OBJETOS Y TIEMPO REAL EN UN PROBLEMA DE REINGENIERÍA

Roberto Uzal (*), Adriana Echeverría (**) Alberto Sánchez (***), Juan José Aversa (***) y Alberto Dams (**)

Resumen

Se informa sobre una aplicación en ambiente de base de datos relacionales, inicialmente implementada utilizando un 4GL. La complejidad de las consultas, la optimización del uso de la red teleinformática y los criterios de facturación de los servicios suministrados, obligaron a utilizar, en las especificaciones, criterios y herramientas de los métodos estructurados destinados a sistemas en tiempo real.

Surgió la necesidad de replantear parte de la implementación con herramientas que hacen un uso intenso de conceptos de la orientación a objetos. En el proceso de reingeniería se aplicaron ideas expuestas por I. Jacobson y F. Lindström, S. Mullender y los aspectos instrumentales recomendados por J. Ellis.

Los fundamentos de este trabajo fueron discutidos por equipos de proyectos de investigación de la FIUBA y de la UNSL. Los aspectos instrumentales se corresponden con desarrollos en una empresa comercial.

Domicilios:

Dr Roberto Uzal
Virrey Olaguer y Feliú 3052 4/D
(1426) Capital federal

Ing Alberto Dams
Virrey Olaguer y Feliú 2476 4/A
(1426) Capital Federal

Lic Adriana Echevería
Cabildo 1183 9/D
(1426) Capital Federal

Prof Alberto Sánchez
Ejército de los Andes 950 - (1er piso Depto Informática)
e-mail : alfanego@unsl.edu.ar
(5700) San Luis

Sr Juan José Aversa
Ejército de los Andes 950 - (1er piso Depto Informática)
e-mail : jjaversa@unsl.edu.ar
(5700) San Luis

OBJETOS Y TIEMPO REAL EN UN PROBLEMA DE REINGENIERÍA

Roberto Uzal (*), Adriana Echeverría (**) Alberto Sánchez (***), Juan José Aversa (***) y Alberto Dams (**)

1. Introducción

1.1. Aspectos conceptuales acerca de la reingeniería de sistema en producción con el propósito de reformular parte de las aplicaciones, utilizando conceptos de la orientación a objetos.

Jacobson y Lindström plantean tres casos posibles de reingeniería de viejos sistemas para su replanteo, con herramientas que utilizan conceptos de la orientación a objetos:

a) Un cambio integral de la técnica de implementación, sin cambios en la funcionalidad.

b) Un cambio parcial en la técnica de implementación, sin cambio en la funcionalidad. Sólo una parte de los viejos programas serán reimplementados utilizando conceptos de la orientación a objetos.

c) Cambios restringidos al ámbito de la funcionalidad.

I. El caso que se informa queda incluido en la variante b). Los pasos sugeridos por Jacobson y Lindström son los siguientes:

II. Identificar la parte del sistema que será reimplementada usando técnicas de la orientación a objetos.

III. Preparar un modelo de análisis de la parte a ser modificada. Se incluye el entorno en la mencionada modelización.

IV. Establecer la correspondencia entre dicho modelo de análisis y los objetos presentes en la vieja implementación.

V. Iterar los pasos anteriores hasta lograr un aceptable ajuste.

VI. Ejecutar en paralelo:
   VI a. Diseñar el nuevo subsistema y sus interfaces con el viejo sistema.
   VI b.Modificar las partes remanentes del viejo sistema generando las interfaces necesarias con el nuevo subsistema.

VII. Integrar y probar el funcionamiento del nuevo subsistema interactuando con el viejo sistema modificado.

1.2. Justificación de la utilización de criterios y herramientas de tiempo real en la especificación del sistema actualmente en producción.

Si se hubiesen dimensionado el hardware y los recursos teleinformáticos según la hipótesis de máxima carga (pico de carga), esto hubiese afectado el período de recupero de la inversión y comprometido la rentabilidad. Por lo tanto, se llegó a una solución de compromiso que implica que de producirse el evento correspondiente al pico de carga, algunas transacciones serán abortadas.

En otros términos, existen transacciones con una muy baja demanda de recursos y, por otra parte, otras que por la muy alta complejidad de la consulta (tablas intermedias de gran tamaño, funcionamiento de algoritmos de interpretación alfafonética, etc.), consumen un importante volumen de las prestaciones de los equipos y red teleinformática.

El sistema "toma conciencia" de la complejidad de la transacción cuando la misma ha avanzado un importante tramo en su ejecución: no es lo mismo consultar por el señor "Obdulio Cirilo Sturzenegger" que por "Juan Carlos Pérez". El esfuerzo computacional para recuperar por similitudes alfafonéticas al segundo de ellos, es muchísimo mayor.

El sistema está preparado para abortar transacciones muy complejas que se produzcan durante los picos de carga. Esta medida es el "mal menor" tanto para el funcionamiento global del sistema como para el usuario remoto al evitarle excesivos de utilización de los recursos teleinformáticos cuando el completamiento de la transacción sobrepasa determinado límite de tiempo.

Las consultas implican transacciones anidadas. La facturación se produce en una instancia intermedia de la ejecución de la transacción. Si dicha transacción es abortada es necesario efectuar el "rollback" de todos los aspectos relacionados con la facturación ya concretada.

Consideraciones de este tipo provocaron que, a pesar de no tratarse estrictamente de un sistema de tiempo real ("blando"), resultara conveniente el uso de técnicas de especificación típicas de dichos sistemas.

Al pensarse en replantear parte de la aplicación con una herramienta de implementación que utiliza abundantes conceptos de la orientación a objetos, aparece como atractiva la propuesta de J. Ellis que también utiliza, en un enfoque que podría clasificarse como "híbrido", conceptos de la orientación a objetos para especificaciones de sistemas en tiempo real.

1.3. Aspectos instrumentales para el replanteo de las especificaciones para la reelaboración de parte de la aplicación utilizando una herramienta de implementación que hace un intenso uso de conceptos de la orientación a objetos.

Los aspectos conceptuales expuestos por Jacobson y Lindström sustentaron, en el trabajo que se informa, la utilización de las técnicas sugeridas por Ellis con las siguientes ventajas:

a) El enfoque de J. Ellis resulta particularmente atractivo cuando se requiere utilizar conceptos de la orientación a objetos para especificaciones en tiempo real.

b) Las técnicas de J. Ellis permiten ser utilizadas con el soporte de herramientas CASE de bajo costo, basadas en el enfoque estructurado tradicional.

c) Se verificó una aceptable correspondencia entre la forma de las especificaciones generadas como consecuencia de la aplicación de la metodología de J. Ellis y la herramienta de implementación utilizada en el replanteo de una parte del viejo sistema

En los puntos subsiguientes del trabajo se describirán las prestaciones del viejo sistema, la forma en la que reformularon parte de las aplicaciones, las variantes verificadas luego de la reingeniería y las conclusiones a las que arribaron los autores de este trabajo.

2. El antiguo sistema

El sistema suministra elementos de juicio para el análisis de riesgo crediticio brindando información acerca de personas físicas y jurídicas en cuanto a su actitud frente a sus obligaciones financieras. Los datos almacenados requieren un espacio físico de alrededor de trece gigabytes.

Las actuales aplicaciones, que interactúan con una base de datos con ciento treinta y siete tablas, presentan un esquema de lógica distribuída pero restringida a la interacción entre el servidor de la base de datos y el servidor de las aplicaciones. Dicho de otra forma, no toda la lógica de la aplicación reside en el servidor de aplicaciones; parte de los algoritmos, los más íntimamente ligados al tratamiento de los datos, residen en el servidor de la base de datos. Por otro lado, existe "mirroring" de algunas tablas críticas en el servidor de aplicaciones.

Un pasaje de parámetros no trivial a través de la red teleinformática correspondiente a este ambiente de lógica distribuída, la optimización de consultas muy complejas que implican la generación de tablas temporales de considerable tamaño y el manejo de una aplicación crítica en un entorno distribuído implementado con una significativa optimización de las inversiones, constituyen aspectos novedosos, en nuestro medio. El sistema evolucionará para llevar la interfaz con el usuario final a un esquema de presentación remota (actualmente se emulan terminales no inteligentes). El proceso de reingeniería de esa parte de las aplicaciones constituye el núcleo del presente informe..

La arquitectura prevista implica, por lo tanto, lógica distribuída ente el/los servidor/es de aplicaciones y el/los servidor/es de la base de datos y presentación distribuída entre el servidor de aplicaciones y los clientes de los usuarios finales en un entorno LAN/WAN.

Prestaciones de la aplicación

- Consultas sobre personas físicas por número de documento
- Consultas sobre personas físicas por apellido y nombre
- Consultas sobre empresas por CUIT
- Consultas sobre empresas por razón social
- Servicio de consultas sobre cheques (existencia de la cuenta, su estado y antecedentes del librador del cheque)

Las consultas por nombre o razón social disponen de facilidades alfafonéticas, es decir, si un nombre o apellido no es escrito correctamente, dentro de determinados límites se lo recupera y se lo somete a consideración del usuario con otros criterios para asegurar su identificación, por ejemplo, el domicilio.

Los algoritmos para la recuperación de personas por nombre y apellido o razones sociales de empresas, según distintos grados de similitud alfafonética, constituyeron un desarrollo no trivial implementado como rutinas en lenguaje C que son invocadas desde el código de cuarta generación.

Aunque la consulta se haga por número de documento o CUIT, dado que en la mayoría de los antecedentes judiciales no consta dicha información numérica, se deben efectuar búsquedas alfafonéticas que complementen la primera recuperación.

Existen facilidades, de disitnto grado de complejidad, para la administración del sistema. Se menciona, por ejemplo, la actualización en línea de la base de datos en lo que hace a inhabilitaciones del Banco Central.

3. La reformulación de parte de las aplicaciones

Se verificó una importante demanda, por parte de los usuarios finales, de facilidades para interactuar con el sistema desde un entorno gráfico. Se ha requerido, por ejemplo, disponer de calificaciónes de personas físicas o jurídicas, en cuanto al riesgo crediticio, por intermedio de gráficos e ideogramas. Existen herramientas que permiten pasar del actual esquema de terminal no inteligente (en el usuario final) a un entorno gráfico con incremento del rendimiento.

Los fundamentos para encarar la evolución, siguiendo a Jacobson y Lindström, son los siguientes:

- Es posible y, en la mayoría de los casos también necesario, reemplazar solamente partes de un sistema. En general no es realista sustituir completamente al sistema antiguo. Este enfoque puede colocar al proyecto fuera de la viabilidad económica. Es necesario encontrar el modo de reemplazar al viejo sistema por partes, de manera de permitir la amortización de las inversiones oportunamente realizadas.

- Es posible integrar técnicas de programación modernas, que utilicen conceptos de la orientación a objetos, en un sistema pre existente no orientado a objetos.

- Un cambio en el dominio de la aplicación es frecuentemente local en el sentido que involucra un comportamiento o una ocurrencia claramente delimitada.

- Todo sistema de información tiene una limitada vida útil. En general los cambios que se realicen sobre él debilitarán su estructura y provocarán que los próximos cambios sean más costosos. Jacobson y Lindström sugieren analizar de qué naturaleza deberán ser los cambios en un sistema, considerando en el mismo el "valor para el negocio" y la "flexibilidad ante el cambio' del sistema estudiado. Se verifican cuatro posibles combinaciones: a) "valor para el negocio", bajo; "flexibilidad ante el cambio", baja , b) "valor para el negocio", bajo; "flexibilidad ante el cambio", alta c) "valor para el negocio", alto; "flexibilidad ante el cambio", baja y d) "valor para el negocio", alto; "flexibilidad ante el cambio", alta.

Se recomiendan los siguientes enfoques:

Para el caso a) descartar el viejo sistema
Para el caso b) recurrir al enfoque tradicional de mantenimiento del sistema
Para el caso c) efectuar la reingeniería del sistema
Para el caso d) agregar las nuevas prestaciones requeridas

El hecho de cambiar la naturaleza de la herramienta de implementación, por un lado, y la rentabilidad alcanzada por el negocio, por el otro, colocan al trabajo que se informa en el caso c).

En el proceso de reingeniería correspondiente al caso informado se siguió el esquema habitual:

a) Ingeniería reversa de las aplicaciones actuales.
b) Definición de los cambios necesarios
c) Ingeniería "hacia adelante"

a) Ingeniería reversa de las aplicaciones actuales, comprende

- Contar con un grafo de bajo nivel con una descripción concreta de los componentes del sistema actual y sus interrelaciones.

- Disponer de un grafo abstracto que sea representativo del comportamiento y estructura lógica del sistema.

- La determinación de la correspondencia entre los grafos abstracto y concreto que se mencionaron.

b) Definición de los cambios necesarios

Los cambios necesarios pueden ser de funcionalidad o de técnicas de implementación. Los cambios de funcionalidad, es decir aquellos que se originan en cambios de las "reglas de negocio" han sido, tradiconalmente, los que han debido ser encarados con mayor asiduidad. Sin embargo, la evolución tecnológica y requerimientos de los usuarios en lo que hace a amigabilidad de las aplicaciones han generado una creciente demanda de cambios de técnicas de implementación.

Es necesario aclarar, por un lado, que los cambios de técnica de implementación, tal como el que se informa, no constituyen procesos sencillos y, por el otro, que es difícil encontrar ejemplos "puros". En la mayoría de los casos reales se encuentran combinaciones de los dos tipos de cambios descriptos.

c) Ingeniería "hacia adelante"

La naturaleza de la herramienta seleccionada provocó que la Ingeniería "hacia adelante" y la implementación del nuevo sistema fueran dos aspectos casi indistinguibles.

4. Las herramientas metodológicas utilizadas

La propuesta de John Ellis comprende

- Un modelo esencial de requerimientos
- Un modelo esencial de objetos

El modelo esencial de requerimientos describe, en forma abstracta, los aspectos sustanciales del sistema global al más alto nivel.

El modelo esencial de requerimientos está conformado por:

- Un diagrama de interfaces con entidades externas
- Un listado de eventos externos / respuestas
- Un diagrama de objetos y relaciones entre objetos

El modelo esencial de objetos es una representación abstracta del sistema global en la que se identifican los objetos esenciales del sistema, incluyendo el comportamiento y las interacciones necesarios para satisfacer las especificaciones del modelo esencial de requerimientos.

El modelo esencial de objetos se estructura mediante:

- Diagramas de flujos entre objetos
- "Template" o patrón de objetos
- Diccionario de datos

Síntesis de la naturaleza de las técnicas o herramientas con las que se implementan los modelos:

- Diagrama de interfaces con entidades externas: Define los límites del sistema y determina qué es lo que atraviesa dichos límites.

- Listado de eventos externos / respuestas: Define los eventos externos al sistema que requieren respuesta del mismo.

- Diagrama de objetos y relaciones entre objetos: Representa los objetos significativos del entorno externo y los objetos más importantes dentro del sistema y la relaciones entre ellos.

- Diagramas de flujos entre objetos: Es un conjunto jerárquico de diagramas que identifican los objetos más importantes del dominio del problema y explicitan los comportamientos e interacciones necesarios.

- "Template" o patrón de objetos: Describe la estructura de los objetos esenciales en lo que hace a sus atributos y métodos. Los objetos identificados en los diagramas de flujos entre objetos requieren una definición en el correspondiente patrón.

- Diccionario de datos: suministra definiciones, en lenguaje estructurado, correspondientes a "flujos", objetos y atributos.

La utilización del enfoque de John Ellis presentó las siguentes ventajas:

- Fue posible, para la formulación de los modelos, utilizar herramientas CASE orientadas hacia los métodos estructurados (Ward-Mellor).
- Resultó especialmente apta para el planteo de los modelos sugeridos por I. Jacobson y F. Lindstöm (1), es decir:

    - Una representación del sistema en su estado actual en un determinado nivel de abstracción (esta primera representación conviene que sea a un nivel de abstracción bajo).
    - Una representación lógica del sistema a un nivel tal que permita discutir eventuales cambios en su funcionalidad.
    - Una forma de plasmar decisiones de diseño o estudiar variantes de implementación
    - Una técnica que permita relacionar aspectos de las dos versiones de la implementación
    - Una técnica que delimite la parte del sistema que será objeto de la reingeniería

La orientación hacia especificaciones de tiempo real de la propuesta de Ellis permitió abordar particularidades de ese tipo en el antiguo sistema.

5. Las aplicaciones luego de la reingeniería

Inicialmente el sistema implicaba un desarrollo según el enfoque de lógica distribuida implementado sobre un servidor de la base de datos, en el que también residiría la lógica de la aplicación más íntimamente relacionada con el modelo de datos, y un servidor de aplicaciones en el que se ejecutaba la mayor parte de la aplicación. También se aprovechaba el segundo de los computadores mencionados para efectuar "mirroring" de los archivos más críticos de la base de datos.
Existían también servidores de aplicaciones con menores incumbencias, en distintos puntos del interior del pais.

Los usuarios finales accedían remotamente a los servidores de aplicaciones emulando terminales no inteligentes con computadores personales.

La reformulación de la aplicación, actualmente a nivel prototipo, permite no sólo llevar la interfaz a un modo gráfico mucho más atractivo y amigable, sino también significa pasar del mencionado esquema de terminal no inteligente a una arquitectura cliente - servidor en la modalidad presentación remota o lógica distribuida, según el tipo de instalación de las estaciones de trabajo de los usuarios finales.

La herramienta para implementar la parte del sistema sujeta al proceso de reingeniería, fue seleccionada atendiendo a las siguientes prestaciones que ofrece, a saber:

a) Una componente llamada "window painter" o pintador de pantallas, el cual permite crear gráficamente todas las ventanas que forman la interfaz para las aplicaciones.

Utilizando esta componente se consigue reducir el tiempo y el esfuerzo necesarios para desarrollar aplicaciones en ambiente de base de datos,presentadas al usuario en un modo totalmente gráfico. Se puede dibujar una gran variedad de objetos y controles gráficos, especificar los atributos visuales y propiedades de cada ventana y los de los objetos contenidos en ella, así como establecer relaciones lógicas o "ligaduras" entre las tablas y las columnas de la base de datos y los distintos objetos presentes en cada ventana.

Cada ventana está asociada con una base de datos o con una Super View, lo cual permite crear "SuperTables" que facilitan la selección de campos de las tablas de la base y que automáticamente toman un aspecto adecuado para el usuario, dentro de la ventana.

b) El módulo creador de aplicaciones. Éste proporciona un "front - end" gráfico para manejar los distintos módulos que forman las aplicaciones y ofrece una interfaz gráfica para definir y manejar los distintos módulos de la aplicación, tales como códigos fuente, archivos WIF, librerías de clase y otros componentes, todos ellos residiendo en el repositorio de la aplicación.

Esta herramienta permite la posibilidad de elegir de qué manera se compilará el código fuente, generando código C o código - p, según se decida en el plan de desarrollo y también ofrece los mecanismos para examinar y corregir errores.

c) El "debugger" gráfico interactivo que posibilita identificar y analizar errores en tiempo de ejecución de los programas. Con este depurador de programas se puede analizar el código ejecutable en forma interactiva y la depuración se desarrolla en un ambiente gráfico amigable para el programador.

d) Un lenguaje de base de datos poderoso y versátil. Se trata de un lenguaje de alto nivel, diseñado para soportar aplicaciones manejadas por eventos, que presenta una sintaxis flexible y poderosa que incluye un conjunto completo de construcciones orientadas a objetos, permitiendo la creación de clases base y clases derivadas, manejo de herencia simple y múltiple, polimorfismo y enlace dinámico.

e) Librerías de clase. Las librerías de clase son componentes reusables de las aplicaciones. Estas permiten incorporar con facilidad las reglas del negocio, establecer la conectividad con la base de datos y agregar nuevas funcionalidades a la aplicación con mínimo esfuerzo.

Se puede utilizar una librería intrínseca, una generada "a medida" de las necesidades de un desarrollo en particular, las cuales pueden ser escritas en el lenguaje nativo del producto, en C o en C++, o librerías de clase de terceras partes.

Resumiendo, la herramienta elegida es una familia de productos que cooperan entre sí para ofrecer un ambiente integrado de desarrollo orientado a objetos, tal que los desarrollos emprendidos con la misma soportan sin problemas un cambio en la escalabilidad.

Por otra parte, este producto cuenta con el software de conectividad necesario para el cliente, sin necesidad de agregar otro en el caso de que el sistema de gestión de base de datos instalado en los servidores sea el de la misma familia.

6. Conclusiones

En el trabajo que se informa concurren esfuerzos en lo conceptual desarrollados en el seno de los proyectos de investigación "Paralelismo: Software y hardware" de la FIUBA y "Tiempo real" de la UNSL, por un lado y los aspectos de desarrollo e implementación concretados en una empresa comercial.

Las prestaciones del viejo sistema eran satisfactorias pero, a juicio de los usuarios, presentaban como aspectos posibles de ser mejorados un incremento del rendimiento en lo que hace a los tiempos de respuesta y una mayor amigabilidad de la interfaz. Al trasladarse parte del esfuerzo computacional a las estaciones de trabajo de quienes realizan las consultas se acortaron los tiempos de ejecución de los algoritmos distribuídos y la herramienta utilizada permitió implementar una interfaz de características atractivas y funcionales.

Respecto de la forma en la que se reformuló parte de las aplicaciones se destaca la aplicabilidad del enfoque de Jacobson y Lindström y la practicidad de las propuestas de Ellis. El hecho de que la herramienta correspondiente a la primera versión de las aplicaciones manejara el concepto de "encapsulamiento" a través de los "stored procedures"

facilitó el trabajo para compatibilizar las interfaces del nuevo módulo y la parte no modificada del viejo sistema. Esta re ingeniería parcial evidencia un período de recupero de la inversión razonablemente corto. Una re ingeniería integral estaba fuera de la viabilidad económica.

En cuanto a las variantes verificadas luego de la reingeniería se señala que además de mejoras de los tiempos de respuesta y de un incremento de la amigabiliadad de la interfaz, en los casos de estaciones de trabajo de los usuarios finales con importantes prestaciones fue posible extender el enfoque de presentación remota al de lógica distribuída con mayores mejoras en la velocidad para responder a consultas de gran complejidad.

Referencias

Jacobson, I. y Lindström F., **"Re - engineering of old systems to an object - oriented architecture. OOSPLA '91**, pp. 340 - 350
Date, C.J. **"What is a Distributed Database System?"**, **Relational Database Writings** 1985 - 1989. Reading Mass.: Addison - Wesley (1990).
Shatz S. **"Develoment of Distribuited Software"**, MacMillan, 1993.
Ellis, J. **"Objectifying Real-Time Systems"**, SIGS Books, 1994
Mullender, Sape. **"Distributed Systems"**, Addison Wesley, 1993, 2d ed.

(*) UBA y UNSL
(**) UBA
(***) UNSL