Metodologías, procesos y entornos para sistemas de tiempo real

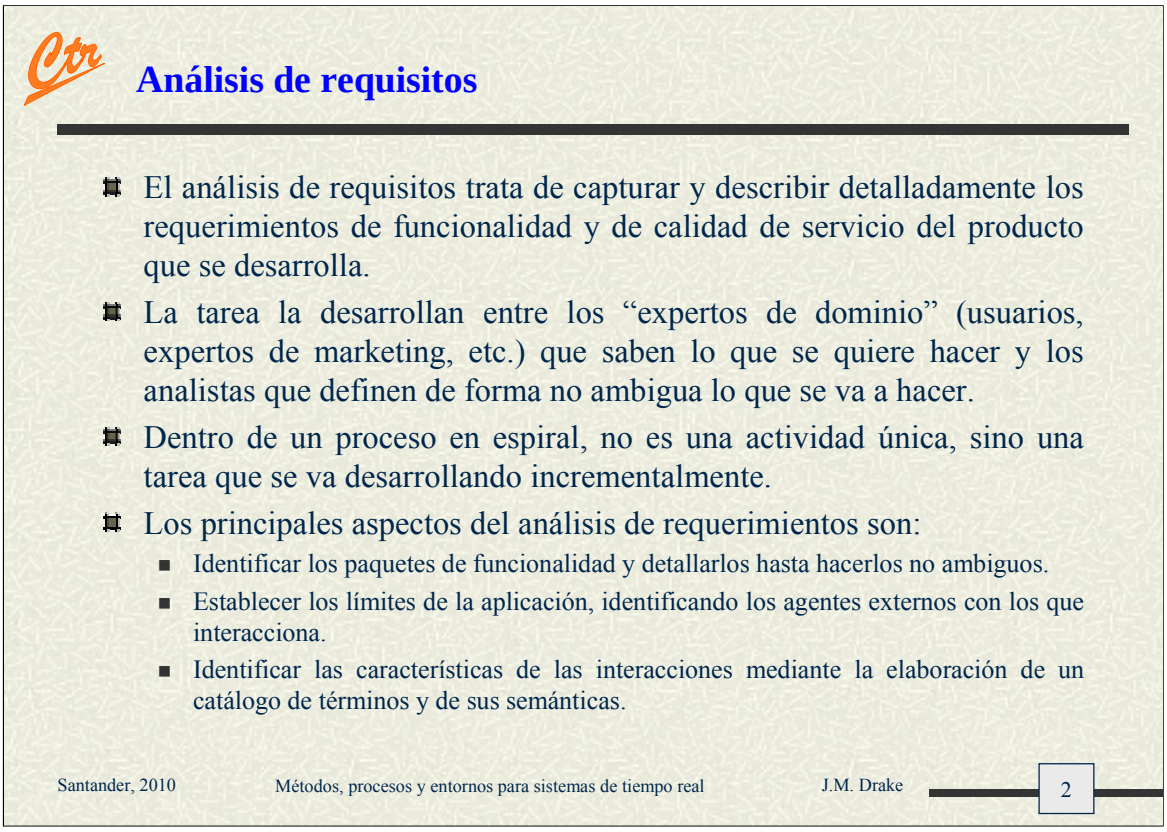
Master de Computación

Especificación de una aplicación

José M. Drake
Computadores y Tiempo Real

Santander, 2012

1



Para construir *algo* primero debe entenderse lo que debe ser ese *algo*. El proceso de entender y documentar una aplicación software se llama "Análisis de requisitos". En general los requisitos expresan *qué* se supone debe hacer una aplicación y no intentan expresar *como* logra estas funciones.

El análisis inicial de un sistema debe tratar de descubrir los requerimientos del producto final que se desarrolla en detalle.

Unos de los principales objetivos de UML es hacer que este análisis sea lo suficientemente intuitivo para que los clientes y expertos en el dominio que solicitan el producto puedan comprenderlo, y lo suficientemente formal y riguroso para que se establezca una formulación no ambigua que pueda ser utilizada por los técnicos que la desarrollan.

Los aspectos básicos que deben tratarse en esta fase son:

•Determinar los paquetes de funcionalidad y de la calidad de servicio del producto, formulados de una forma independiente de su implementación, y refinar y detallar estas especificaciones hasta que den lugar a una especificación no ambigua del producto que se desarrolla.

•Identificar los actores externos al sistema que interactúan con la aplicación de forma relevante.

•Identificar la semántica y las características de los mensajes que intercambian los actores con el sistema que se desarrolla.

•Refinar los protocolos de interacción que usan los actores para llevar a cabo las diferentes transacciones que se pueden realizar con el sistema.

El análisis de requisitos es una necesidad, no un un lujo. Para apoyarlo considérese su efecto sobre las pruebas del producto concluido. Si alguien le proporciona una caja negra con un cable rojo, rosa y morado que sale de ella, sería imposible probarlo. No se sabe que hace, para que sirve.

Proceso de análisis de requisitos

1. Identificar al cliente.
2. Entrevistar al cliente.
   - Identificar deseos y necesidades.
   - Utilizar las herramientas de expresión de requisitos (las ofrecidas por UML).
   - Bosquejar las interfaces de usuario (protocolos y GUIs)
   - Identificar las plataformas hardware que debe soportar el software.
3. Elaborar un documento de los requisitos de usuario (Debe validarse con el cliente)
4. Inspeccionar los requisitos de usuario.
5. Elaborar los requisitos detallados mediante documentos gráficos y textuales.

Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 3

El proceso de análisis de requisitos requiere diferentes actividades de alto nivel y que son desarrollados por múltiples agentes (usuarios, expertos de dominio, expertos de marketing, programadores, etc.)

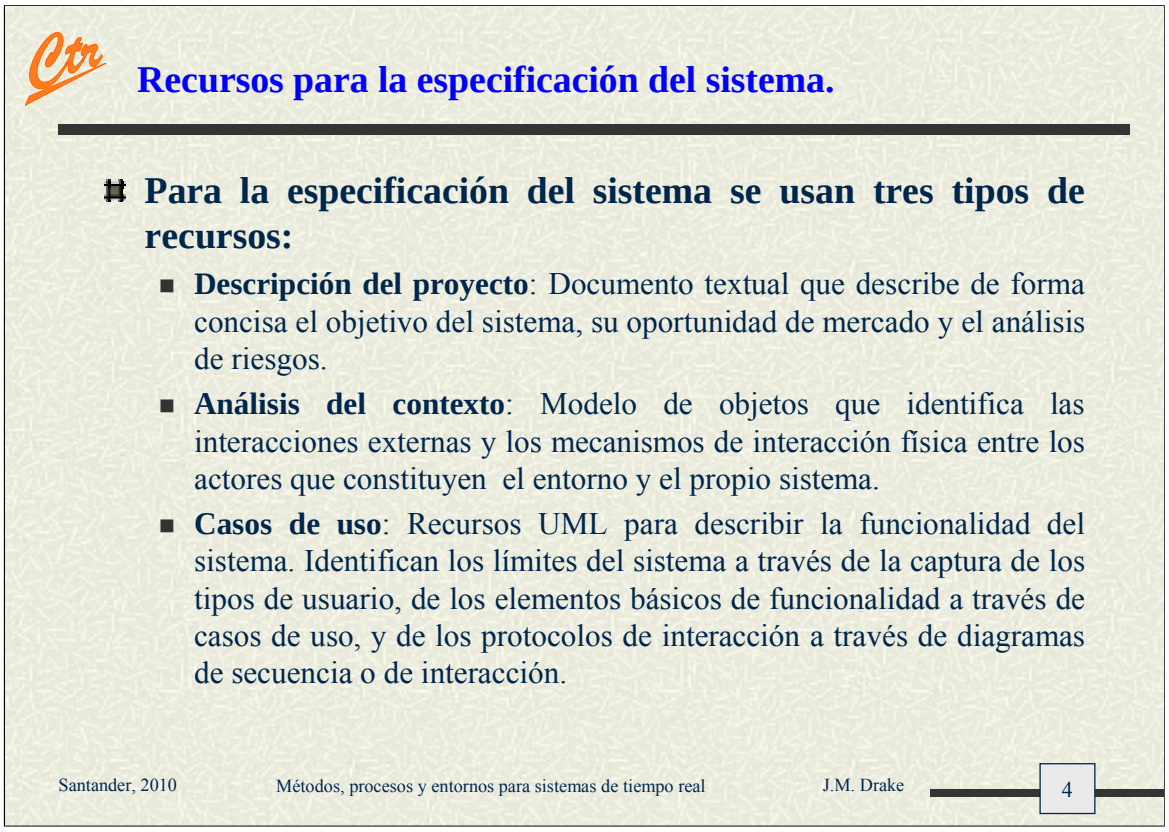
Para la formulación de las especificaciones existen diferentes estándar, el mas conocido es el estándar ANSI, IEEE 830-1993.

Para todas las etapas se pueden utilizar métricas tales como:

- Tiempo dedicado a su análisis.
- Cantidad producida (páginas de requisitos, minutos de interacción con el cliente, etc.)
- Calidad deducida de la autoevaluación (Tasas de defectos en las inspecciones).

Estándar ANSI IEEE 830-1993

1. Introducción
   - Propósito.
   - Definiciones acrónimos y abreviaturas
   - Panorama
   - Alcance
   - Referencias
2. Descripción general
   - Perspectiva del producto:(Interfaces del sistema, del usuario, hardware, software, de comunicación, restricciones de memoria, operaciones, requisitos de adaptación)
   - Funciones del producto
   - Restricciones
   - Distribución de requisitos
   - Características del usuario
   - Suposiciones y dependencias
3. Requisitos específicos (Se desarrollará mas adelante)
4. Información de apoyo.



Los sistemas interactúan con su entorno externo (operadores, usuarios, otros sistemas, dispositivos, etc.) y la funcionalidad básica que tienen que ofrecer debe formularse en función de este contexto y con independencia de la forma en que se construyen internamente.
Existen tres vías que pueden utilizarse para realizar la formalización de los requerimientos:
•Descripción del proyecto: Es un paso previo que aunque es obvio tiene una gran importancia, y consiste en generar un documento que de forma concisa resuma la información inicial relativa al proyecto que se inicia. En él debe incluirse la naturaleza y objetivo del proyecto, las características mas relevantes, su oportunidad de mercado, y un análisis de los riesgos que conlleva. Debe ser un documento breve con solo dos o tres páginas, pero que establece un punto de arranque en el que los diferentes responsables de su ejecución (clientes, expertos de dominio y desarrolladores) tienen el mismo concepto sobre lo que se desarrolla.
•Análisis del contexto: Trata de especificar la funcionalidad del sistema a través de la descripción de las interacciones que se pueden producir entre el sistema y el entorno externo. Se formula como diagramas de objetos en los que el sistema aparece como una caja negra sobre la que se identifican los elementos de interacción (sensores y actuadores) y también se identifican los actores externos que interactúan con él, así como los tipos de mensajes que se producen definiendo su semántica y la información que transmiten.
•Casos de uso: Es el recurso específico de UML para describir la funcionalidad y las características de calidad de servicio del sistema. Se basa en identificar los límites del sistema a través de la captura de los actores, de los elementos básicos de funcionalidad a través de casos de uso, y de los protocolos de interacción a través de diagramas de secuencia o de interacción.

Descripción del proyecto

- Un proyecto que se inicia siempre debe partir de un documento breve que lo describa y plantee sus principales características.
- Sirve de contrato para que todos los que participan en su promoción tengan el mismo concepto sobre su contenido y objetivos.
- El documento debe ser breve (2 o 3 páginas) y debe ser realizado por una o dos personas y aceptado por todos los participantes.
  - Usuarios del producto
  - Responsable de la empresa
  - Programadores
  - Los que encargan y financian el producto
  - Administradores
- El documento debe contener:
  - La naturaleza y objetivos del producto.
  - Las características más relevantes.
  - La oportunidad de mercado del producto.
  - Análisis de riesgos para el desarrollo del proyecto.

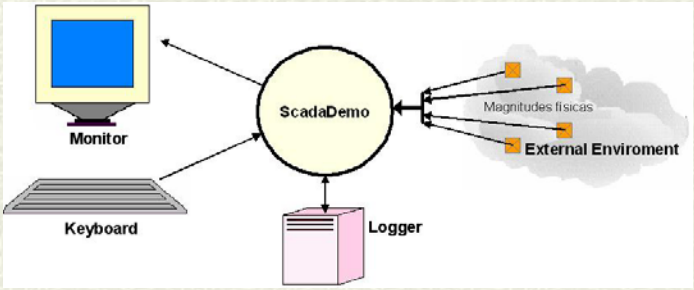
Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 5

A las personas que tienen intereses en el producto que se desarrolla, le denominamos *interesados (stakeholders):*
•Usuarios del producto: son aquellos que van a utilizar directamente el producto como operarios.
•Los que encargan o financian el producto
•Los directivos de la empresa en que se desarrolla el producto.
•Los administradores.
•Los desarrolladores siempre que utilicen la aplicación como forma de actualizar su tecnología.

Cada requisito debería ser:
•Expresado en el lenguaje del usuario.
•Incluido en una lista organizada que facilite su localización y acceso sea sencillo.
• Estar numerado y ser referenciado en el código.
•Ser verificable de forma aislada e ir acompañado de pruebas que posibiliten su verificación.

Descripción de sistema SCADA

La aplicación *ScadaDemo* supervisa un conjunto de hasta 16 magnitudes físicas que son leídas a través de un conjunto de tarjetas de entrada/salida dotadas de conversor A/D e instaladas en diferentes procesadores accesibles mediante la red.



Ejemplo de descripción del producto.

Supervisar una magnitud significa:
•Leer la magnitud a una determinada frecuencia configurable, y almacenar temporalmente los valores leídos en un buffer.
•Calcular el valor promedio de los valores de la magnitud leída, dentro de una ventana temporal establecida para cada magnitud.
•Almacenar en un logger permanente los valores promedios calculados.
•Comprobar si cada valor promedio calculado se encuentra dentro de un rango de alarma definido mediante dos valores de umbral definidos para cada magnitud, y mostrar en la consola un evento de alarma si el resultado de la comprobación es cierta.

Las magnitudes que son supervisadas y las características de la supervisión de cada una de ellas se leen de un fichero de configuración en la fase de arranque de la aplicación.

El contenido del fichero de configuración es:
•Una lista con una entrada por cada magnitud que va a ser supervisada.
•Por cada magnitud se especifica, El IP del computador en que está la tarjeta, el puerto por el que se comunica con ella, la línea analógica de la que se lee (tarjeta/línea), una etiqueta, el periodo con el que se lee T_{Sampl}, el periodo con el que se registra T_{Reg}, los factores de escala representación/valor físico y los valores que definen el rango de alarma.

Ejemplo de descripción del producto.

La interacción entre el operador y la aplicación se realiza a través de un terminal dotado de teclado y monitor.
Las opciones de interacción que hay son:

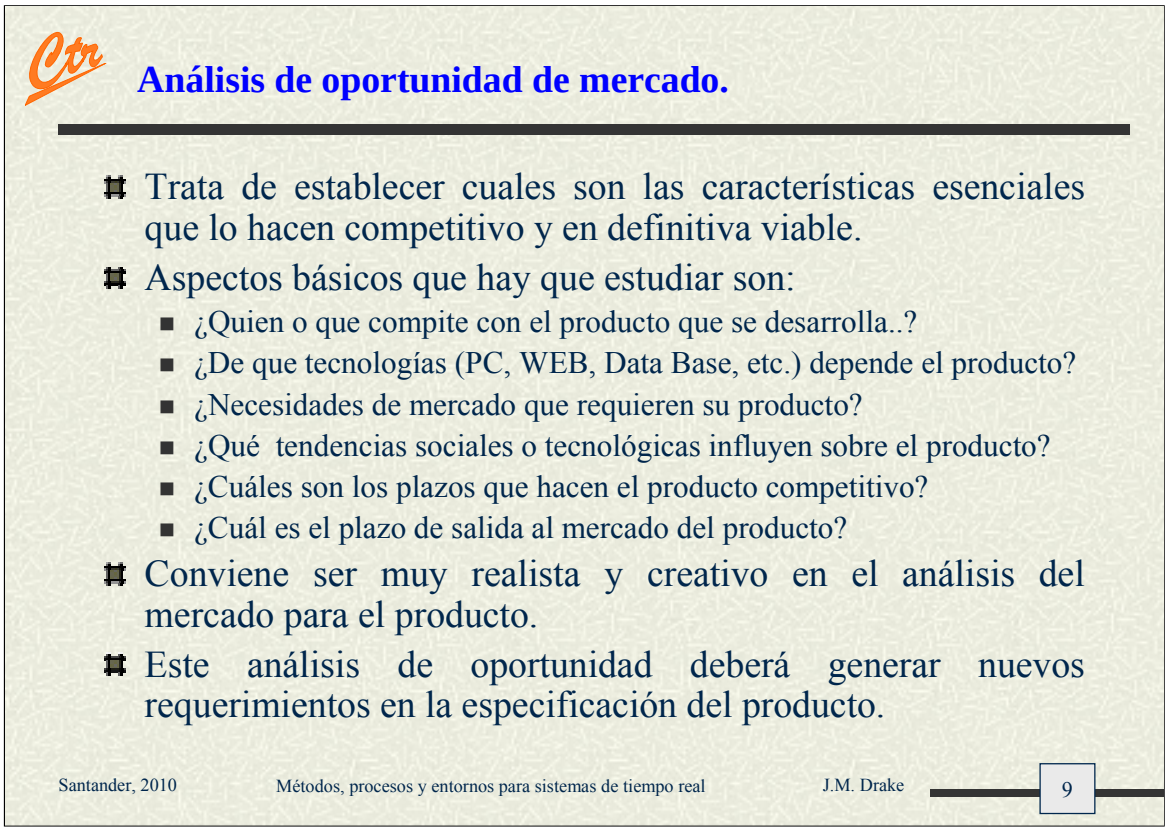
- •En el monitor se visualiza un mensaje cada vez que se registra en el logger un valor de alguna magnitud.
- •En el monitor se registra un mensaje cada vez que

•El operador pulsa la letra "T" para terminar la aplicación.
•El operador pulsa la letra "C" para recargar la aplicación con un nuevo fichero de configuración

Los requisitos de tiempo real de la aplicación, son:

- •Los valores de las magnitudes se adquieren con un jitter máximo de 5 ms.
- •El número de muestras de cada magnitud que se promedian antes de ser registrados son siempre el mismo: parte entera de T_{Reg}/T_{sampl},
- •Todos los valores que se registran deben ser visualizados en el monitor, al menos durante un tiempo igual a $T_{Reg}/2$

•Todos los valores que se registran deben ser salvados persistentemente en la base de datos externas.



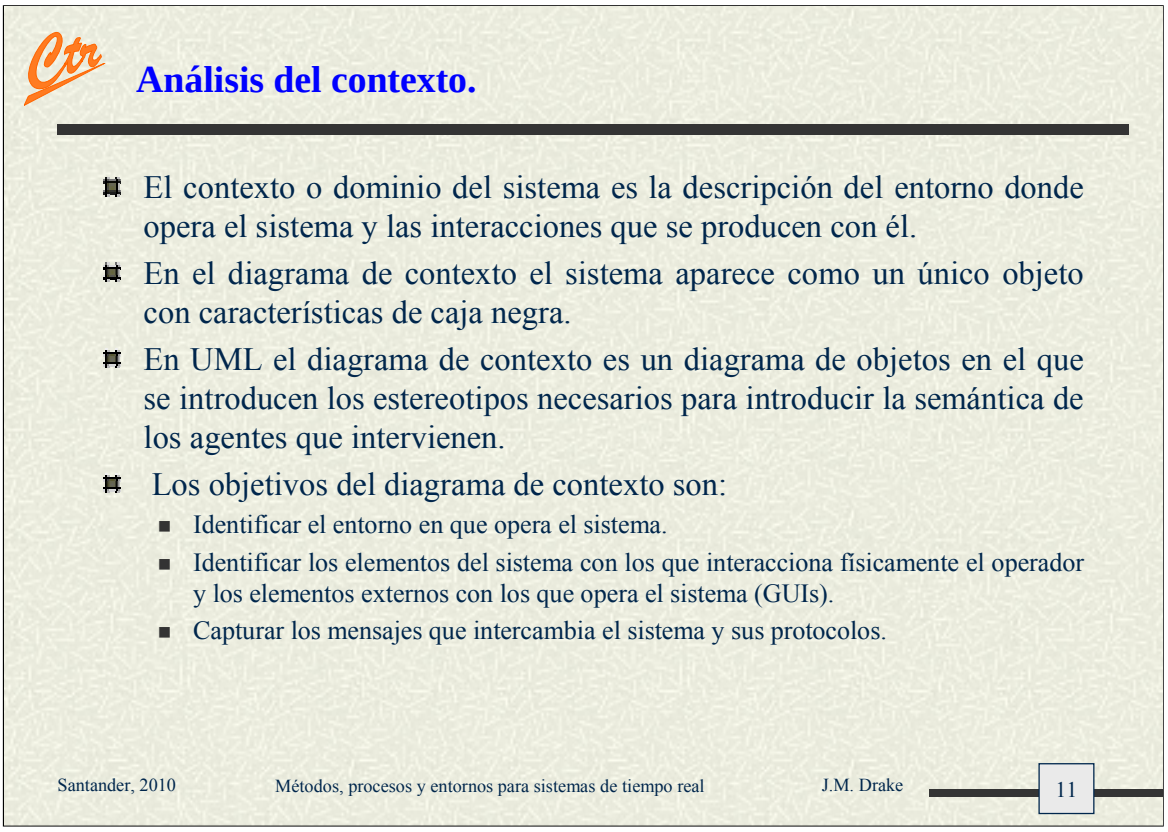
El proceso de análisis de requisitos debe contener un análisis de factibilidad y un análisis de oportunidad de mercado. El resultado debe ser un informe que recomienda si es conveniente llevar a cabo la ingeniería de requerimientos y el proceso de desarrollo del sistema.
Un estudio de factibilidad es un estudio corto, que responde a si el producto contribuye a los objetivos de negocio de la empresa. Debe responder a varias preguntas:
•¿Corresponde el producto a los objetivos generales de la empresa?
•¿Se puede utilizar el producto haciendo uso de la tecnología actual y con las restricciones de tiempo y costo?
•¿Puede integrarse el sistema a otros que existen en la organización?
Un estudio de factibilidad comprende la evaluación y recolección de la información y a la elaboración de los informes.

Análisis de riesgos.

- El análisis de riesgos trata de analizar las características o cosas que pueden inducir a errores y fallos.
- Su objetivo no es siempre solucionarlos sino estar preparados para cuando aparezcan
- Aspectos que deben considerarse:
  - Relativo a las personas: Falta de experiencia. Inexperiencia con la tecnología, incapacidad de encontrar las personas.
  - Relativos al sistema: Número de usuarios, sistemas de los que se dependen (Software, bases de datos,etc.), posibilidades de fallos.
  - Relativos a los recursos: Demasiados usuarios, fallos en los suministradores de los que se depende.
  - Relativos a la tecnología:Tendencia y cambios de la tecnología.
  - Relativos a la empresa: Ausencia de interés de los clientes, crecimiento excesivamente rápido.
- Es interesante que se analice el caso de éxito excesivo.

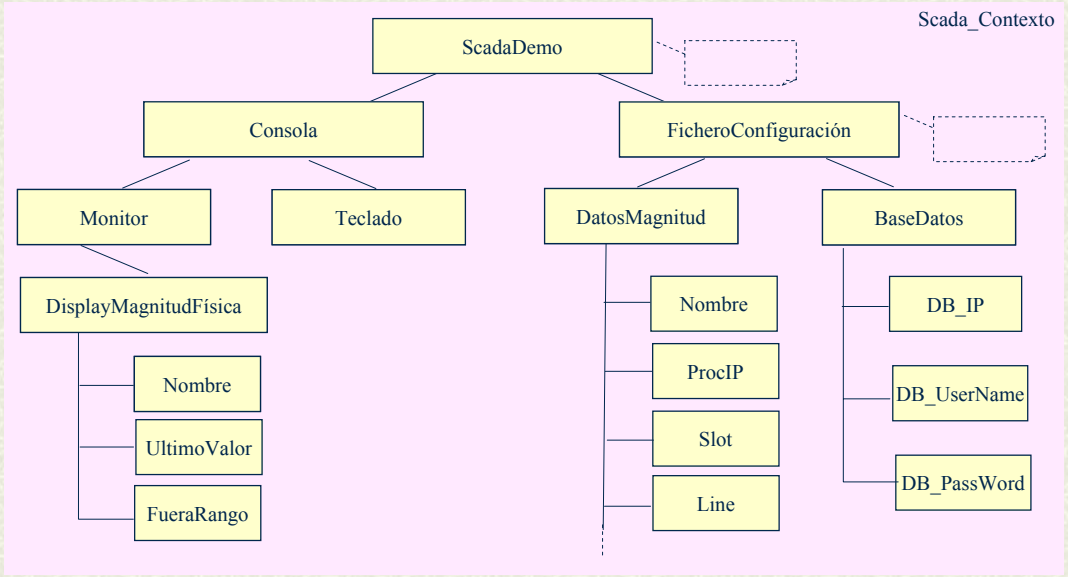
Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 10

Se necesita considerar los factores de riesgo en el proyecto. Se necesita incluir las cosas que pueden conducir a equivocaciones. Describir solo las características que nos gustarían que ocurrieran nos llevan directamente al desastre. Es muy importante tener en cuenta que cosas pueden ir mal y planificar el desarrollo del proyecto para que no ocurran.
También hay que prever la posibilidad de que el proyecto tenga un éxito salvaje que nos puede sobrepasar y conducir también al desastre.

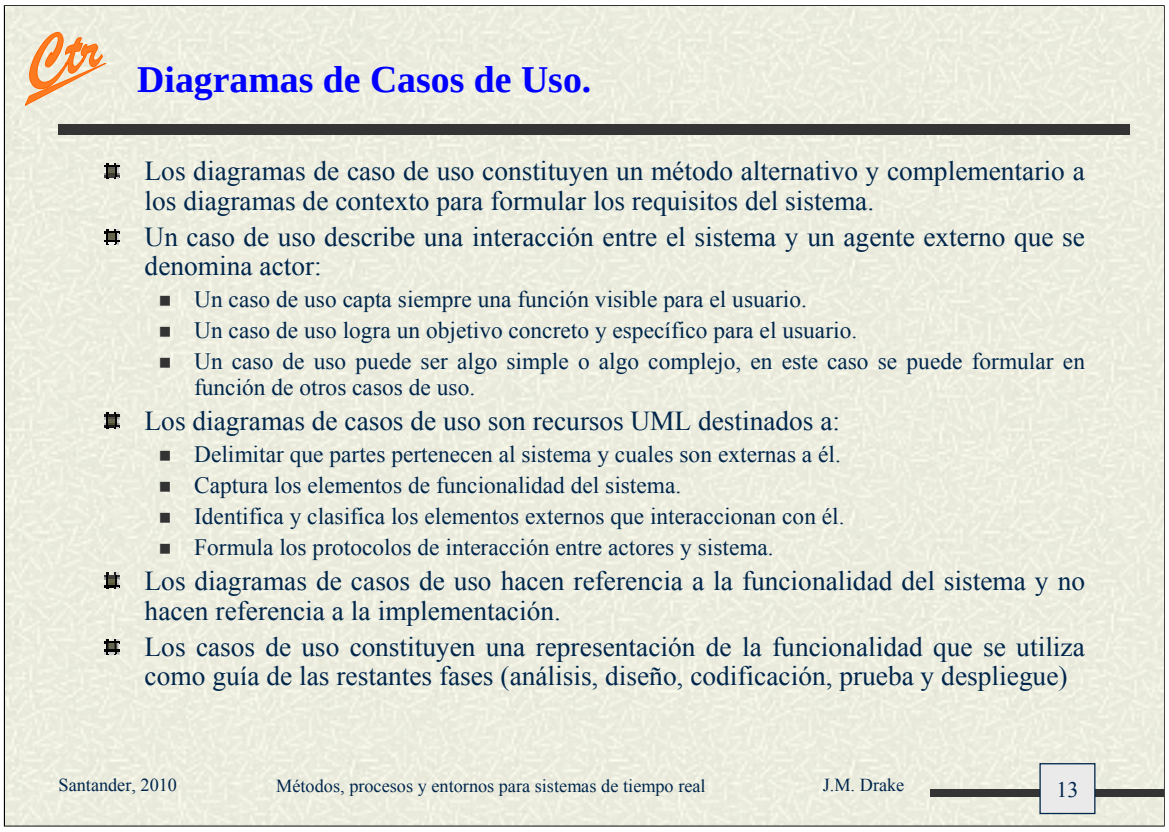


El contexto del sistema es un mapa del mundo que es de interés para el sistema. Esta herramienta es directamente heredada de las estrategias estructuradas.
El término sistema representa a todo aquello que es objeto del diseño y se muestra con características de caja negra. Incluye todo el sistema bajo desarrollo: el software, el hardware eléctrico y mecánico. En el diagrama de contexto se representa como un simple elemento (el sistema), rodeado de múltiples elementos que constituyen el entorno donde opera.
UML no tiene un recurso específico para representar los diagramas de contexto, por ello, se utiliza un diagrama de objetos o diagrama de colaboración, en el que la semántica de los objetos se explicita mediante estereotipos.
El beneficio de utilizar los diagramas de contexto es que se captura el contexto en que opera el sistema, incluyendo los actores con los que interacciona el sistema. Así mismo también ayuda a identificar y caracterizar los mensajes y eventos que fluyen entre el sistema y su entorno.

Diagrama de contexto de la aplicación ScadaDemo

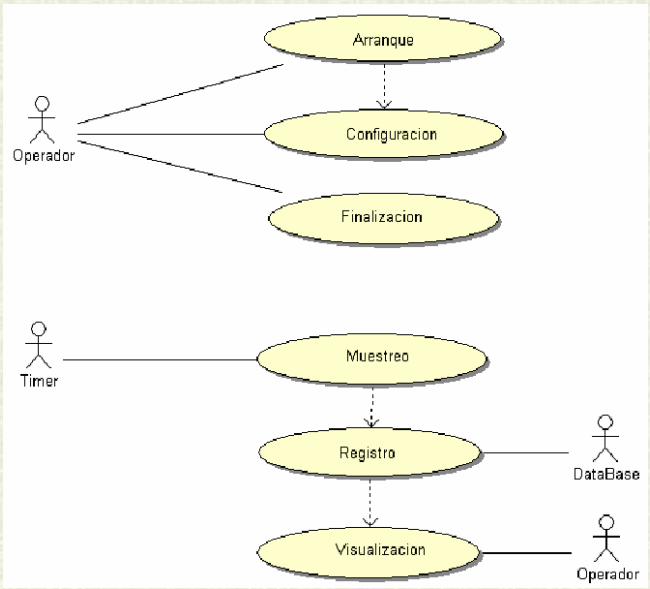


Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 12



Los diagramas de casos de uso son un método alternativo y complementario a los diagramas de contexto como medio de especificar los requisitos de una aplicación software.
Jacobson creó la idea de los casos de uso al observar que, a pesar del gran número de ejecuciones potenciales, muchas aplicaciones están concebidas en términos de un número relativamente pequeño de interacciones típicas.
Muestran los tipos básicos de interacción entre el sistema y los elementos del entorno que operan con él. Los diagramas de casos de uso proporcionan un recurso alternativo bien para verificar el diagrama de contexto o de ayuda para construirlo.
Los casos de uso capturan una vista general de la funcionalidad del sistema con un método muy adecuado para ser interpretado por personas no técnicas como son los usuarios y los expertos de dominio. Suelen interpretarse como una guía de los escenarios de uso del sistema sobre los que se especifican los requerimientos del sistema.
Los casos de uso son también un método de descomposición de la funcionalidad del sistema en elementos de funcionalidad más básica, ya que UML proporcionan una semántica para expresar los casos de usos mas generales en función de casos de usos mas simples.
Los casos de uso son contenedores de la información que describen los requerimientos del sistema. Estos se pueden formular con diferentes niveles de detalles, mas cualitativos para el uso de los expertos de dominio y mas detallado y formales para el uso de los analistas de la aplicación.

Elementos de los diagramas de casos de uso UML.



Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 14

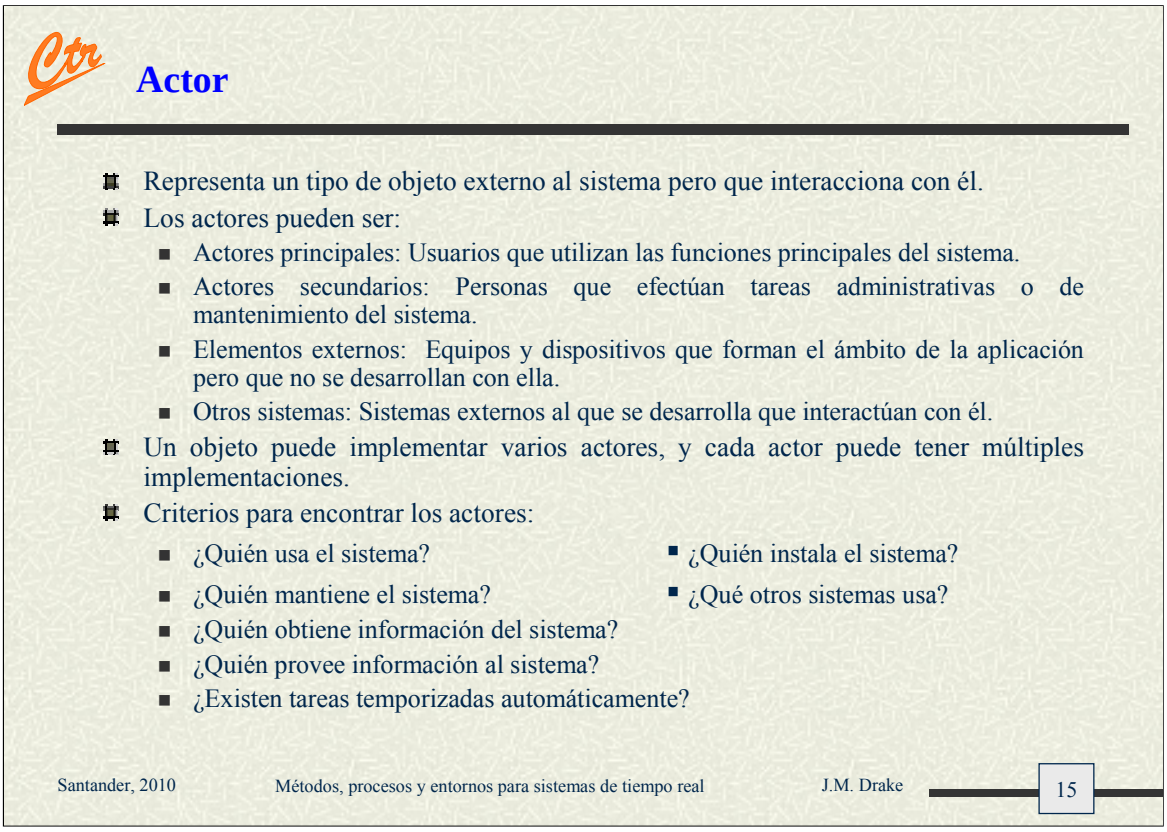
El modelo de casos de uso de una aplicación se compone de actores, el sistema (como una caja negra) y los propios casos de uso. La funcionalidad de un sistema se formula examinando las necesidades funcionales que requiere del sistema de cada actor, expresadas en forma de familias de interacciones que se incluyen en un caso de uso.

Un actor representa un papel interpretado por una persona u otro sistema que interacciona con el sistema que se describe. La misma persona física puede representar diferentes papeles y así mismo, muchas personas pueden ser representadas por un mismo actor. Los actores deben ser descritos de una forma clara mediante un texto de sólo algunas líneas.

Los casos de uso se determinan observando y precisando, actor por actor, las secuencias de interacción (que llamaremos escenarios) que llevan a cabo con el sistema. Los casos de uso agrupan a familias de escenarios que desempeñan un mismo papel funcional visto desde el usuario. Los casos de uso son abstracciones del diálogo entre los actores y el sistema: describen interacciones potenciales, sin entrar en los detalles de cada usuario.

Los escenarios son instancias de los casos de uso. Los escenarios representan secuencias de mensajes entre objetos que colaboran para producir algún tipo de comportamiento del sistema. Los escenarios constituyen un método muy intuitivo de representar los detalles de los requerimiento, en particular los que implican restricciones temporales.

Los enlaces entre actores y casos de uso representan los protocolos, que constituyen el hilo de conductor de la secuencia de eventos que intercambian ambos.



Los actores son cualquier cosa que interacciona el sistema que se desarrolla, por ejemplo, personas, otros software, hardware, dispositivos, redes, almacenes de datos, etc.. Cada actor define un particular "role". Cada entidad externa al sistema puede ser representado por uno o mas actores. Así, una persona física puede ser representada por varios actores , debido a que la persona juega diferentes papeles con relación al sistema. O varios objetos físicos pueden estar representados por un mismo actor, porque ellos mantienen una misma interacción en relación con el sistema.

Existen cuatro grandes tipos de actores:

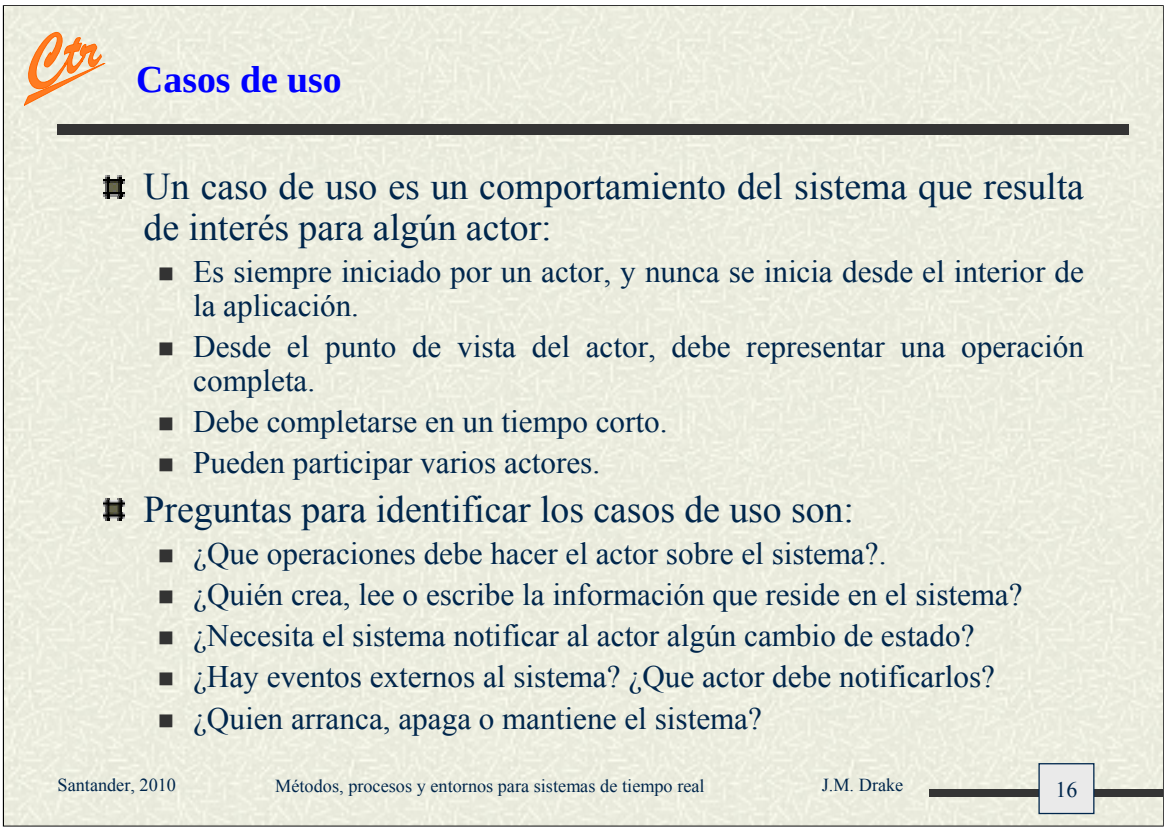
•Los actores principales: Agrupan a alas personas que utilizan las funciones principales del sistema. En el caso del software de un cajero son los clientes que hacen uso de él.

•Los actores secundarios: Son las personas que hacen tareas administrativas o de mantenimiento. En el ejemplo de un cajero es p.e. la persona que recarga el cajero.

•Los Elementos externos: Agrupa a los dispositivos que forman parte del ámbito de la aplicación y que deben ser utilizado. En el ejemplo del cajero pueden ser p.e. la impresora.

•Otros sistemas: Agrupa a los sistemas externos con lo que el sistema debe interactuar. En el ejemplo del cajero, un actor de este tipo puede ser el sistema del sistema bancario.

Los actores son siempre externos al sistema. Para tratar de localizar los actores, debemos buscar categorías de cosas que aparezcan como consecuencia de las siguientes preguntas: ¿Quien usa el sistema? ¿Quién instala, arranca y apaga el sistema? ¿Quien mantiene al sistema? ¿Que otros sistemas usa el sistema? ¿Quién obtiene información del sistema? ¿Quién provee información al sistema? ¿Ocurre algo de forma automática en tiempos prefijados?.



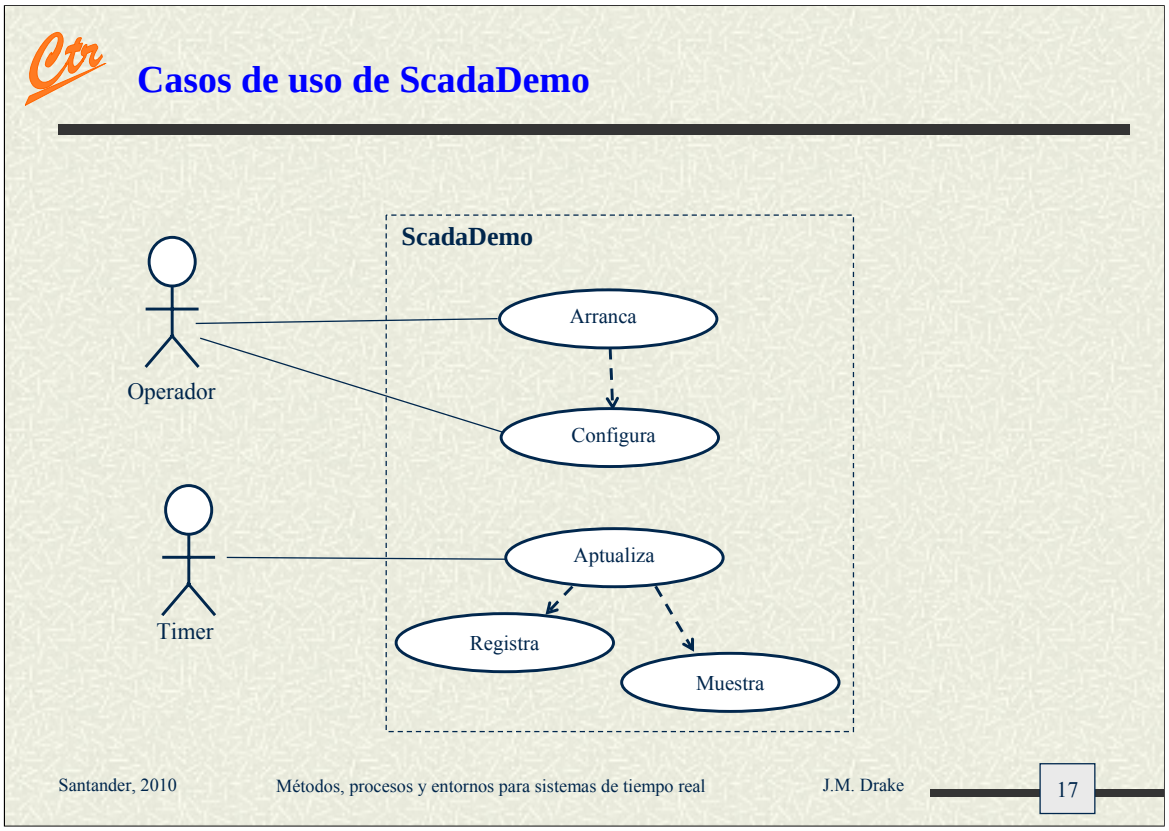
Un caso de uso es un comportamiento del sistema que produce un resultado de interés para algún actor. Los casos de uso describen cosas que los actores quieren que el sistema haga. Un caso de uso debe ser una tarea completa desde el punto de vista del actor, y debe corresponder a una tarea que se realiza en un tiempo relativamente breve,especialmente si debe ser realizado por múltiples actores. Cuando estas condiciones no se cumplen, es mejor definir varios casos de uso independientes.

Un caso de uso describe no solo una funcionalidad o una motivación, sino también una interacción entre un actor y el sistema bajo la forma de un flujo de eventos. La descripción que proporciona el caso de uso se refiere a lo que se espera que la interacción realice y no a como lo realiza.

Un caso de uso debe ser simple, y en caso contrario, se le debe fragmentar.

En UML los casos de uso son siempre iniciados por un actor, y no deben iniciarse espontáneamente desde el interior del sistema.

Es importante no solo considerar los casos de uso que describen la funcionalidad básica sino también considerar aquellos que describen las tareas de mantenimiento como, quien y como se arranca el sistema, se apaga, verifica el correcto funcionamiento de la instalación, etc. Todas estas funciones requieren prever una funcionalidad adicional que deben modelarse como casos de uso.



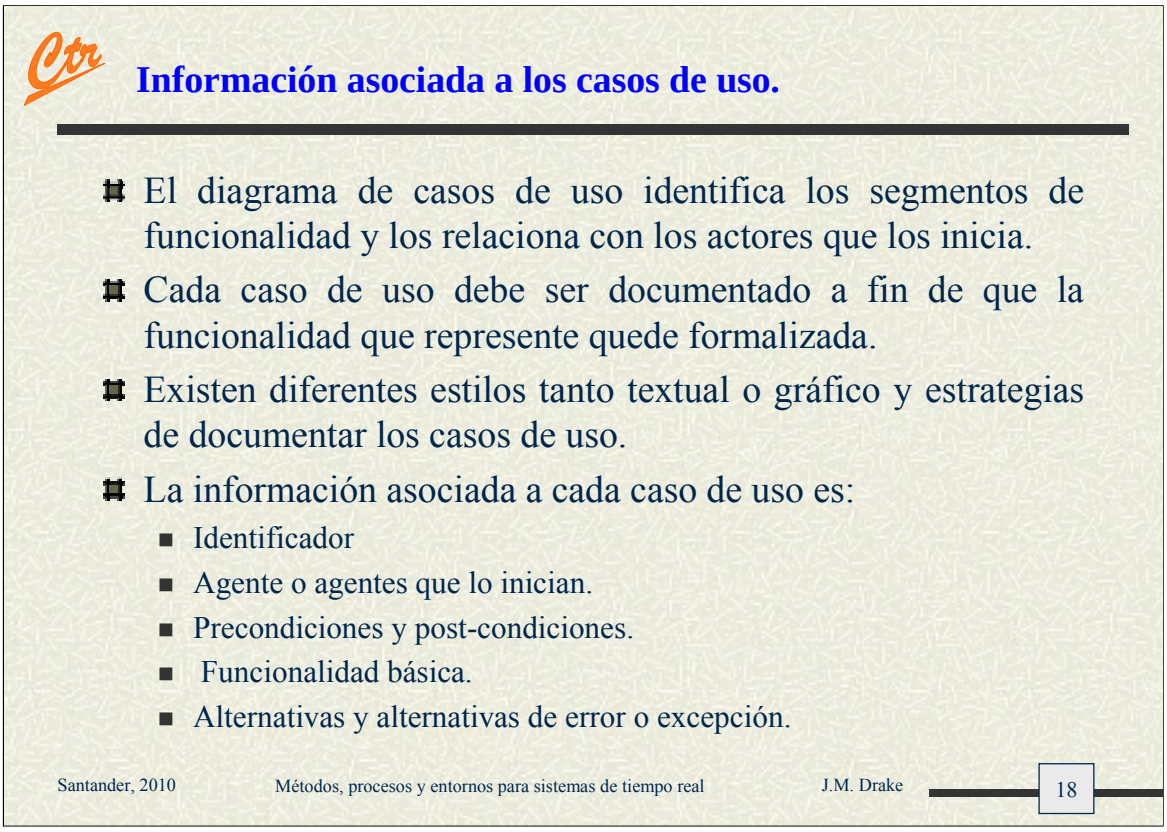
El modelo de casos de uso de una aplicación se compone de actores, el sistema (como una caja negra) y los propios casos de uso. La funcionalidad de un sistema se formula examinando las necesidades funcionales que requiere del sistema de cada actor, expresadas en forma de familias de interacciones que se incluyen en un caso de uso.

Un actor representa un papel interpretado por una persona u otro sistema que interacciona con el sistema que se describe. La misma persona física puede representar diferentes papeles y así mismo, muchas personas pueden ser representadas por un mismo actor. Los actores deben ser descritos de una forma clara mediante un texto de sólo algunas líneas.

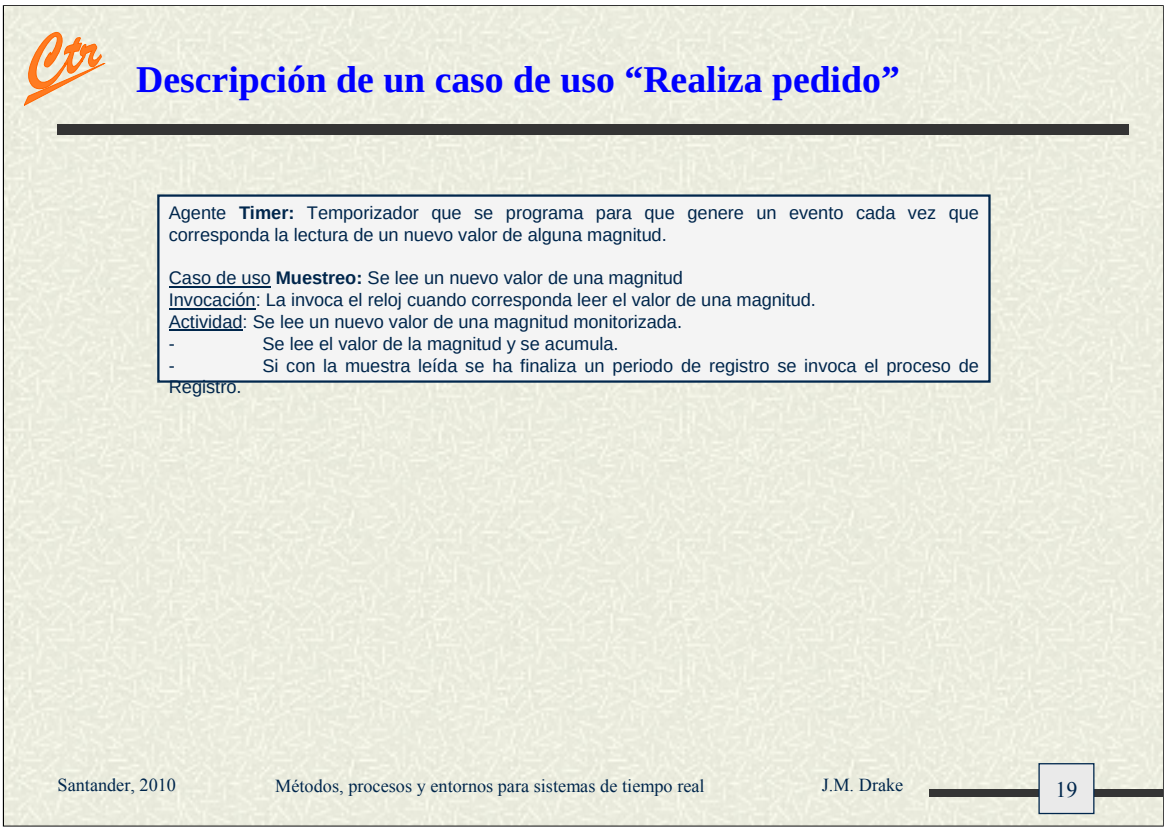
Los casos de uso se determinan observando y precisando, actor por actor, las secuencias de interacción (que llamaremos escenarios) que llevan a cabo con el sistema. Los casos de uso agrupan a familias de escenarios que desempeñan un mismo papel funcional visto desde el usuario. Los casos de uso son abstracciones del diálogo entre los actores y el sistema: describen interacciones potenciales, sin entrar en los detalles de cada usuario.

Los escenarios son instancias de los casos de uso. Los escenarios representan secuencias de mensajes entre objetos que colaboran para producir algún tipo de comportamiento del sistema. Los escenarios constituyen un método muy intuitivo de representar los detalles de los requerimiento, en particular los que implican restricciones temporales.

Los enlaces entre actores y casos de uso representan los protocolos, que constituyen el hilo de conductor de la secuencia de eventos que intercambian ambos.



Durante la fase de inicio del proyecto hay que decidir lo que hace el sistema que se desarrolla, y debe identificarse la funcionalidad desde un punto de vista conceptual y abstracto. Durante la posterior fase de elaboración hay que detallar los requerimientos del sistema hasta que este quede no ambiguo y pueda ser desarrollado por los diseñadores y programadores.
Cada caso de uso incluye todos los detalles sobre lo que tiene que hacerse para conseguir la funcionalidad. Se necesita considerar la funcionalidad básica, las alternativas a ella, el funcionalidad que se produce cuando se presentan errores o estados de excepción, las precondiciones o estado de partida previsto para el caso de uso y las post-condiciones o situación que tiene que producirse cuando el caso de error concluya. Los casos de uso puede incluir pueden incluir condiciones, bifurcaciones, alternativas y bucles de flujo de control.
Cuando el caso de uso es sencillo, es fácil describir linealmente la información asociada, sin embargo cuando el caso de uso se hace complejo y presenta muchas ramas o bucles, existen diferentes estilos de describir los casos de uso.
La descripción de un caso de uso comprende los siguientes elementos:
•Inicio del caso de uso:Identifica el evento o interacción que desencadena el caso de uso.
•Final del caso de uso: El evento o situación con la que el caso de uso concluye.
•La interacción entre sistema y los actores: describe la secuencia abierta de interacciones que constituyen el caso de uso.
•El intercambio de información: Describe las informaciones o datos que se intercambian en las interacciones.
•La cronología y el origen de las informaciones: Informaciones que se requieren (internas o externas)
•Las repeticiones de las iteraciones: Bucles de iteraciones que se realizan dentro del caso de uso.
•Las opciones de las iteraciones:Bifurcaciones o ramificaciones de interacciones.


Descripción de un caso de uso “Realiza pedido”
Agente **Timer:** Temporizador que se programa para que genere un evento cada vez que corresponda la lectura de un nuevo valor de alguna magnitud.
Caso de uso **Muestreo:** Se lee un nuevo valor de una magnitud
Invocación: La invoca el reloj cuando corresponda leer el valor de una magnitud.
Actividad: Se lee un nuevo valor de una magnitud monitorizada.
- Se lee el valor de la magnitud y se acumula.
- Si con la muestra leída se ha finaliza un periodo de registro se invoca el proceso de Registro.
Santander, 2010
Métodos, procesos y entornos para sistemas de tiempo real
J.M. Drake
19

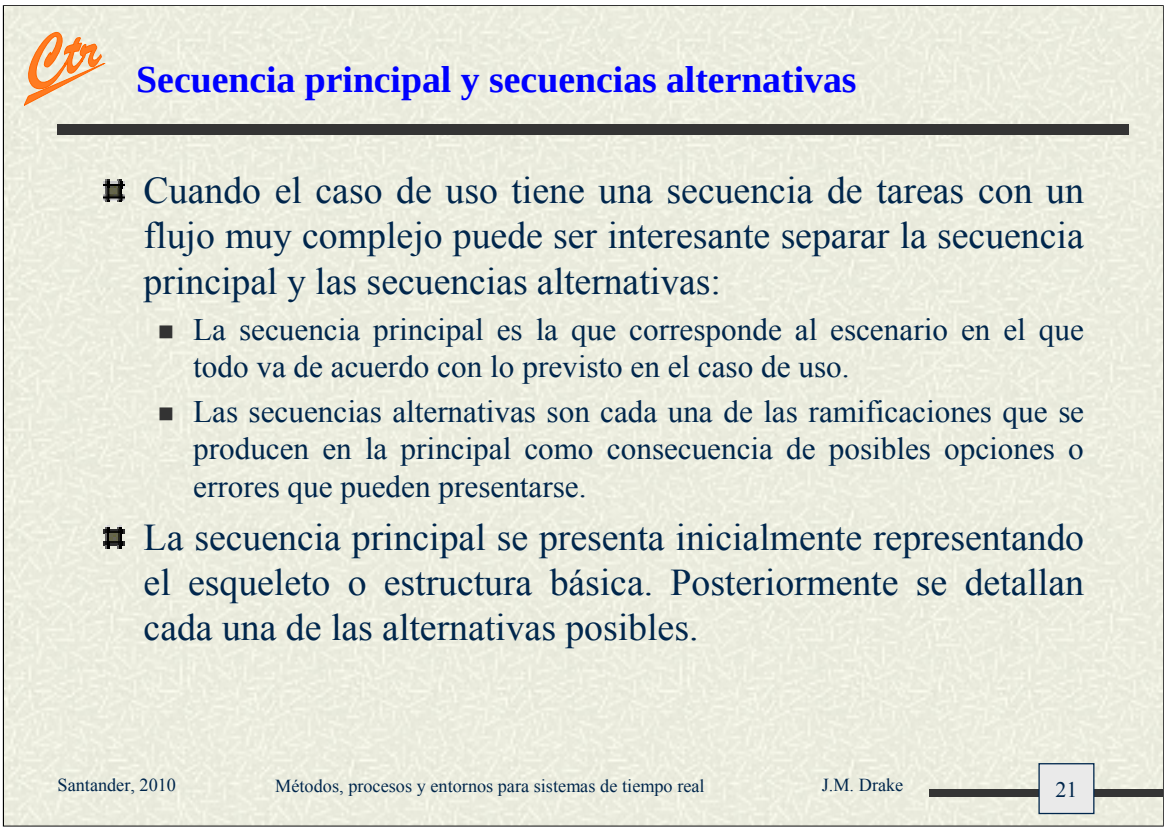
Pre- y post- condiciones

- Una pre-condición es la situación en que debe estar el sistema para que el caso de uso se ejecute en la forma prevista. Si la precondición no se satisface, el caso de uso puede seguir secuencias no previstas.
- Una post-condición es una característica que se tiene que satisfacer en el sistema si se ha partido de unas precondiciones ciertas y se ha ejecutado el caso de uso.

  La post-condición tiene que ser cierta con independencia de que rama o alternativa del caso de uso se ha seguido.

Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 20

Las precondiciones y post-condiciones que ocurría o ocurre antes y después de que se ejecute el caso de uso. Ellos establecen el estado en que debe estar el sistema al iniciarse el caso de uso (precondición) o en que estado va a quedar el sistema al terminar el caso de uso (post-condición). La post-condición debe cumplirse con independencia de que rama o alternativa de flujo del caso de uso se ha seguido.



Un caso de uso detallado puede ser bastante complicado. Esto no suele ser lo que ocurre en la fase de inicio. Sino cuando evoluciona y va acumulando detalles. A fin de hacer que los casos sigan siendo básicamente legibles,es conveniente describirlo en dos bloques: Primero se describe la funcionalidad básica utilizando la secuencia principal, y luego se describen las variaciones que pueden realizarse sobre él a través de secuencias alternativas que hacen referencia al principal.

La secuencia principal corresponde a la ejecución del caso de uso mas habitual, aquel que no tiene errores ni se opta por atajos. A menudo esta secuencia se denomina secuencia “feliz”, la que se produce cuando todo va bien. Al no tener secuencias alternativas, la secuencia principal es lineal y en consecuencia muy legible.

Una secuencia alternativa es una que se produce cuando se elige una situación o evento posible diferente del considerado en la principal.

Las secuencias alternativas son muy útiles para declarar eventos de error, de excepción o de aborto en el que el instante de ocurrencia no está bien definido (puede ocurrir en cualquier instante).

Secuencias alternativas al caso de uso Realiza_perdido

Identificador: Realiza_pedido **Actores que lo inician:** Cliente y Agente.

Precondiciones: Un cliente registrado en el sistema ha accedido correctamente al sistema.

Secuencia de eventos de flujo:

1. El cliente introduce su nombre y dirección.

2. Si el cliente introduce el ZIP, el sistema introduce la ciudad y región.

3. El cliente introduce los códigos de los productos que desea incluir en el pedido.

... (ver trasparencia 28)

Secuencias alternativas:

•En cualquier instante, antes de pulsar "Ejecutar" el cliente puede pulsar "Cancelar". El pedido no es registrado y el caso de uso finaliza.

•En el paso 8, si cualquier información es incorrecto, el sistema avisa al cliente para que corrija la información.

•En el paso 9, si el pago no es confirmado, el sistema avisa al cliente para que corrija las información de la tarjeta o cancele. Si el cliente opta por corregir, se pasa al paso 6. En caso contrario, termina.

Herramientas gráficas UML para requisitos detallados

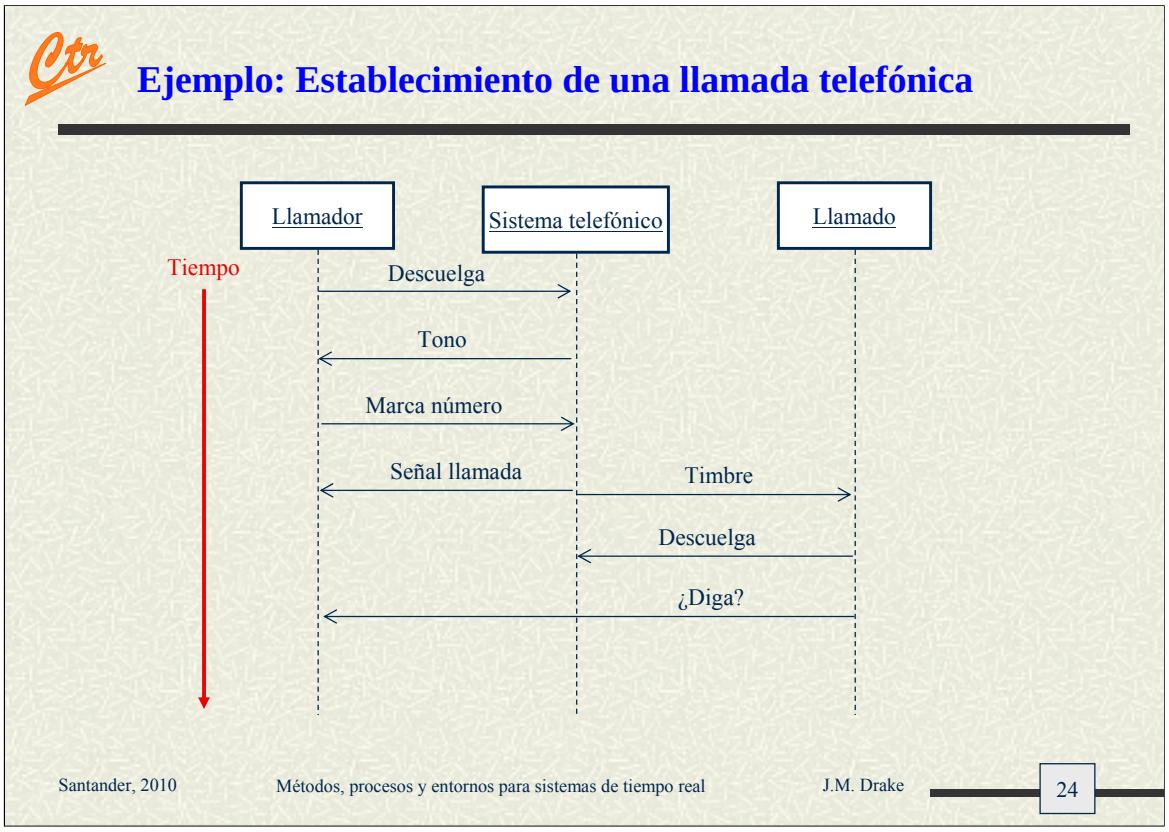
- Estáticas: Recurso de organización
  - **Casos de usos**
- Dinámicas: Recurso para mostrar el comportamiento dinámico:

  Describen escenarios (casos particulares)
  - **Diagramas de secuencias**
  - **Diagramas de colaboración.**

  Describen la dinámica completa:
  - **Diagramas de actividad (modelo operativo)**

Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 23

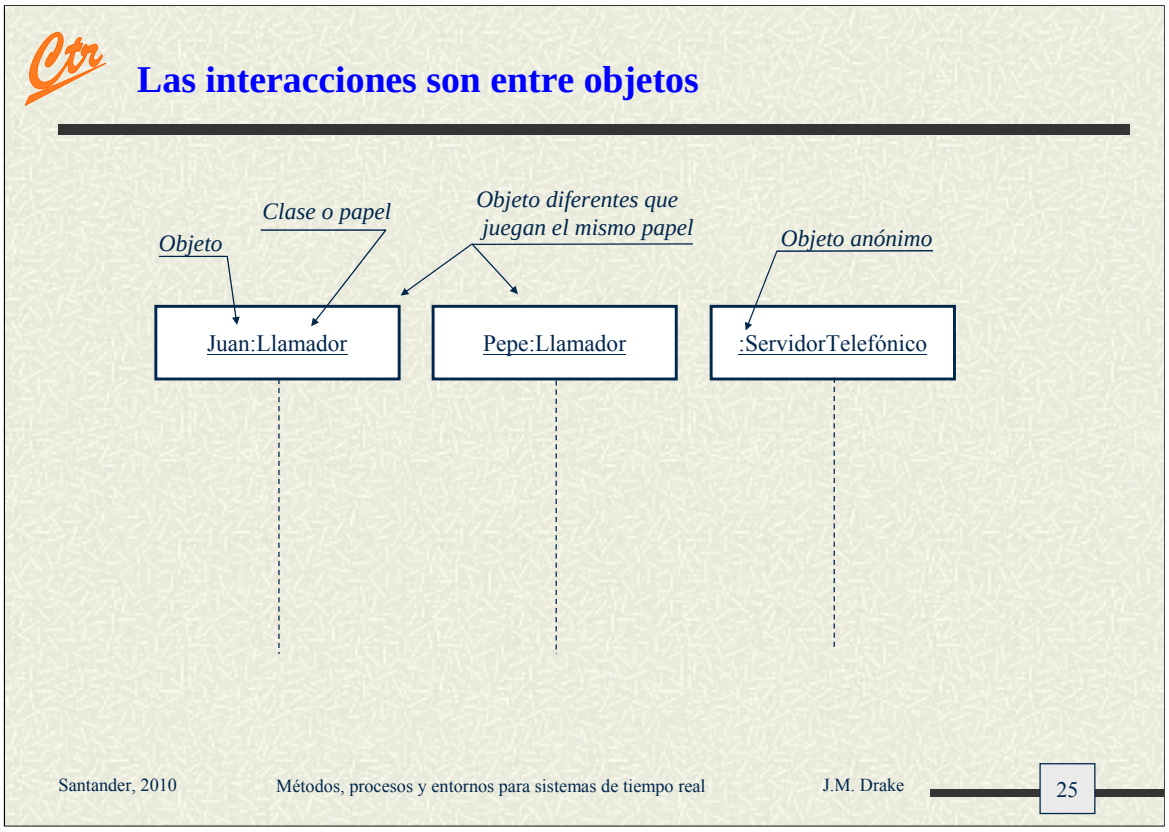
UML ofrece muchos recursos para expresar la funcionalidad de los elementos estructurales. Estos recursos pueden utilizarse bien para especificar la funcionalidad, o bien para describir detalladamente esa funcionalidad. En el primer caso se puede considerar como una herramienta de ayuda al análisis o diseño, en el segundo caso, puede llegar incluso a constituir un lenguaje de programación, del que se puede generar automáticamente el código.
Elementos para la descripción de elementos estructurales y su funcionalidad:
•**Diagrama de estado**: sirve para describir un elemento estructural a través de una máquina de estados finitos.
•**Diagrama de actividad**: es un diagrama de estado con una semántica específica formulado como forma de expresar algoritmos complejos a través de diagramas de flujo.
Elementos para la descripción de mecanismos de operación de interacción entre grupos de objetos del sistema:
•**Diagrama de secuencias**: Permite representar a través de diagramas de transferencia de mensajes entre objetos organizados en el tiempo cualquier escenario de intercolaboración entre ellos.
•**Diagrama de colaboración**: Permite representar a través de diagramas gráficos los escenarios de colaboración que se establecen entre grupos de objetos.



En la figura se muestra un ejemplo de diagrama de secuencias que describe el proceso de comunicación de dos personas (Llamador y llamado) a través de una línea telefónica.

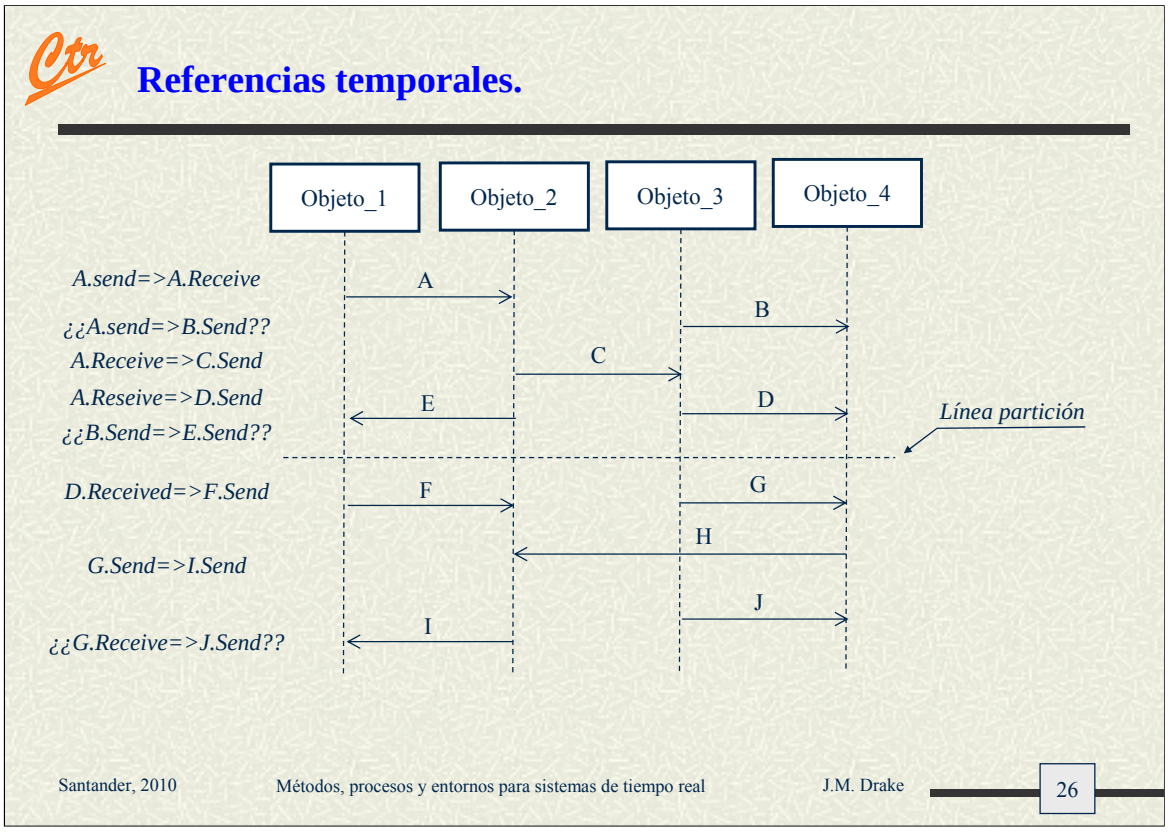
La interacción entre objetos se conciben como mensajes que se intercambian entre objetos.

El diagrama de secuencias no describe la totalidad de las posibilidades que se pueden plantear, sino que únicamente describe una secuencia posible. Por ejemplo, en este caso no se contempla la posibilidad de que el objeto Llamado esté comunicando, y en consecuencia falle la conexión. Esta situación diferente debería formularse con otro diagrama de secuencia.



Los participantes en un diagrama de secuencias son objetos concretos. Su denominación consta del identificador del objeto y de la clase o papel al que pertenecen.

Puede designarse un objeto como representativo de una clase haciendo únicamente referencia a la clases.



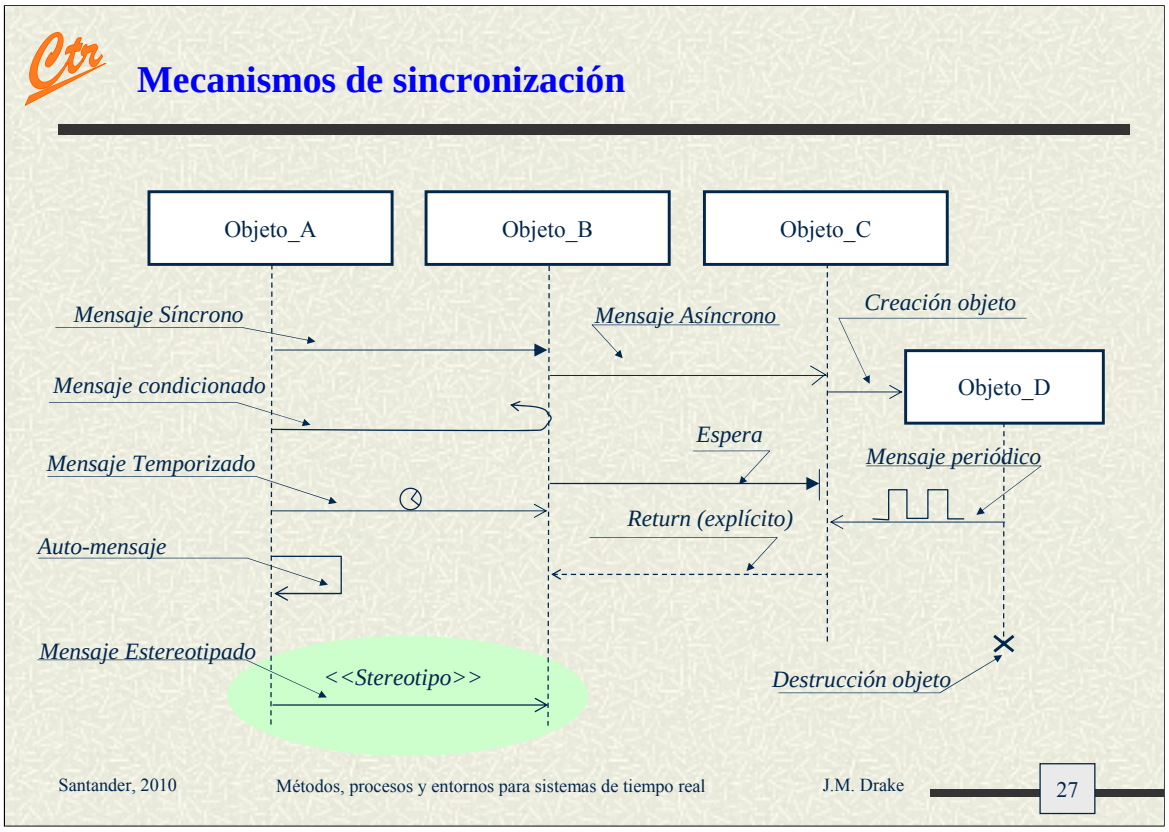
Aunque la dirección vertical representa la evolución del tiempo, solo son comparables los instantes de las acciones si están referenciadas a la actividad de un mismo objeto.

• La acción de enviar un evento es siempre anterior a su recepción. Así, A.Send =>A.Receive (la acción A.Send precede a la acción A.Receive)

• Para dos acciones que no tengan referencia con un objeto común no son ordenables:
A.Send puede preceder o seguir a B.Send.

• Cuando existe una secuencia de referencias comunes, puede establecerse la precedencia.
G.Send=>G.Receive=>H.Send=>H.Receive=>I.Send=>I.Receive

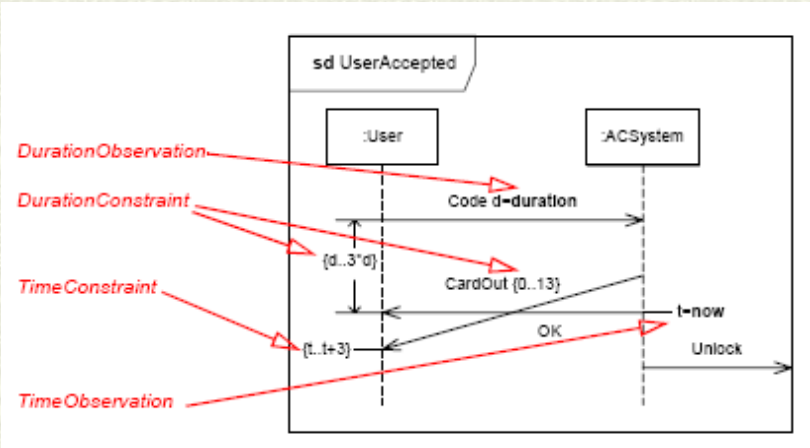
• A través de una línea de partición horizontal, se pueden establecer precedencias explícitas:
E.Send=>E.Receive=>G.Send=>G.Receive

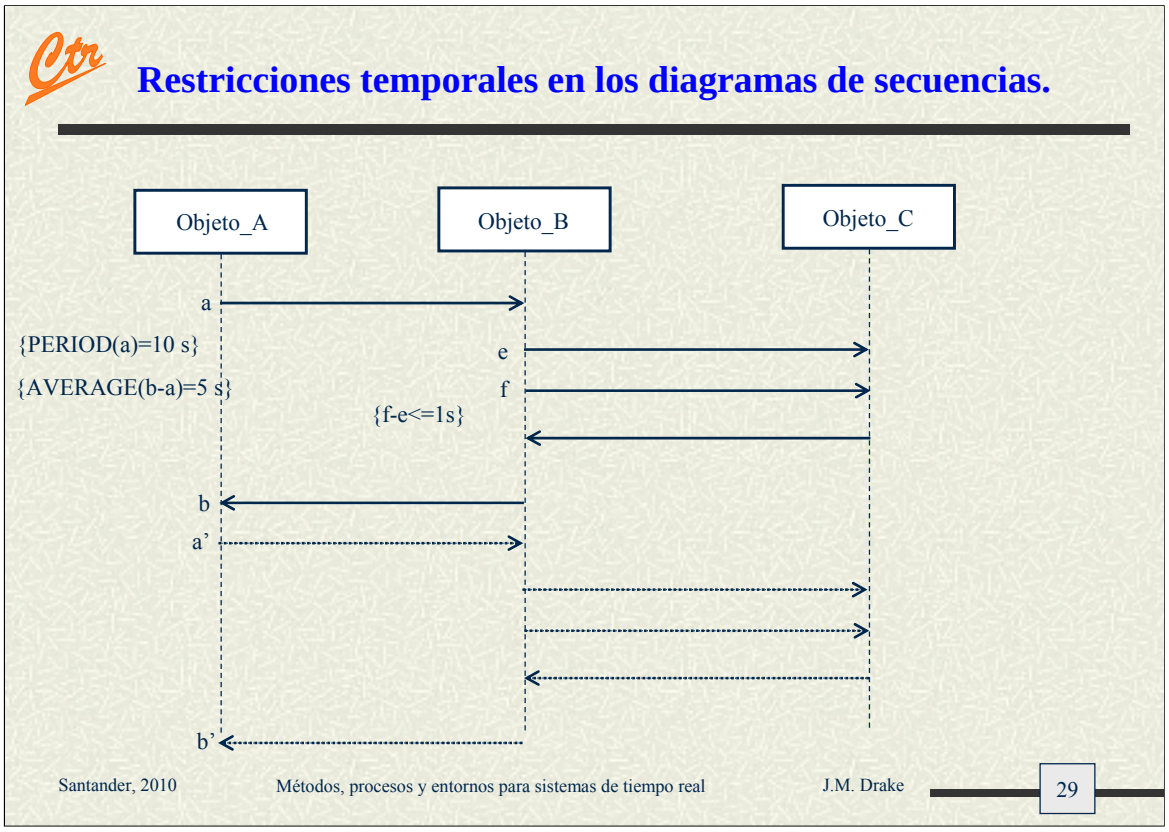


Mensaje Síncrono: El flujo en el objeto que envía el mensaje se suspende hasta que es recibido por el objeto receptor.
Mensaje Asíncrono: El emisor transfiere el mensaje y continua sin esperar a que el mensaje sea recibido.
Mensaje Condicionado(Balking): El mensaje es transferido si el receptor está disponible para aceptarlo de forma inmediata, en caso contrario, no se transfiere.
Mensaje Temporizado: El mensaje es transferido con un tiempo de guarda. Si transcurre el tiempo de timeout y el receptor no lo acepta, el mensaje no se emite.
Espera: Un objeto se suspende hasta que otro objeto alcanza un cierto estado.
Return: Habitualmente el retorno de una invocación es implícita. Algunas veces interesa manifestar su ocurrencia.
Los patrones de generación de los mensajes pueden explicitarse mediante iconos.

Todos estos iconos están definidos en el estándar UML, pero no suelen estar soportados por las herramientas CASE. Por ello, se suele utilizar el método equivalente de la inclusión del estereotipo explícito.

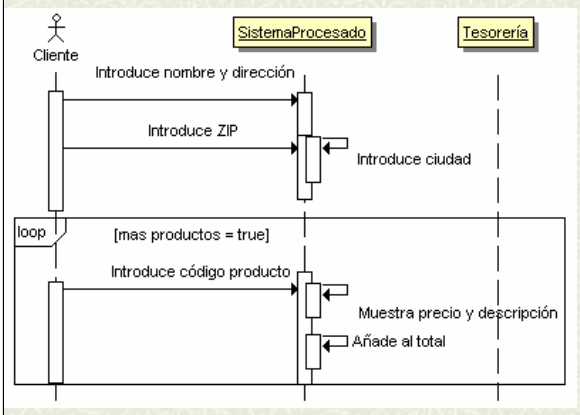
Especificación de tiempos en UML2

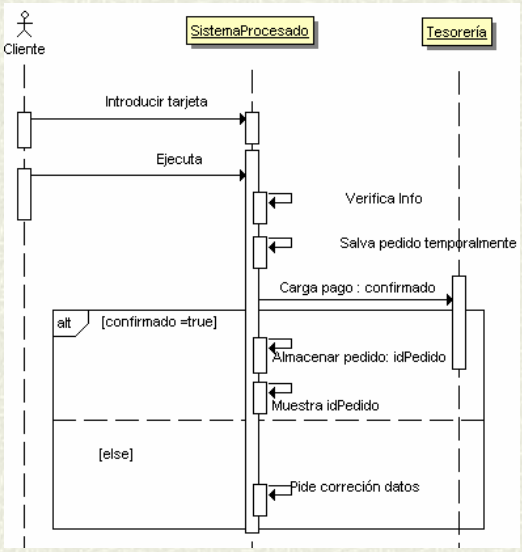




Los instantes en que se producen las acciones se pueden designar mediante etiquetas. Y haciendo uso de ellas se pueden formular múltiples restricciones sobre las respuestas temporales de los sistemas.

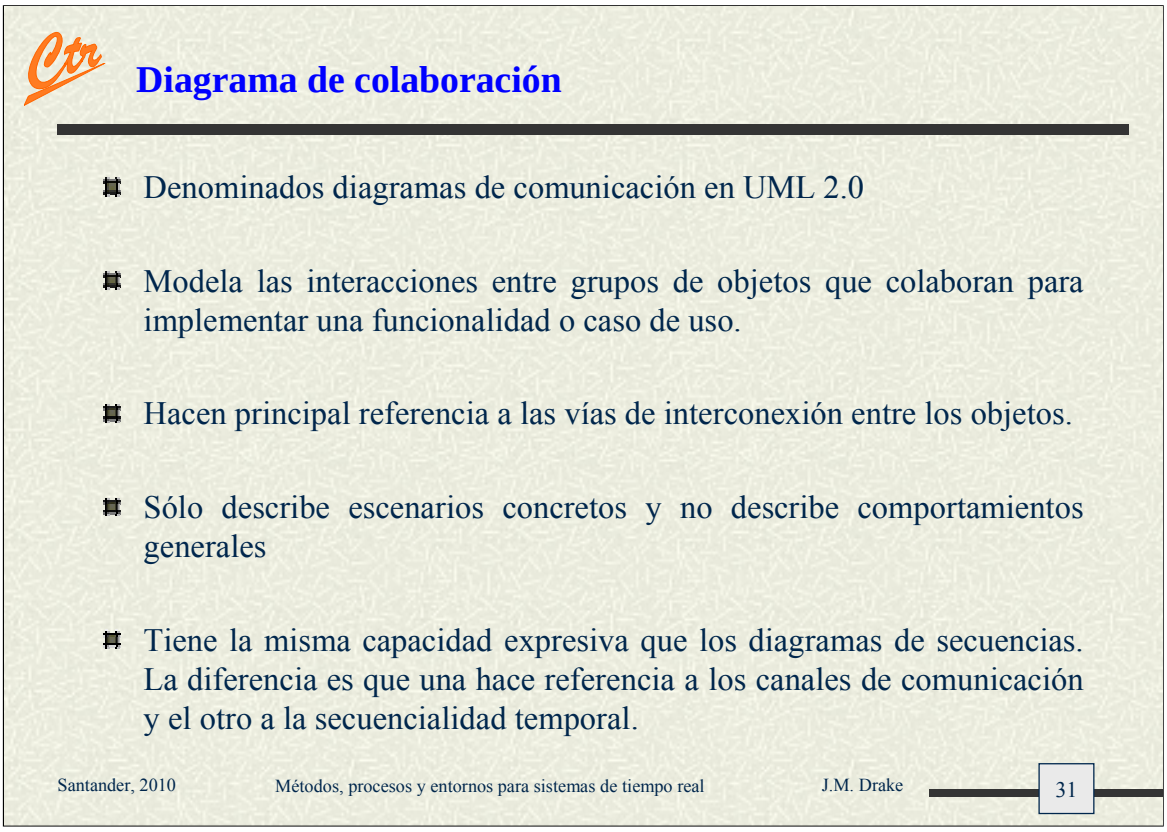
Sintaxis en diagramas de secuencias: fragmentos





Continuación

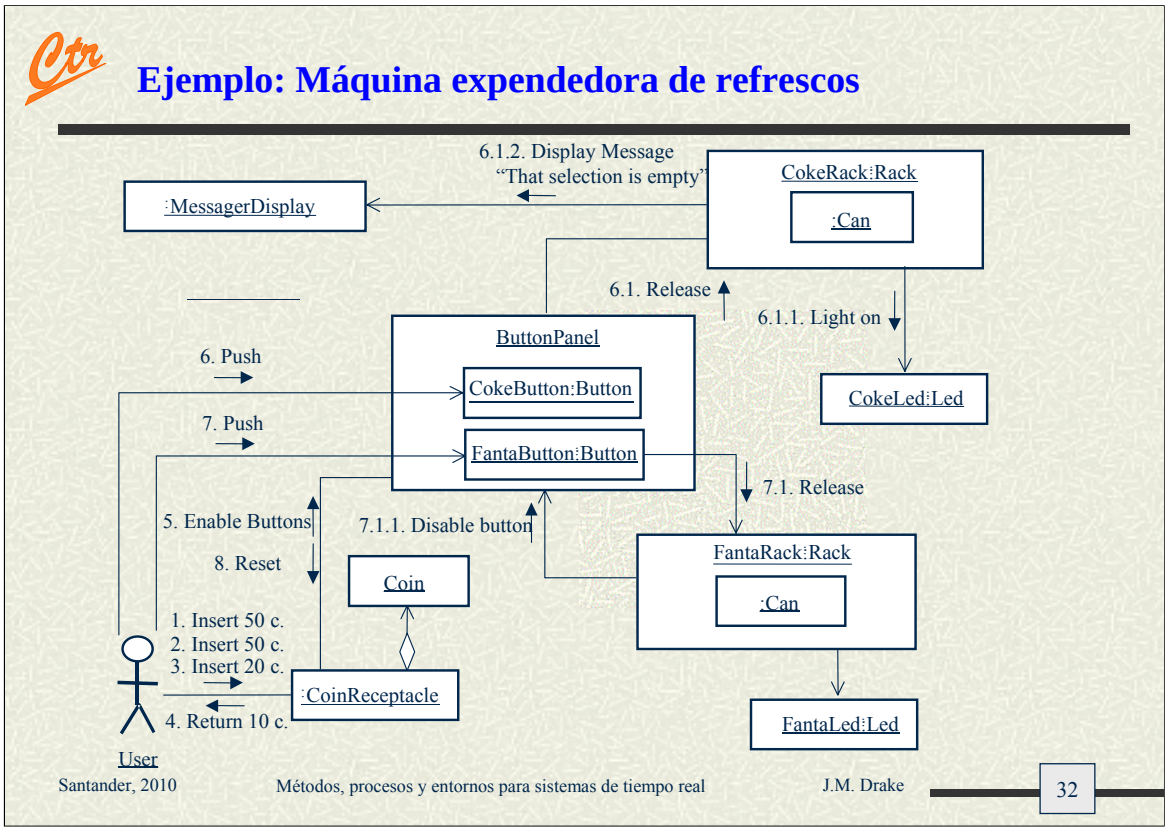
Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 30



Los diagramas de colaboración muestran interacciones entre objetos, resaltando la estructura espacial estática que permite la colaboración del grupo de objetos. Los diagramas de colaboración expresan a la vez el contexto de un grupo de objetos (a través de la representación de los objetos y de los enlaces que están establecidos entre ellos) y las interacciones entre los objetos (por la representación de envío de mensajes).

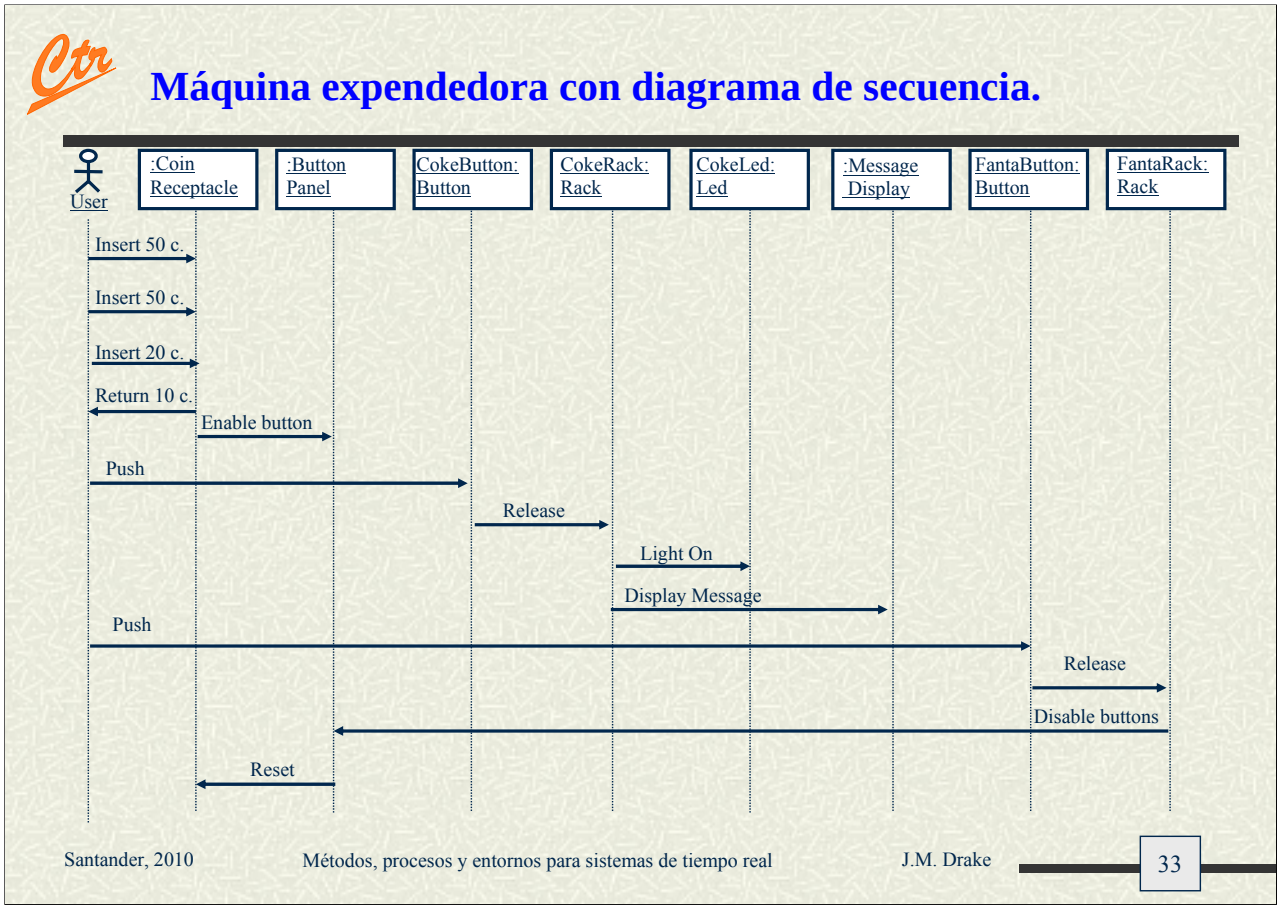
El contexto de una interacción puede comprender los argumentos, las variables locales creadas durante la ejecución, así como los enlaces entre los objetos que participan en la interacción.

Una interacción se realiza mediante un grupo de objetos que colaboran intercambiando mensajes. Los mensajes se representan a los largo de los enlaces que enlazan los objetos por medio de flechas orientadas hacia el destinatario del mensaje.

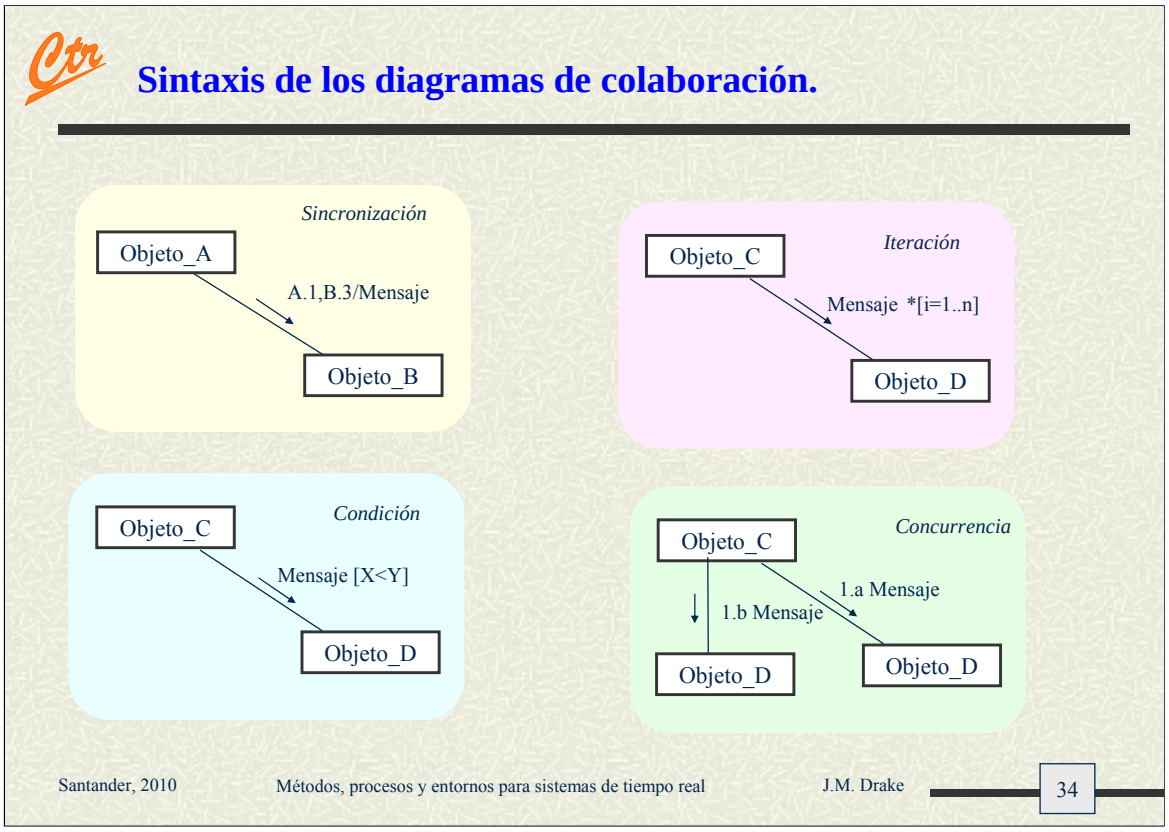


Escenario ejemplo: Drink Machine: El usuario introduce 60 céntimos como dos monedas de 50 céntimos y una moneda de 20 céntimos, y la máquina retorna una moneda de 10 céntimos. Selecciona Coke, pero la máquina no tiene este tipo de bebida y lo manifiesta a través de un LED y un mensaje. Luego elige Fanta que si está disponible, por lo que la máquina la suministra.

En los diagramas de colaboración el tiempo (orden en el tiempo) se puede expresar introduciendo un índice en los mensajes.
En sistemas concurrentes en los que existen diferentes sesiones de interacción se utiliza un segundo índice de sesión.



Escenario ejemplo: Drink Machine: El usuario introduce 60 céntimos como dos monedas de 50 céntimos y una moneda de 20 céntimos, y la máquina retorna una moneda de 10 céntimos. Selecciona Coke, pero la máquina no tiene este tipo de bebida y lo manifiesta a través de un LED y un mensaje. Luego elige Fanta que si está disponible, por lo que la máquina la suministra.



En los diagramas de colaboración se pueden formular diferentes técnicas expresivas:

Sincronización: Un mensaje se envía si previamente se han producido otras interacciones.

Condición: Un mensaje se envía si se verifica una cierta condición de entorno

Iteración: Un mensaje se envía por un número determinado de veces (FOR) o mientras se verifique una determinada condición (WHILE).

Diagramas de actividad

- Los diagramas de actividad son un tipo especializado de diagrama de estados que modelan el comportamiento dinámico de un procedimiento, o caso de uso haciendo énfasis en el proceso que se lleva a cabo.
- La transición básica entre estados es producida por conclusión de la actividad asociada al estado.
- Los diagramas de actividad es uno de los elementos de modelado que son mejor comprendidos por todos, ya que son herederos directos de los diagramas de flujo.
- Modela flujos de control secuenciales y concurrentes.

Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 35

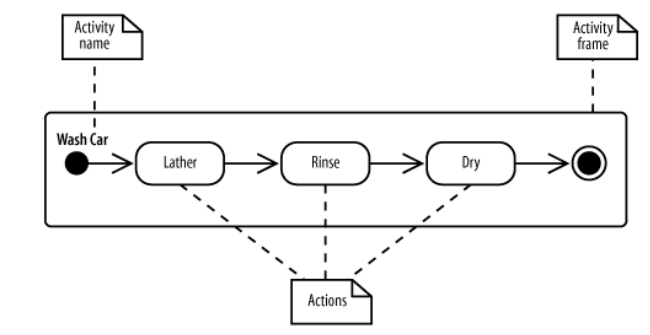
Un diagrama de actividad es una variante de un diagrama de estados, organizado principalmente respecto de actividades que fluye como consecuencia de la finalización de las mismas. Es una herramienta muy útil para la descripción de un método o función y de un caso de uso.

Un diagrama de actividad representa el estado de ejecución de un programa bajo la forma de un desarrollo de etapas agrupadas secuencialmente en ramas paralelas de flujo de control.

Cada actividad representa una etapa particular en la ejecución del método que describe. Las actividades se enlazan por transiciones que se ejecutan automáticamente como consecuencia de la finalización de la actividad, y que se representan por flechas. Cuando una actividad termina se desencadena la transición y empieza la actividad siguiente. Las actividades no tienen transiciones internas, ni transiciones desencadenadas por eventos.

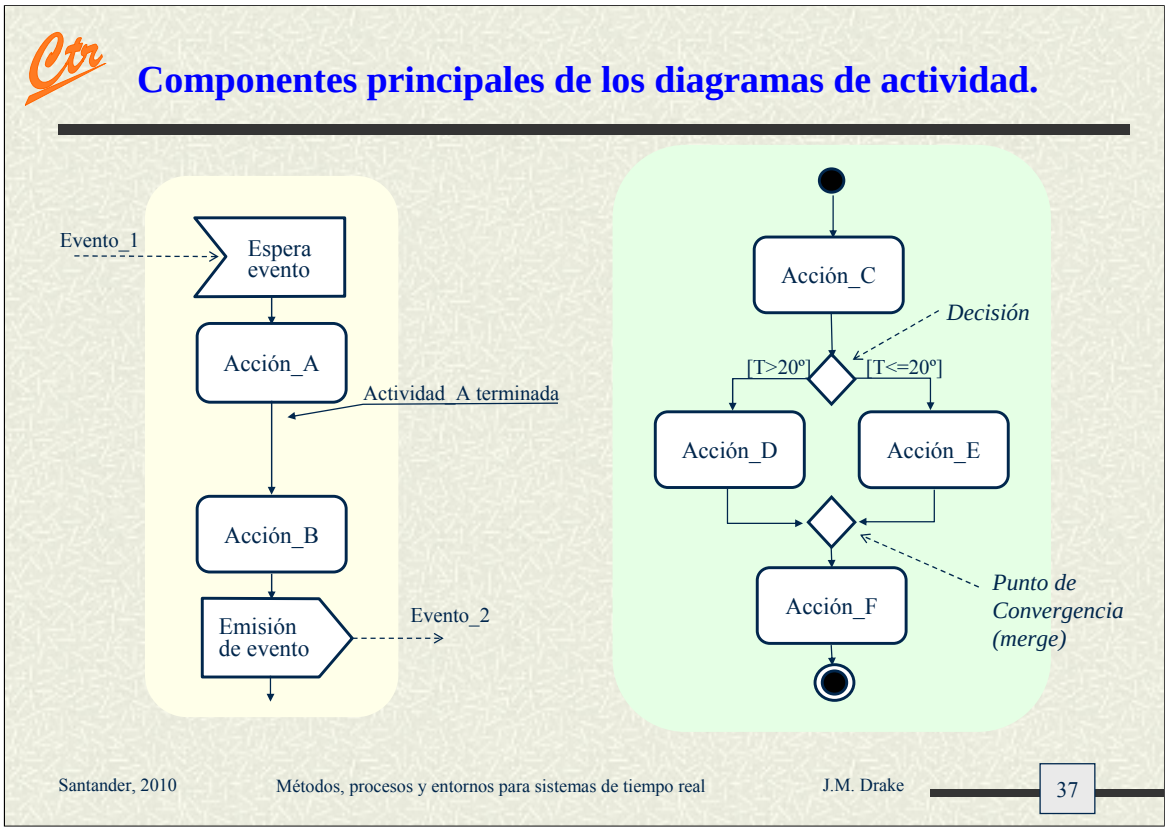
Actividades y acciones

- Una acción es un paso de un proceso que tiene la semática "run to completion" (Se inicia para ser terminado)
  - Ejemplo de acciones: Crear o eliminar un objeto, invocar un procedimiento, realizar un cálculo simple, etc
- Una actividad es un conjunto de acciones que modelan un proceso. No tiene la semántica "run to completion". Una actividad se modela mediante un diagrama de actividad.
- Enjabonar, enjuagar o secar un coche son acciones de la actividad "Lavar un coche"



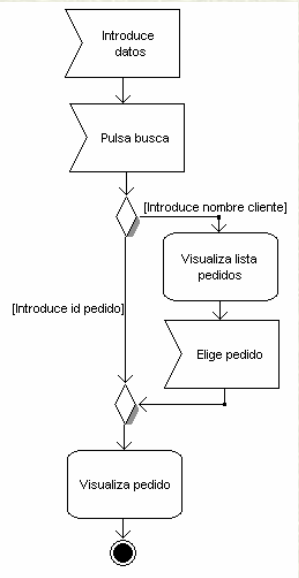
Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 36

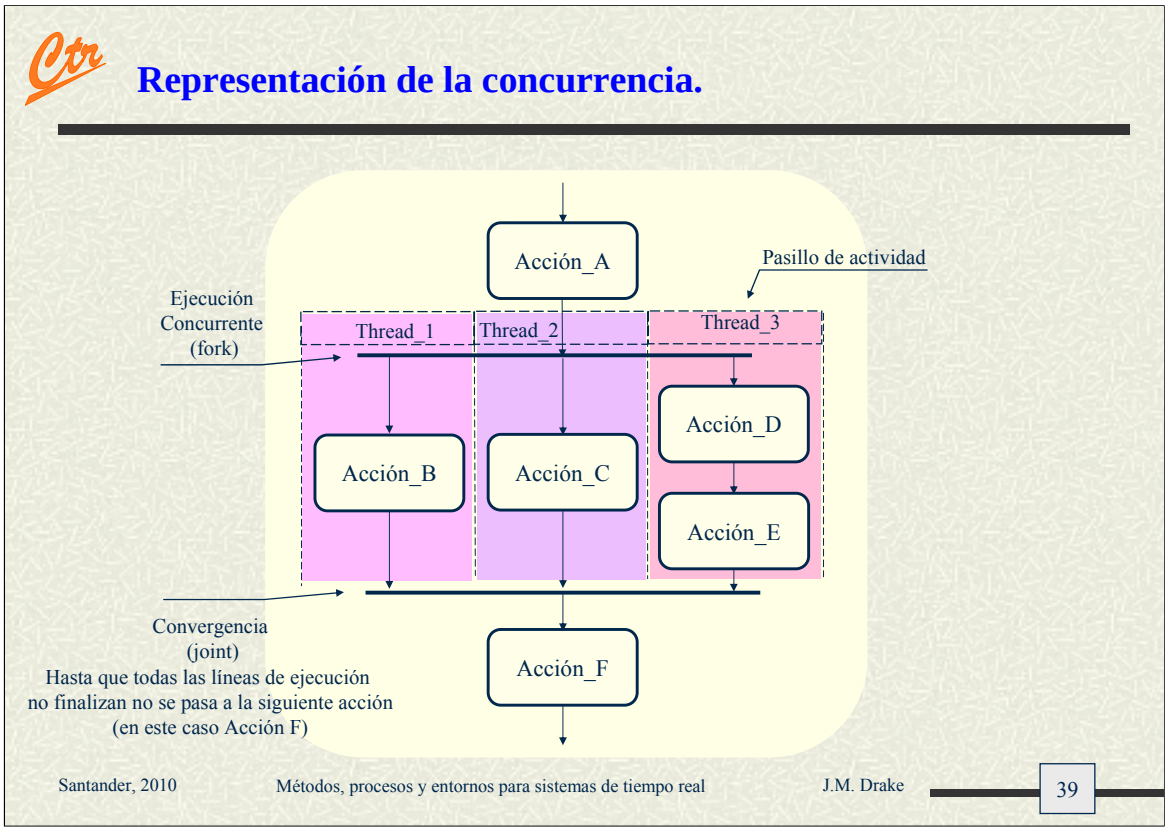
Una acción es una especificación de una primitiva ejecutable que tiene como consecuencia un cambio de estado en el modelo, y puede consistir en la transferencia de un mensaje, modificar un enlace o cambiar de valor un atributo.
Las acciones son primitivas computacionales simples que tienen la semántica de iniciarse para ser terminada ("run-to-completion"). Esto no significa que no pueda ser expulsada de su ejecución por otra de mayor prioridad, sino que aun en este caso posterior se retorna a ella para completarla. Un objeto que este ejecutando una acción, no acepta nuevos mensajes hasta que la concluya.



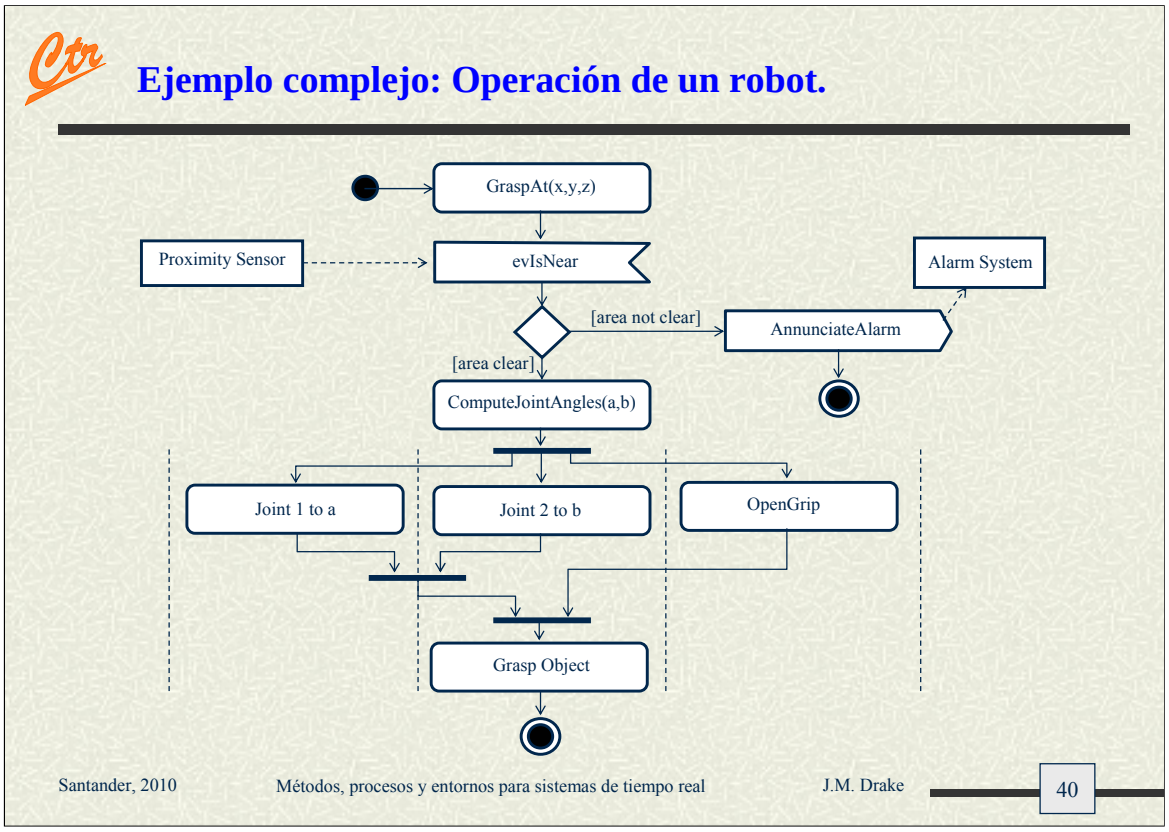
Acción: Tarea que es realizada dentro de procedimiento o caso de uso que se describe.
Espera de evento: El flujo de control se suspende hasta que se genera un evento o mensaje.
Emisión de un evento: Se genera un evento o mensaje que puede ser un resultado o una forma de interacción con otras secciones de la tarea.
Bifurcación condicionada (Branching): El flujo de control pasa a una u otra rama del diagrama, en función de que satisfaga o no una cierta condición. Las condiciones deben ser no ambiguas y completas (no repetir ningún caso ni dejar ningún caso sin alternativa)
Convergencia (Merge): Varia líneas alternativas se encauzan hacia una secuencia de actividades.

Ejemplo: Caso de uso “Busca Pedido” detallado





Los diagramas de actividad permiten formular ejecuciones concurrentes de actividades.
Barras de generación de concurrencia (Fork): El flujo de control se replica en varias líneas que se ejecutan concurrentemente.
Barras de sincronización (Joint): Las diferentes líneas de control de entrada se suspenden en ella hasta que todas la ha alcanzado. Luego se unifican en una única linea de control.
Pasillos de actividad (Swimlane): Representan a los objetos o procesos que soportan las diferentes líneas de flujo de control concurrentes.

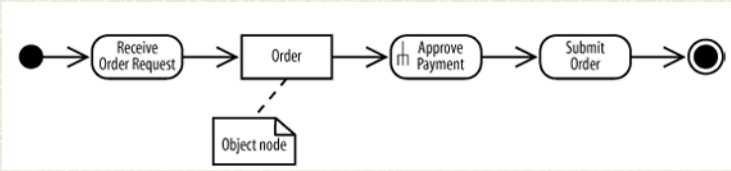


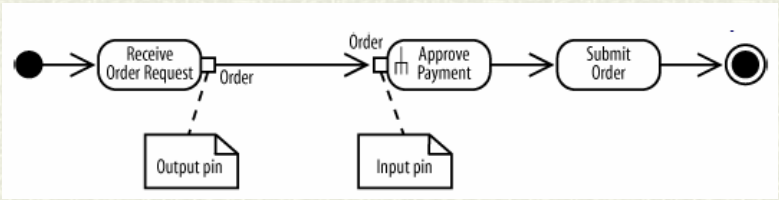
Ejemplo Operación de un robot: Modela el proceso de cogida de un objeto mediante un brazo robotizado con una pinza.

•Recibe las coordenadas del objeto que debe ser agarrado.
•Verifica que el área de desplazamiento está libre. En caso contrario genera una alarma y no la ejecuta.
•Computa la modificación de las coordenadas de las articulaciones internas en función del destino.
•Concurrentemente mueve las articulaciones internas, y abre la pinza.
•Cuando todas las anteriores han finalizado, cierra la pinza y coge el objeto.

Otros aspectos de los diagramas de actividad: objetos

Representación de objetos





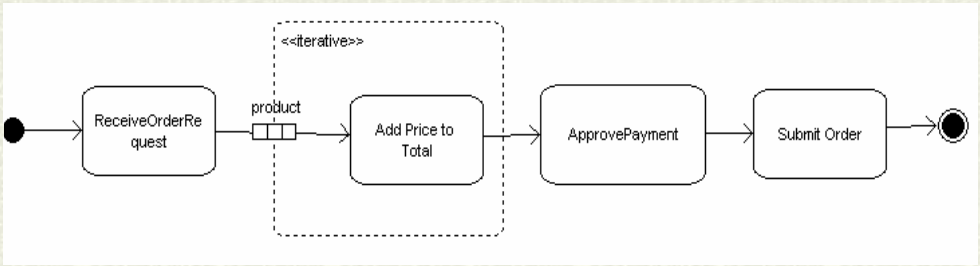
Representación de objetos como entrada o salida de acciones

Representación de objetos cambiando de estado

Santander, 2010 Métodos, procesos y entornos para sistemas de tiempo real J.M. Drake 41

En un diagrama de actividad se pueden representar los objetos de datos que se generan, se consumen o se intercambian en un proceso y que son relevantes para su descripción. Para representarlos se utilizan los denominados “Object Node”
Cuando un objeto de datos se representan como una caja, significa que esos datos existen en el punto de flujo de control en que se insertan, se intercambian entre las acciones entre las que es insertado.
Los objetos pueden mostrar los estados en que se encuentran y pueden representar los puntos de inicio y finalización de la actividad que representan.

Diagramas de actividad complejos



Regiones de expansión:
Realiza las acciones interiores para todos los elementos del pin de entrada