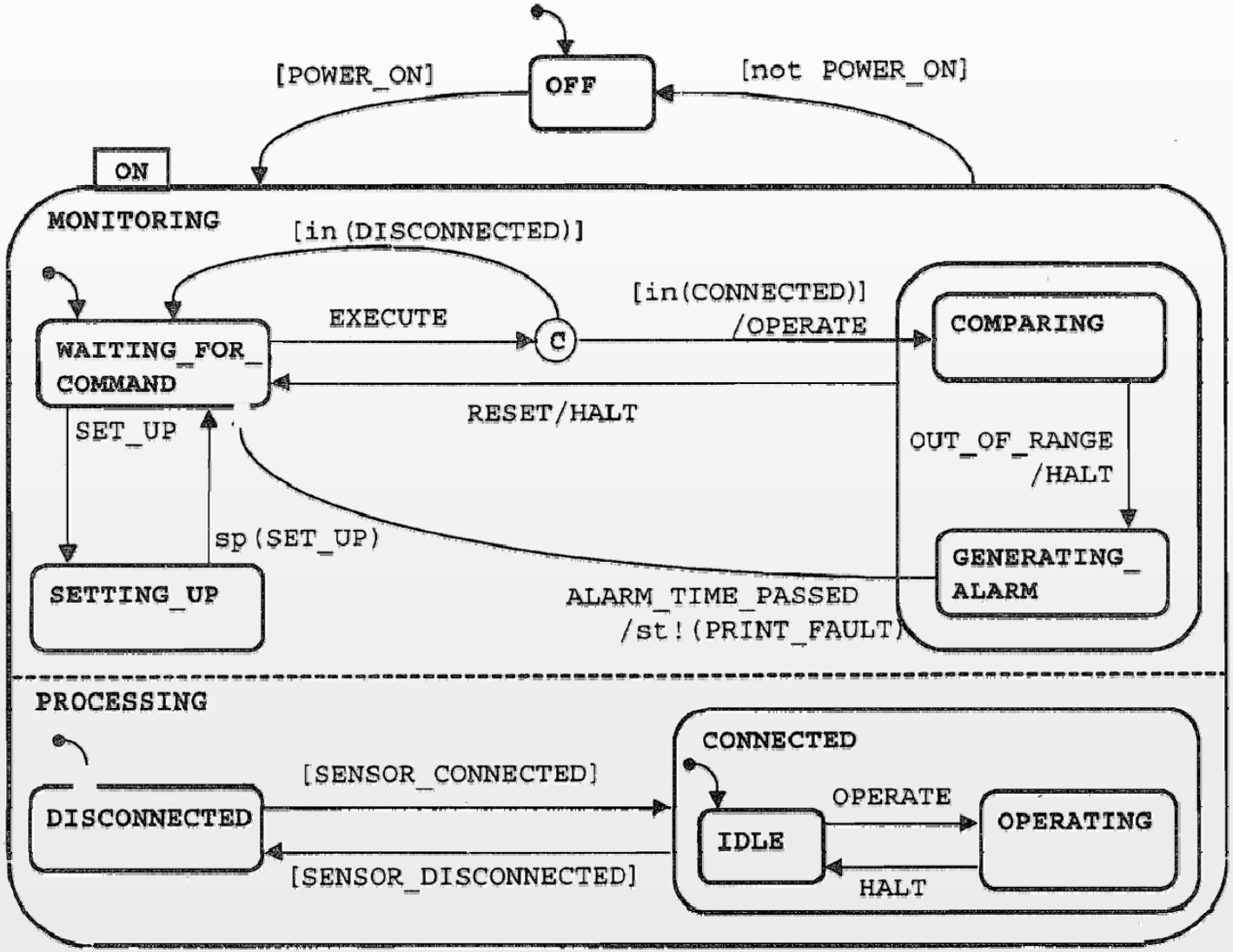
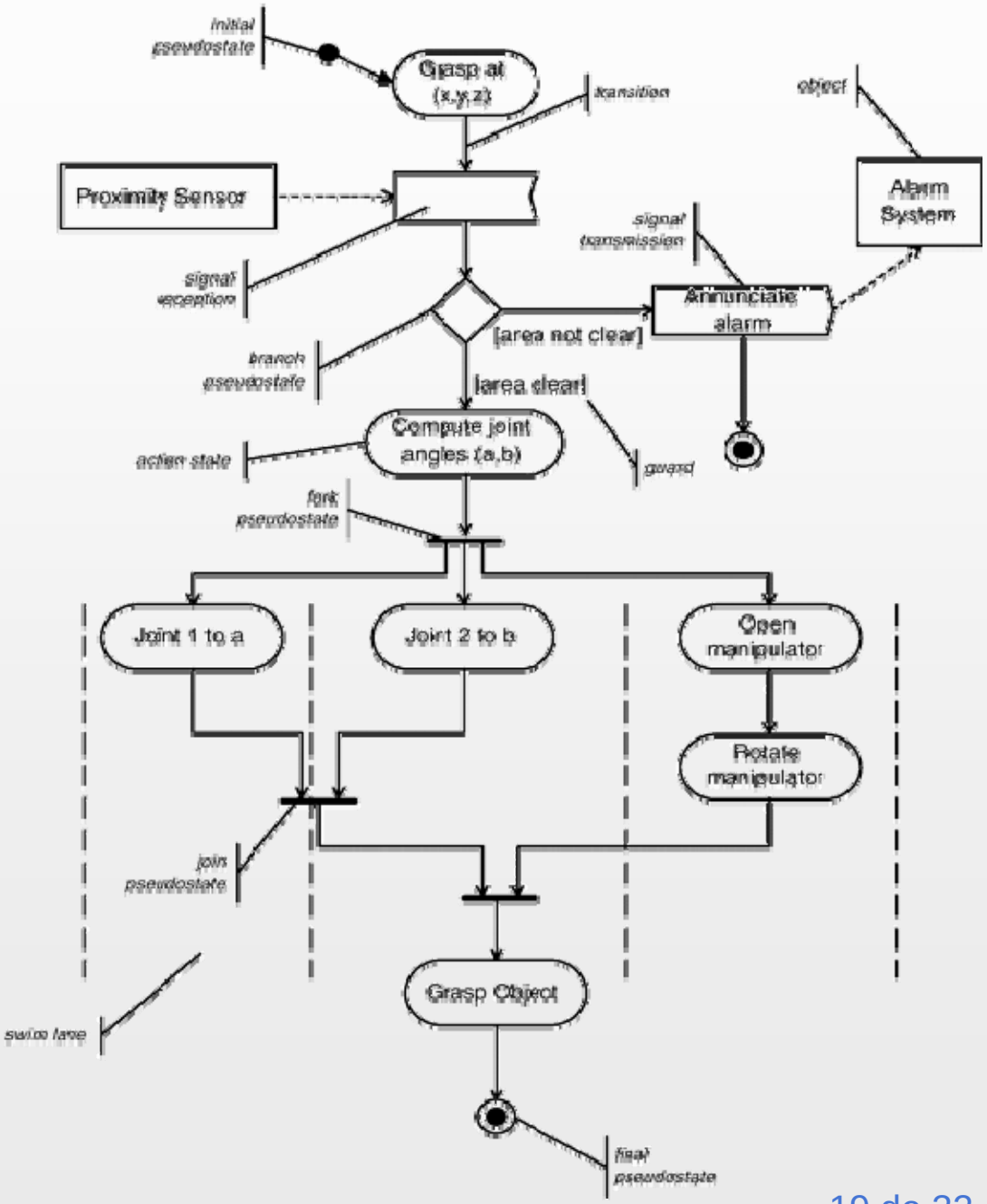


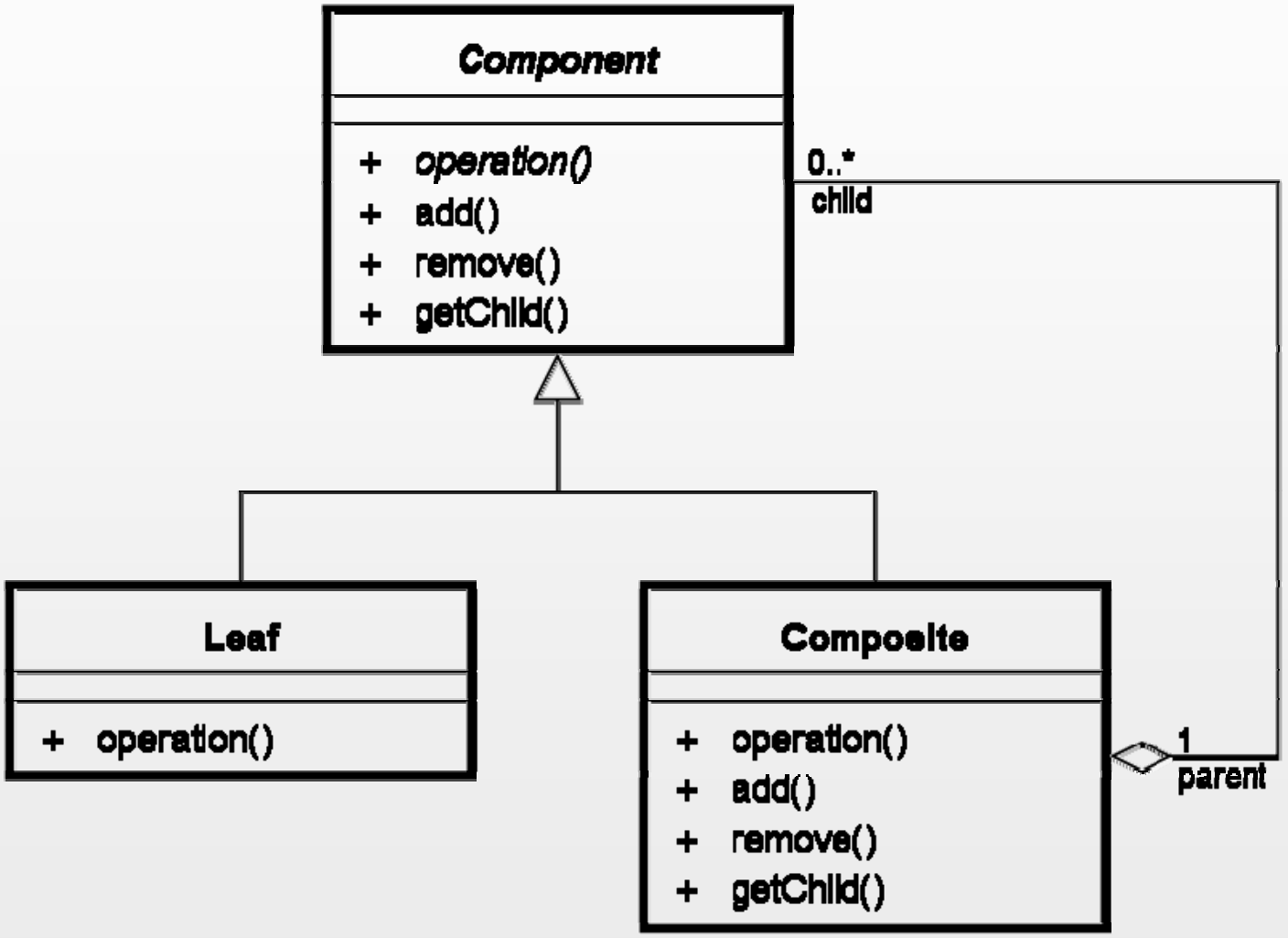
Diagrama de Actividad

- **Mezcla de los viejos diagramas de flujo con concurrencia basada en tokens, estilo redes de Petri.**
  - Ojo, es un diagrama de flujo *de control*, no de datos



Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems; **p. 28; B.P. Douglass; Addison Wesley; 2002**

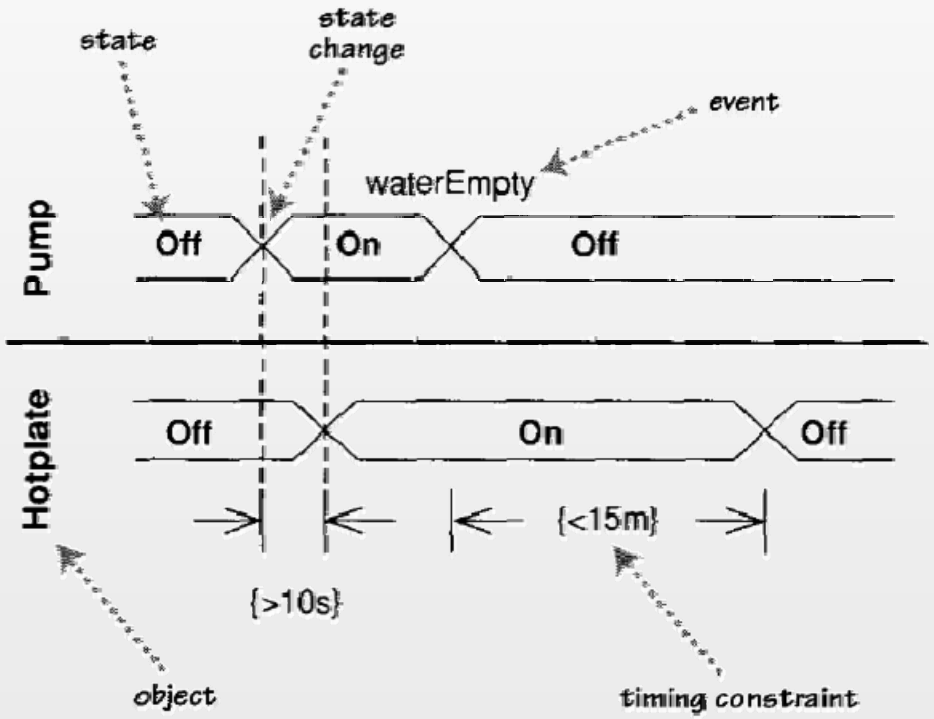
Diagrama de Clases



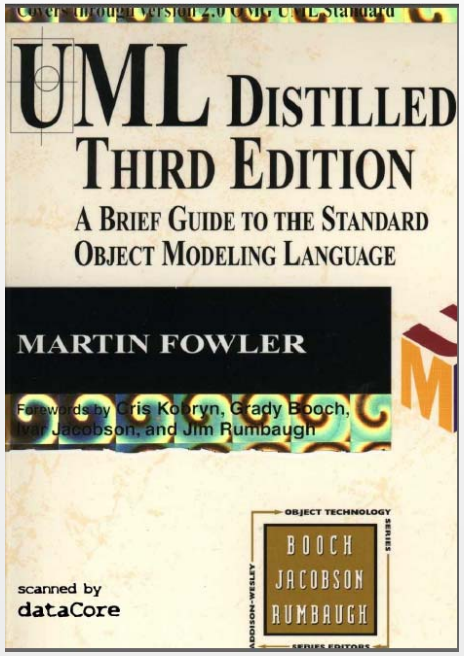
"Image:Composite UML class diagram.svg"*; Wikipedia*; **disponible el 29/11/08 en <http://commons.wikimedia.org/wiki/Image:Composite_UML_class_diagram.svg>**

Diagrama de Temporización

- **O** *timing diagram*
- **Como los que se usan en digitales**



Timing diagram showing states as areas



El diagrama se tomó de este libro, que ofrece un resumen detallado (160 páginas) del UML 2

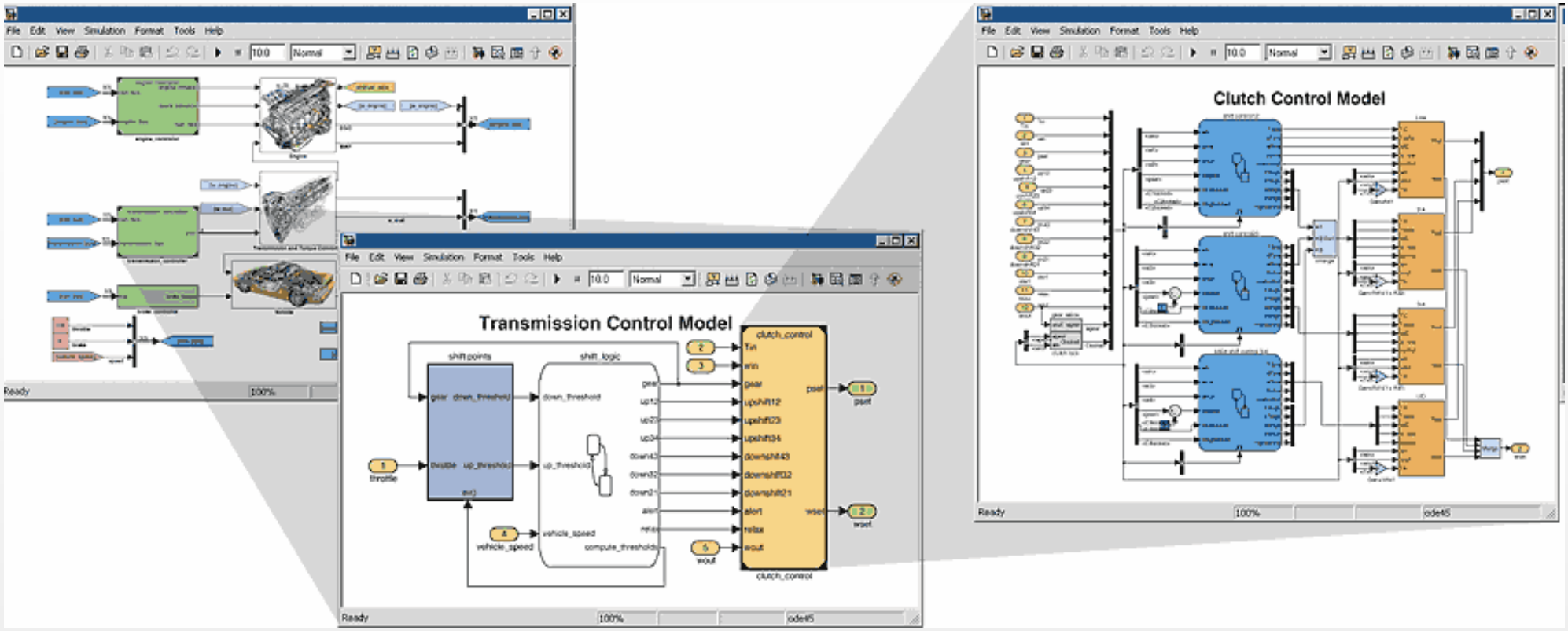
Otros Diagramas

- **Componentes, Paquetes, Instalación**
  - Para mostrar particiones lógicas y físicas del sistema.
- **Comunicaciones, Repaso de Interacciones**
  - Alternativas al diagrama de secuencia, para mostrar interacciones entre objetos.
- **Objetos**
  - Similar al de clases pero muestra objetos.
- **Estructura Compuesta**
  - Similar al de clases, incorpora jerarquías y categorizaciones.

Desarrollo Basado en Modelos

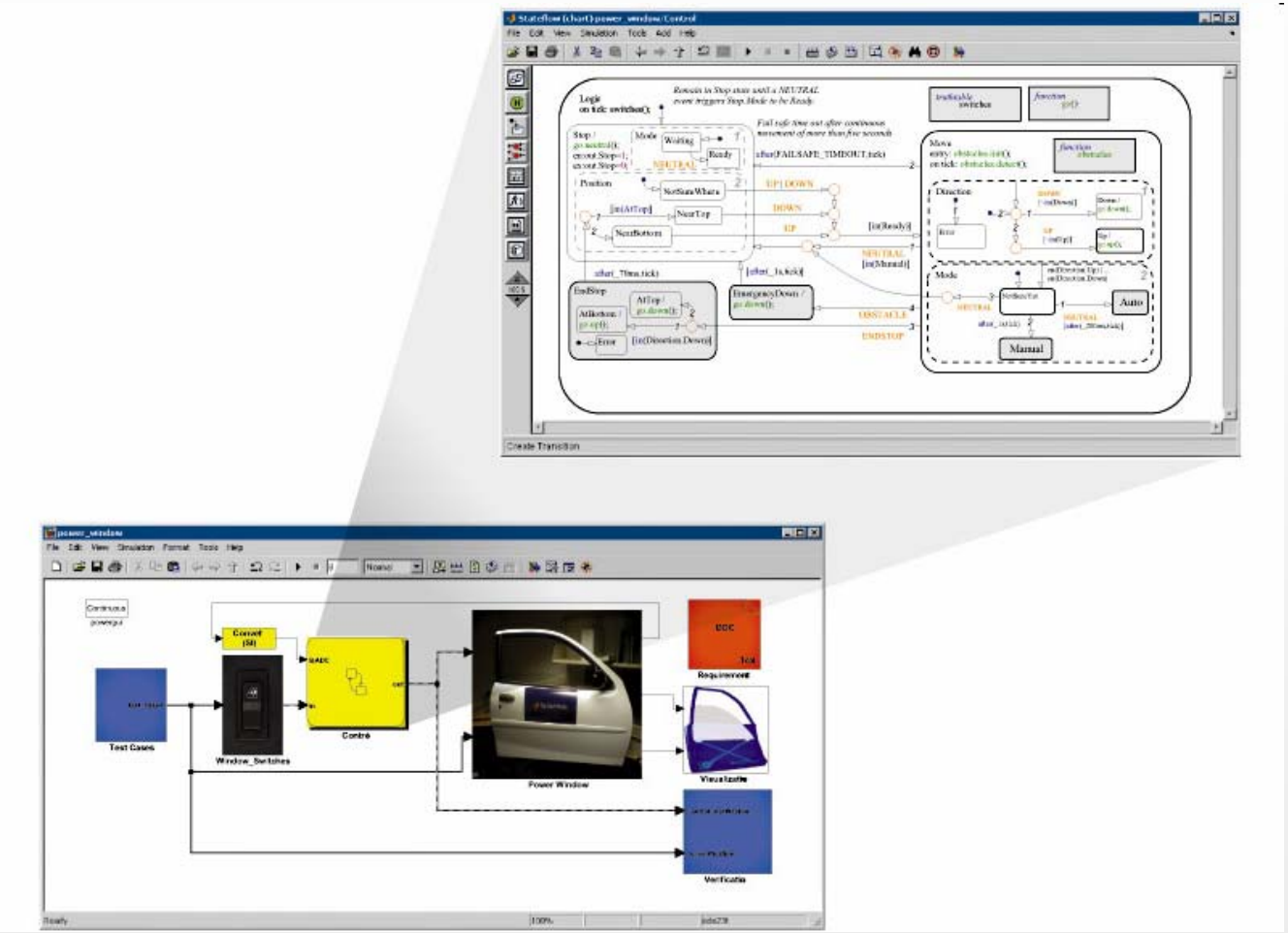
- **Significa que modelos ejecutables sean el "código fuente" principal de los desarrollos.**
- **Se lo llama Model-Driven Architecture (MDA) o Model-Driven Development (MDD)**
  - "There are many views and opinions about what MDA is and is not." (Alan Brown, Staff, IBM)
- **Tratan la provisión de herramientas basadas en modelos, para creación, transformación, testeo, análisis, simulación, ing. inversa, etc.**
  - Ejemplo de MDD para aplicaciones embebidas generales: **Telelogic Rhapsody** (de IBM)
- **A este tipo de herramientas antes se las categorizaba como *CASE* (Computer-Aided Software Engineering)**
  - El término CASE fue tan abusado que ahora se lo utiliza poco.

Simulink®



- **Es un entorno para modelado (gráfico) de los fabricantes de Matlab**
  - Pueden embeberse código Matlab en un modelo Simulink, y viceversa.
    - También puede embeberse código en C, Fortran o Ada
  - Puede usarse Matlab para analizar la salida de un modelo Simulink
- **Los modelos son ejecutables**
  - O sea que sirven para detectar y corregir fallas, temprano
    - Además de ser útiles para diseñar y documentar
- **Trabaja con diagramas de flujo de datos, en tiempo discreto o continuo**
  - Por lo tanto, es especialmente útil para DSP y control

Stateflow® (para Simulink)



- **Es un plug-in para poder incluir statecharts en los modelos que se diseñan con el Simulink**

Real-Time Workshop® (para Simulink)

- **Plug-in que "compila" el modelo en código C o C++ para microcontroladores**
  - Con llamadas al sistema operativo que se elija, o a ninguno
- **Genera código para**
  1. Inicialización
  2. Implementación de los algoritmos del modelo
  3. Instrumentación (para ajustar parámetros y "data logging")
- **Video de demo en:**
  http://www.mathworks.co.jp/products/demos/rtw/introduction/index.html
- **Está pensado también para simulación de tipo *hardware in the loop* (HIL)**
  - O sea, simular el *entorno* que interactúa con un sistema embebido

Fuente: embedded.com

Telelogic Rhapsody® (IBM)

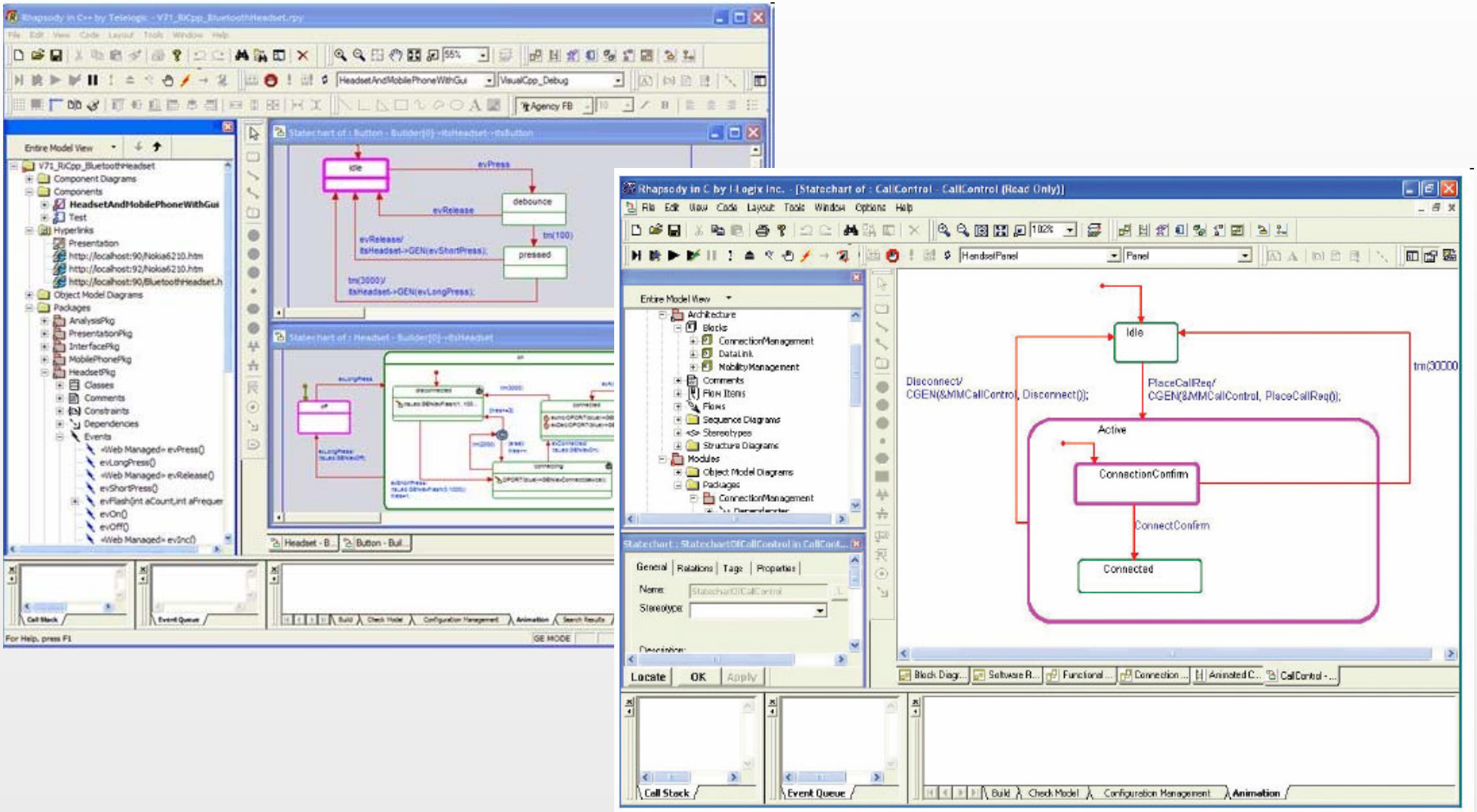
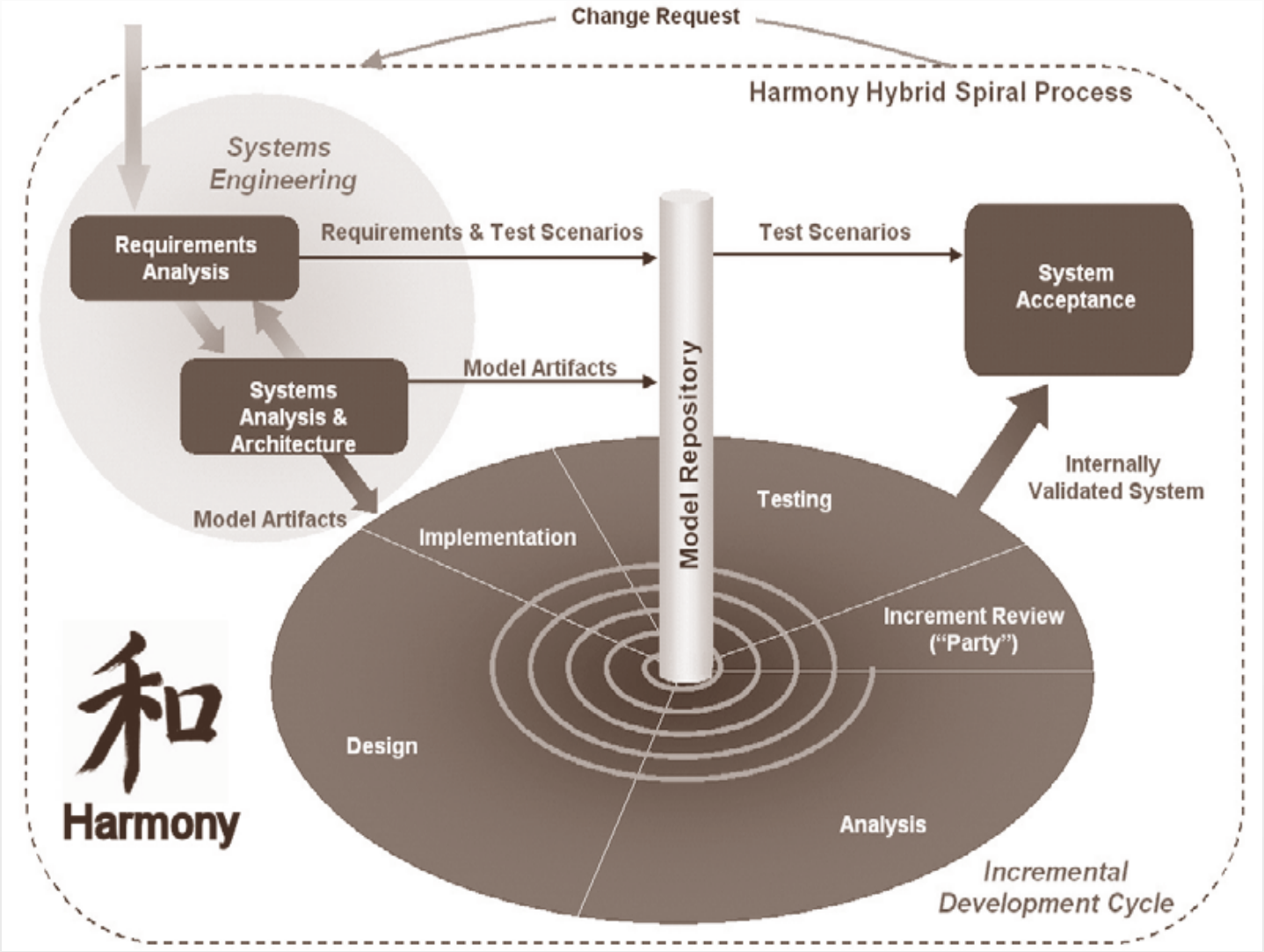


Figure 1 The recently introduced Version 7.0 Rhapsody from Telelogic generates C, C++, or Java code directly from a standard UML model.

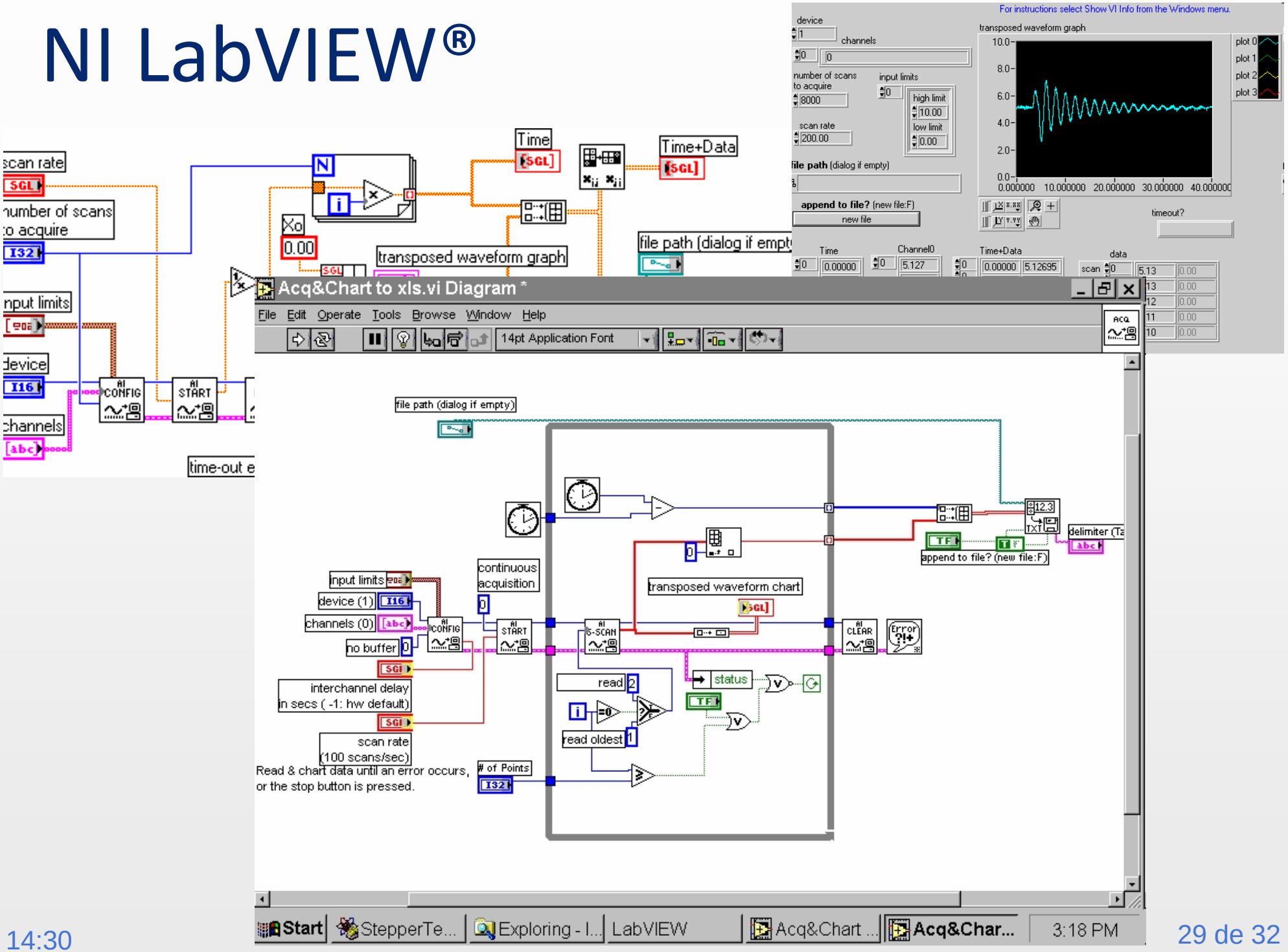
Metodología Asociada al Rhapsody

Harmony Process

- De I-Logix, que fue adquirida por Telelogic, que fue adquirida por IBM
- Emplea UML, y modelo de ciclo de vida que combina espiral y V



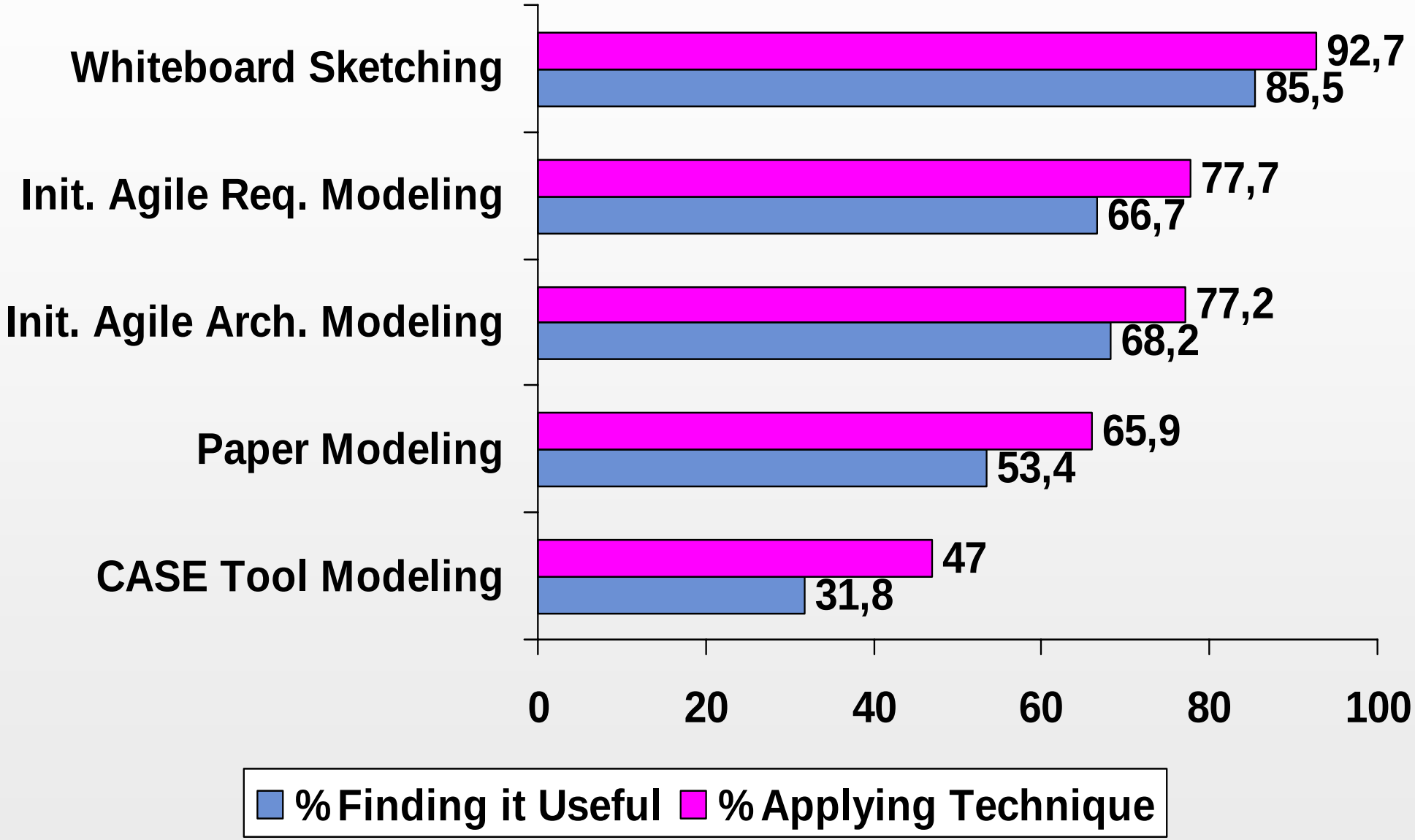
NI LabVIEW®


Acq&Chart to xls.vi Diagram *
File Edit Operate Tools Browse Window Help
14pt Application Font
transposed waveform graph
transposed waveform chart
file path (dialog if empty)
append to file? (new file:F)
continuous acquisition
input limits
device (1)
channels (0)
no buffer
interchannel delay in secs (-1: hw default)
scan rate (100 scans/sec)
# of Points
Read & chart data until an error occurs, or the stop button is pressed.
read
read oldest
status
Start
StepperTe...
Exploring - I...
LabVIEW
Acq&Chart ...
Acq&Char...
3:18 PM

Mejores Prácticas del *Agile Modeling*

- **“Initial Requirements Modeling”**
  - En la iteración 0, después del *release plan*, se empieza modelando el entorno y un planteamiento general de los requerimientos fundamentales del sistema y la interfaz al usuario
- **“Initial Architecture Modeling”**
  - En la iteración 0 se modela también, en líneas generales, la arquitectura pretendida, para empezar a identificar los módulos a desarrollar
- **“Just Barely Good Enough” (JBGE) artifacts**
  - No pasarse con el esfuerzo puesto en un modelo o documento
    - Ej.: si sólo lo queremos para entendernos con un colega, no hace falta pasarlo en la PC
    - Ej.: si sólo necesitamos analizar una parte, no hace falta modelar todo
- **“Model Storming”**
  - Cuando surge un problema para pensar, se hace una reunión rápida, se piensa entre todos usando modelos (generalmente sobre un pizarrón), y después cada uno sigue con lo suyo
- **Aprovechar el modelado, también, para**
  - La automatización de las pruebas de aceptación
  - Usarlo como documentación, reduciendo la documentación *ad hoc* y dejándola para el final
    - “La documentación es el diseño”
- **Para ver más: http://www.agilemodeling.com**

Survey Says: Agilists are Modeling



Para Terminar

- **El propósito de estas clases, es que**
  - Incluyan lenguajes de modelado entre los que usan profesionalmente
    - Para sumarlo al castellano, el lenguaje matemático, C, etc.
  - Identifiquen situaciones en donde son útiles
    - Ej. Para plantear problemas confiablemente, comunicarse con colegas y clientes, documentar, debuggear temprano, simulación de HIL, MDD, etc.
  - Facilitar la profundización de estos temas
- **¿Preguntas? ¿Comentarios?**
- **Hay un video con una demo del Telelogic Rhapsody**
  - “RhapsodyCDemo.wmv”; Telelogic (IBM)
- **Hay una presentación con una introducción al LabView de National Instruments**
  - “LabVIEW Introduction.ppt”; National Instruments
- **¡Gracias!**