# Modelado de Aplicaciones con Procesos Concurrentes y Distribuidos

Universidad Nacional de La Matanza - Departamento de Ingeniería e Investigaciones Tecnológicas
Florencio Varela 1903 - (1754) San Justo - Bs As – Argentina - Tel: 011 4480-8900 - Interno 8752

**Mg. Daniel A. Giulianelli** dgiulian@unlam.edu.ar
**Ing. Rocío A. Rodríguez** rrodri@unlam.edu.ar
**Ing. Pablo M. Vera** pablovera@unlam.edu.ar

**RESUMEN**

En este paper se presenta un nuevo diagrama al cual hemos denominado D.E.A. "Diagrama de Estados Activos". El mismo permite modelar todos los aspectos de las aplicaciones con procesos concurrentes y distribuidos. Como punto de partida se evaluaron los distintos modelos de UML 2.0 y viendo que ninguno de ellos se adaptaba por completo a las necesidades presentadas, se elabora éste modelo que toma las características de algunos diagramas de UML y agrega elementos necesarios para este tipo de aplicaciones. El DEA permite visualizar en un único diagrama aspectos que de modelarse con UML implicarían la construcción de varios diagramas y el uso de estereotipos.

## 1. ANTECEDENTES

Con la intención de modelar aplicaciones con procesos concurrentes y distribuidos usando UML (ver introducción a UML [3], [6] y [9]), nace en el año 2005 este trabajo de investigación.

A fin de difundir los resultados obtenidos en las distintas versiones preliminares del presente trabajo, realizamos ponencias en los siguientes eventos [8]:

1) Congreso Argentino de Ciencias de la Computación. Universidad Nacional de Entre Ríos (CACIC 2005).
2) Jornada de Jóvenes Investigadores de Universidades Nacionales organizada por la Universidad Nacional de San Luís (JI2005).
3) Congreso Argentino de Ciencias de la Computación organizado por la Universidad Nacional de San Luís (CACIC 2006).
4) Jornada Iberoamericana de Ingeniería de Software e Ingeniería del Conocimiento, organizado por la Pontificia Universidad Católica del Perú (JIISIC07).

Uno de los objetivos del presente trabajo es presentar las mejoras realizadas sobre el DEA y realizar una comparativa de las ventajas del DEA con respecto a UML 2.0.

## 2. DIAGRAMA DE ESTADOS ACTIVOS

Tomando como referencia a UML nos propusimos construir un Diagrama que reuniera las características necesarias para poder con él modelar procesos concurrentes y distribuidos. Para ello tomamos las particularidades del Diagrama de Transición de Estados (DTE), las correspondientes al Diagrama de Actividades, algunas características del Diagrama de Despliegue e incorporando características propias de los sistemas con procesos concurrentes y distribuidos, construimos un modelo al que hemos denominado "Diagrama de Estados Activos (DEA)".

Este modelo que proponemos lo consideramos adecuado para:

1. Representar un proceso distribuido: Se utilizan las calles que caracterizan al clásico Diagrama de Actividades [7], para determinar en que nodo se realizará la ejecución de ciertos procesos. Para aquellos casos donde el medio de comunicación entre los nodos sea de relevancia, se indicará con una línea entre ambos el tipo de vínculo utilizado, tal como se haría en UML en un Diagrama de Despliegue.
2. Representar la concurrencia de procesos: Se utilizan las líneas de sincronismo del Diagrama de Actividades.
3. Detallar la información de los estados: Además de indicarse el nombre del estado se puede detallar como en el clásico DTE acciones de entrada y salida, transiciones internas y eventos diferidos.
4. Mostrar el cambio de estados: Se utilizan las transiciones del DTE indicando la o las

condiciones, así como el evento y en el caso que existan, las acciones que permiten el paso de estado.

### 2.1. Particularidades del DEA

Del análisis del modelo presentado surgen las siguientes conclusiones:

Al agregar al clásico DTE las calles, queda indicado si el proceso comenzado en cierto estado, al pasar a otro, involucra o no un cambio de nodo. Es decir que al cumplirse cierta condición podrá continuarse el proceso en otro nodo (calle).

Las líneas de sincronismo pueden compartirse entre varios nodos. Por lo tanto el flujo del procesamiento se continúa en un nodo específico cuando este obtenga una respuesta de los otros nodos que están procesando en forma paralela, ya sea por necesitar un resultado o por la finalización de los mismos.

*Particularidades del modelado de procesos distribuidos:* Requiere que los procesos alojados en distintos host se comuniquen para poder intercambiar información. Existen dos formas distintas para realizar dicha comunicación: mediante el envío de mensajes y mediante la utilización de RPC (Remote Procedure Call – Llamadas a Procedimientos Remotos). A su vez los mensajes pueden ser asincrónicos o sincrónicos. Un mensaje sincrónico obliga al emisor a esperar una respuesta antes de continuar con sus tareas mientras que con el asincrónico se envía el mensaje y se continúa con el resto de las tareas. En cambio los RPC son en su mayoría, llamadas sincrónicas ya que actúan como simples llamadas a procedimientos como si estuvieran en una misma computadora.

Para modelar la comunicación entre los procesos se mantiene la nomenclatura habitual de UML donde un mensaje sincrónico se representa con una punta de flecha rellena y un mensaje asincrónico con una punta flecha abierta (es recomendable consultar el manual de especificación de UML 2.0 [5] y [10]). Respetando esta nomenclatura proponemos para mayor claridad diferenciar los RPC de los mensajes usando la nomenclatura mostrada en la Figura 1:

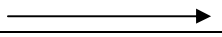
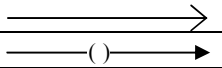
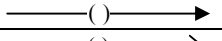
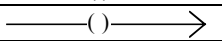
| | |
|---|---|
| Mensaje Sincrónico |  |
| Mensaje Asincrónico |  |
| RPC Sincrónico |  |
| RPC Asincrónico |  |

Figura 1: Nomenclatura mensajes y RPC

Los nodos que interactúan pueden tener diferentes tipos de conexiones físicas entre si, lo que en algunas ocasiones es importante destacar, ya que según sea el tipo de enlace habrá ciertas velocidades de transferencia que variarán. Lo que puede permitir decidir derivar ciertos procesos a un determinado nodo o a otro, o manejar distintos tiempos de time out para las respuestas. Al igual que en el Diagrama de Despliegue de UML se indicará en el DEA a través de una línea entre cada par de nodos la característica del enlace.

*Particularidades del modelado de procesos concurrentes:* Puede requerir la diferenciación de los distintos hilos (threads) de ejecución del sistema. Si bien la propuesta al incorporar las líneas de sincronismo del Diagrama de Actividades ya muestra los procesos o hilos que se ejecutan en paralelo, se propone también para una notación más clara en casos en los que se detallen varios estados dispares dentro de cada hilo, utilizar una notación de calles con líneas punteadas dentro de la calle principal del nodo para indicar el procesamiento independiente de cada hilo (ver Figura 2).

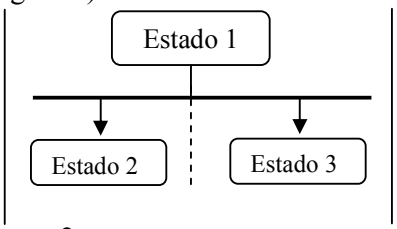



Figura 2: Elementos que conforman al DEA

Cuando se utilizan hilos es muy posible que existan recursos que se deben compartir y por lo tanto es necesario administrar su acceso, ya que solamente un thread puede utilizarlo en un momento dado. Dos métodos comunes para la administración de recursos en un ambiente concurrente son los semáforos y los monitores [4].

Proponemos dos construcciones gráficas para indicar recursos compartidos y el método de acceso a los mismos. Dentro de estas construcciones es posible especificar el tipo de recurso al que nos estamos refiriendo (ver Figura 3).

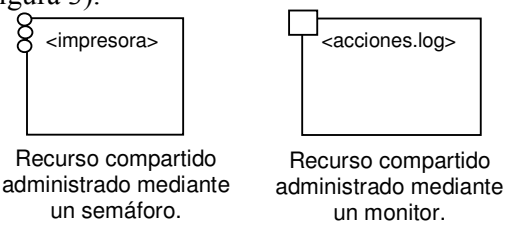



Figura 3: Nomenclatura designada para Semáforos y Monitores

*Cardinalidad:* Cuando se cuenta con múltiples conexiones provenientes de distintos nodos, generalmente existe un proceso principal que monitorea las conexiones y al llegar nuevas conexiones crea distintos hilos para cada una de ellas. A veces, también las acciones que realiza cada uno de esos hilos son idénticas, por lo tanto proponemos la utilización de una nomenclatura similar a un objeto con varias ocurrencias en UML pero aplicado a las calles que está representando el hilo en ejecución (ver Figura 4).

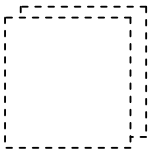


Figura 4: Las acciones que se realizan "n" veces se indican dentro de esta representación gráfica.

## 3. METODOLOGÍA DE MODELADO

Se propone a continuación un listado de pasos que permitirán realizar un Diagrama de Estados Activos, de forma sistemática.

1. Identificar los distintos nodos o locaciones donde se va a ejecutar la aplicación para definir con cada uno de ellos las calles.
2. Identificar el nodo desde el cual se va a iniciar el proceso o la funcionalidad que estamos modelando y ubicar la calle que lo representa en el centro del diagrama para minimizar el cruce de líneas entre las demás calles, ya que es usual que éste sea el nodo principal que comanda al resto.
3. Determinar cuales conexiones entre nodos son relevantes de destacar e indicar entre dichos nodos el tipo de conexión física. Solo se deberán marcar las conexiones entre los nodos que tengan cierta relevancia.
4. Comenzar el modelado desde el nodo ubicado en la calle central, usando la nomenclatura habitual del DTE con el círculo lleno que indica el comienzo del proceso.
5. Continuar el modelado utilizando de forma habitual los estados, cambios de estado y transiciones del DTE. Teniendo en cuenta que es posible utilizar la nomenclatura del diagrama de actividades cuando tenemos actividades que se realizan en paralelo.
6. Cuando el flujo de control principal o uno de los flujos secundarios salen del nodo actual utilizar la nomenclatura de mensajes o RPC propuesta para clarificar de que forma se realiza dicha comunicación.
7. Si uno de los estados requiere la utilización de un recurso compartido indicar si el control de acceso al mismo se realiza mediante un semáforo o un monitor.
8. Para rutinas repetidas dentro de un mismo nodo, es decir que disponen de un control de conexión y para cada una de ellas se realiza un proceso igual, independientemente de que nodo proviene la conexión, englobar dicha funcionalidad dentro de un recuadro de cardinalidad.

## 4. MODELADO POR MEDIO DEL DEA

Se desea modelar el control de acceso a una bóveda de alta seguridad. Esta cuenta con diferentes mecanismos para controlar el acceso a la misma.

Como primera medida de seguridad se requiere poseer la llave que habilita un teclado mediante el cual se ingresa una clave de acceso. Existe un número de veces que se puede ingresar en forma errónea la clave antes de ser disparada la alarma. Una vez dado por válido dicho código se realiza un escaneo de retina para autentificar a la persona. Luego se habilita un teclado en cada una de las dos sucursales, en donde se ingresa una clave de seguridad. Una vez chequeada la validez del código se procede a realizar un escaneo de retina a quienes ingresan dichas claves. Estos

escaneos son procesados mediante los patrones almacenados en el servidor de la casa central. Si son válidos se habilita el acceso a la bóveda.

En la Figura 5 se muestra como queda modelada a través del DEA está aplicación.

## 5. COMPARATIVA: DEA - UML

UML brinda herramientas para el modelado de procesos concurrentes mediante el uso de clases activas, dichas clases indican que sus objetos contienen diferentes hilos de ejecución.

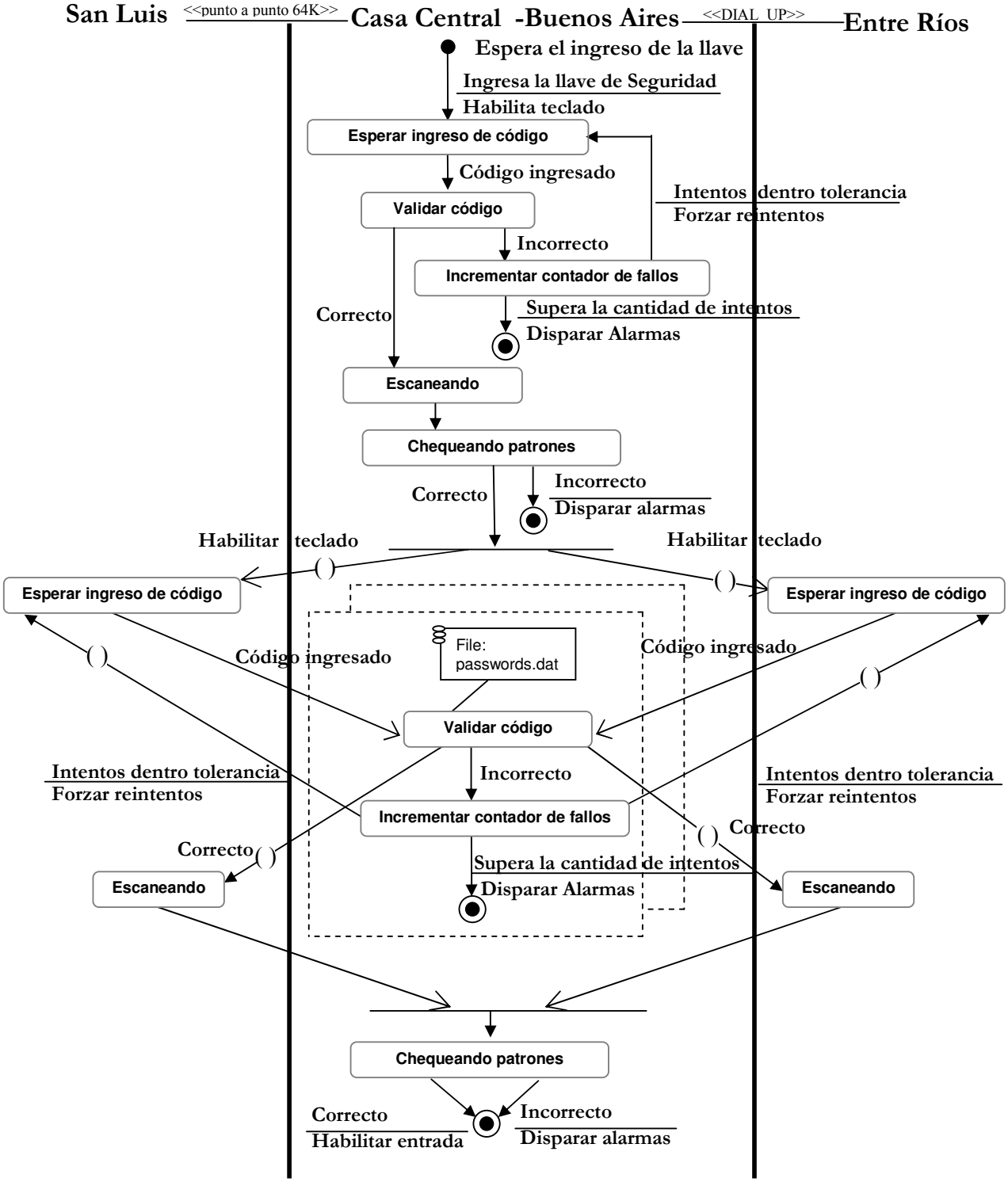



Figura 5: Este DEA integra los elementos típicos de este tipo de aplicaciones anteriormente planteados.

Para poder modelar la comunicación entre diferentes procesos y threads UML propone la utilización de diagramas de objetos adornados con varios estereotipos como ser *location* que nos indica la ubicación física de ese hilo o proceso. También para establecer la ubicación física es posible realizar un Diagrama de Despliegue que indique que procesos ejecuta cada nodo y cuales son las conexiones físicas entre ellos. Si se quiere modelar como se comunican procesos ubicados en distintos nodos es posible que un Diagrama de Colaboración donde cada objeto tenga que estar estereotipado con la propiedad *location* se vuelva muy difícil de seguir. Por ello la propuesta del DEA es utilizar las calles para ver que es lo ocurre dentro de cada nodo y como se comunican unos con otros.

Para la comunicación entre procesos UML propone la diferenciación entre mensajes sincrónicos y asincrónicos. Pero no se especifica una semántica particular para RPC, los que invocan un procedimiento ubicado en otro nodo, sin que en dicho nodo necesariamente, se estuviera ejecutando un proceso que monitoree, a la espera de mensajes. Por ello se hace necesaria la diferenciación entre mensajes y RPC.

En cuanto a la utilización de Diagramas de Estados, UML propone su utilización para modelar el comportamiento de cada una de las clases activas por separado. Siendo necesario para ver su interacción construir un Diagrama de Colaboración.

Para los recursos cuyo acceso debe ser sincronizado, UML propone estereotipar los métodos que por ejemplo deban ser sincronizados dentro de una clase pero no hace referencia a elementos físicos compartidos como ser un archivo, una unidad de disco, etc. Elementos que muchas veces es necesario destacar para ver como varios hilos van a competir por dicho recurso.

## 6. CONCLUSIONES

De la comparativa presentada en el Item 5, puede desprenderse que el DEA es una herramienta que permite modelar aplicaciones con procesos concurrentes y distribuidos, plasmando las características de estas aplicaciones en un único diagrama. Si bien el diagrama parece a simple vista ser más complejo para leer (Figura 5), evita el tener que realizar una serie de modelos, los cuales traen aparejados los siguientes inconvenientes:

1. Tener que relacionar los distintos modelos para lograr llegar a la visión que se observa en el DEA
2. Tener que usar estereotipos para poder modelar características no nomencladas en UML.
3. El conjunto de modelos no logra mostrar la diferencia entre mensajes y RPC. Así como tampoco se puede visualizar la cardinalidad.

Por los motivos expuestos sostenemos que el DEA ayuda al análisis, incorporando las características de los modelos de UML y agregando aquellas características específicas de este tipo de aplicaciones, no soportadas por dichos modelos.

## 7. REFERENCIAS


**LIBROS:**

[1] Booch G, Rumbaugh J y Jacobson I. *El lenguaje unificado de modelado*. Addison Wesley, pp: 392-394, 2003.

[2] Fowler M. *UML Distilled*. Addison Wesley, Third Edition, Pearson Education, pp: 1-16 2004.

[3] Kendall S. *Fast Track UML 2.0,* Apress, California 2004.

[4] Stalling W. *Operating Systems Internals and Design* Principles. Third Edition, Prentice Hall., pp: 208-230, 1998.

**E BOOK :**

[5] *OMG Unified Modeling Language Specification*. Versión 1.5 Marzo 2003

**SITIOS WEB:**

[6] http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/multiple-html/

[7] http://www.creangel.com/uml/actividad.php

[8]http://www.investigamos.com.ar/CONGRESOS.htm

[9]http://www.microsoft.com/spanish/msdn/articulos/archivo/230801/voices/modelsoftware.asp

[10] http://www.uml.org/#UML2.0