INFORMÁTICA
junio del 2002

MODELO CONCEPTUAL DE UN PROYECTO DE SOFTWARE UTILIZANDO EL RAZONAMIENTO BASADO EN CASOS

Resumen / *Abstract*

Este trabajo tiene por objetivo aprovechar el conocimiento acumulado por los diseñadores (proyectistas) de software en nuevos proyectos, utilizando el razonamiento basado en casos (RBC), para obtener un modelo conceptual de UML (lenguaje de modelación unificado) partiendo de una lista de requerimientos candidatos definidos por los analistas. Con esta información, de carácter preliminar, los grupos de desarrollo pueden simplificar de manera notable algunos aspectos de las fases de modelación iniciales que permitan aprovechar el trabajo y las experiencias previas y aceleren por vías probadas el proceso total de elaboración de software. En este trabajo se describen las características generales del funcionamiento de un sistema que se desarrolla en el Centro de Estudios de Ingeniería de Sistemas (CEIS), que aborda estos temas de la ingeniería de software, utilizando la técnica de inteligencia artificial de razonamiento basado en casos.

The present work takes into account the knowledge accumulated by the software developers in different projects, using the based reasoning in cases (RBC). To obtain a conceptual model of UML (unified modeling language) leaving of a listing of requirements candidates defined by the analysts. With this information, of preliminary character, the design's groups can simplify in a remarkable way some aspects of the phases of initial modeling that allow to take advantage of the work and the previous experiences and hurry for proven roads the total process of software elaboration. The work was developed at Research Center for System Engineering (CEIS).

Palabras clave / *Key words*

Calidad de software, razonamiento basado en casos, inteligencia artificial.

Software quality, based reasoning in cases, artificial intelligence

INTRODUCCIÓN

Un punto importante en el desarrollo de proyectos es la captura de los requisitos, debido a que una detección errónea de estos trae consigo serios problemas en el desarrollo del proyecto, desde la afectación en el cumplimiento del cronograma planificado, hasta el deterioro de calidad del producto final y la insatisfacción de los clientes. Cualquier metodología de análisis y diseño para el desarrollo de sistemas tiene como punto de partida la captura de requisitos,[1] obtenidos por los analistas en interacción con los usuarios, que más tarde serán analizados y plasmados en herramientas propias de cada metodología, de manera que cubran las expectativas de los clientes y que se ajusten a las tendencias actuales de desarrollo de aplicaciones.

La obtención de requerimientos es un paso muy importante para el posterior desarrollo de las siguientes etapas,[1,2] pues un error en estas fases iniciales puede dar al traste con un sistema que no cumpla las expectativas de los usuarios y difícilmente aporte valor agregado al negocio para el que debe ser concebido.

Martha D. Delgado Dapena, *Ingeniera Informática, Máster en Ingeniería Informática, Instructora, Centro de Estudios de Ingeniería de Sistemas (CEIS), Instituto Superior Politécnico José Antonio Echeverría cujae, Ciudad de La Habana, Cuba*
e-mail:marta@ceis.ispjae.edu.cu
Veily Machado, *Alumna de 5to. año de Ingeniería en Informática, Instituto Superior Politécnico José Antonio Echeverría cujae, Ciudad de La Habana, Cuba*

El éxito de esta etapa depende en gran medida de la experiencia de los analistas y grupos de proyectos, así como de las habilidades desarrolladas por estos en el desarrollo de sistemas con características similares, mientras más conocimiento sobre este tema tenga el analista, más fácil le será proponer una solución adecuada y adaptarse a los cambios del medio, aún cuando este proceso de análisis de las experiencias anteriores es algo que realizará casi de manera intuitiva y hasta sin darse cuenta.

Con el desarrollo y difusión de las nuevas tecnologías de la información, los sistemas y lo que se espera de ellos será cada vez más complejo y por tanto será también más difícil capturar los requerimientos, tanto es así que muchas veces los equipos de proyectos comienzan a escribir el código sin haber precisado lo que se supone debe hacer este.

Se hace necesario entonces, combinar herramientas con la experiencia de los especialistas para afrontar los nuevos retos que impone diseñar sistemas eficientes y novedosos en las condiciones actuales, sumamente cambiantes. Esta necesidad es aún más imperiosa donde cada grupo e individuo tienen su propia forma de modelar y no siempre tienen todo el conocimiento y experiencia sobre el negocio para el que se va a representar el sistema.

A partir de los requisitos seleccionados como candidatos es necesario comprender el contexto del sistema y definir requisitos funcionales y no funcionales.[3] Una buena política puede ser obtener una representación de un modelo del negocio adecuado y a partir de este continuar con el análisis y diseño.

Para construir un adecuado modelo del negocio es importante definir un diagrama de procesos y las interacciones con los clientes, que permitan a los analistas y programadores comunicarse con los usuarios y definir con la mayor exactitud posible lo que se espera del sistema. Este diagrama de procesos puede ser un diagrama de casos de uso,[4] que deberá ser completado, para el caso de las metodologías orientadas a objetos, con una lista preliminar de clases,[5] lo que constituye un punto de partida para la etapa de análisis en la gran parte de estas metodologías.

Sería extremadamente conveniente y provechoso para los grupos de proyectos contar con herramientas que permitan utilizar la experiencia de otros especialistas en modelaciones de sistemas en el área del negocio que se está analizando y que además la solución propuesta por ellos contribuya de la misma forma a enriquecer y agilizar el trabajo futuro.

RAZONAMIENTO BASADO EN CASOS

El razonamiento basado en casos (RBC) es una técnica de inteligencia artificial que se basa en almacenar ejemplos a partir de los cuales se pueden proponer soluciones a nuevos problemas, utilizando la experiencia acumulada hasta el momento.[6] Esta técnica intenta llegar a la solución de nuevos problemas, de forma similar a como lo hacen los seres humanos.[7]

Cuando un individuo se enfrenta a un nuevo problema comienza por buscar en su memoria experiencias anteriores similares a las actuales y a partir de ese momento establece semejanzas y diferencias y combina las soluciones dadas con anterioridad para obtener una nueva solución. Este proceso es intuitivo y la persona lo realiza prácticamente sin darse cuenta.

Una vez que la persona tiene situadas un grupo de situaciones anteriores similares a la actual, analiza las variantes que se presentan en la nueva situación y cómo puede dar respuesta a estos cambios.

De manera resumida el proceso ocurre como sigue:

- El individuo buscó en su memoria casos similares.
- Intenta inferir una respuesta a partir del caso más similar que encontró.
- Tuvo que realizar algunas concesiones y ajustes para adaptar el caso anterior a la situación actual.

Finalmente, la solución obtenida no es igual a la anterior, pero cumple dos aspectos muy importantes, el primero da respuesta al nuevo problema y el segundo, ha enriquecido su experiencia anterior con la nueva solución.

El funcionamiento del RBC parte de estos principios y para ello comprende cuatro actividades principales:[8,9]

- Recuperar los casos más parecidos.
- Reutilizar el o los casos para tratar de resolver el nuevo problema.
- Revisar y adaptar la solución propuesta, en caso de ser necesario.
- Almacenar la nueva solución como parte de un nuevo caso.

Un nuevo problema se compara con los casos almacenados previamente en la base de casos y se recuperan uno o varios casos. Posteriormente se utiliza y evalúa una solución sugerida por los casos que han sido seleccionados con anterioridad, para ver si se aplica al problema actual.[10-12]

A menos que el caso recuperado sea igual al actual, la solución probablemente tendrá que ser revisada y adaptada, produciéndose un nuevo caso que será almacenado.

La elaboración de un sistema que emplea el RBC presenta dos problemas principales: el primero saber cómo almacenar la experiencia de tal forma que esta pueda ser recuperada en forma adecuada y el segundo, conseguir utilizar la experiencia previa en un problema actual.

La forma de representar y almacenar estas experiencias se realiza a través de casos. Un caso mantiene todos los atributos y características relevantes de un evento pasado. Estas características servirán como índices para la recuperación del caso futuro.

De acuerdo con la naturaleza del problema tratado se define la representación del caso, es decir, cuáles son los atributos

importantes, qué problemas serán tratados, cuál es la solución propuesta, etc. Además, es necesario definir el o los mecanismos de recuperación de casos.

Para la recuperación de los casos existen varias técnicas, entre las que se encuentra el método del vecino más próximo.[6] Este método parte de la definición de los atributos que serán considerados como importantes, para la recuperación, y los pesos asociados a estos. Con esta información se construye una función que será evaluada en los casos de la base de casos y con ella se seleccionará el caso más parecido al problema que se pretende resolver.

El caso más parecido será aquel que más próximo esté del problema actual, en cuanto a sus atributos, es decir, será el caso cuya evaluación de la función objetivo tome el mejor valor dentro de los casos contemplados en la base de casos existente, entendiendo como valor más próximo el valor de la función objetivo que denota menor diferencia entre el caso actual y el caso evaluado.

El RBC ha empezado a desempeña un papel importante en la ingeniería de software y del conocimiento, en la informática comercial y en la gestión del conocimiento.[13-15] Se observa un rápido incremento en el número de aplicaciones disponibles, especialmente de soporte técnico, atención al cliente, comercio electrónico y gestión del conocimiento corporativo.

La utilización de esta técnica en áreas de la ingeniería de software puede contribuir notablemente a elevar la calidad del proceso de desarrollo de software.

RAZONAMIENTO BASADO EN CASOS EN LA OBTENCIÓN DEL MODELO CONCEPTUAL DE UML

En el Centro de Estudios de Ingeniería de Sistemas (CEIS) se desarrolla un sistema que, a partir de los requerimientos funcionales y no funcionales del proyecto, propone un lista de clases, para el caso de los sistemas desarrollados utilizando el enfoque orientado a objetos y en particular la notación UML. El sistema propone un modelo conceptual acorde con la notación UML[16] y para ello contará con una base de ejemplos que puede ser enriquecida con la experiencia de expertos en las empresas desarrolladoras de software.

El uso de estas herramientas puede disminuir el tiempo de modelación del sistema y producir un software más eficiente y con mayor calidad, ya que partirá de una lista preliminar de clases que ha sido obtenido a partir de la experiencia propia del grupo y del trabajo realizado por otros grupos de proyecto con anterioridad. Para una empresa productora de software pueden ser de mucha utilidad sistemas con estas características, pues la proveerá de un banco de ejemplos basados en su propia experiencia de desarrollo y que podrá ser utilizado por nuevos analistas, lo que contribuirá a la formación de estos en las políticas de la organización.

UTILIZACIÓN DE LA BASE DE EJEMPLOS PARA LA DEFINICIÓN DEL MODELO CONCEPTUAL DE UML

A continuación se analiza cómo utilizar la base de ejemplos para obtener el modelo conceptual en una aplicación con las características descritas anteriormente. Un modelo conceptual en UML está constituido por un conjunto de conceptos y las relaciones que se establecen entre ellos. Este es un paso muy importante en cualquier metodología de análisis y diseño orientada a objetos, pues de aquí parte el diagrama de clases que modela la aplicación. Los errores que se comentan en la modelación de este diagrama pueden dar al traste con una aplicación incompleta o que no cumpla las expectativas de los clientes.

Se supone que se desea diseñar un sistema para una clínica multidisciplinaria, donde los pacientes pueden reservar citas para una especialidad determinada y a partir de ahí se hace un seguimiento de las consultas realizadas a este, así como de los análisis indicados por el médico.

En el momento de reservar la cita para un paciente y una especialidad dados se asigna un médico de esa especialidad que tenga espacio en su agenda y se le informa al paciente el día, la hora y el nombre del médico que lo atenderá en su cita. En el momento de la consulta, el médico puede decidir indicar al paciente que se realice análisis para chequear los resultados en la siguiente consulta y verificar la evolución de la enfermedad. Para los análisis de laboratorio se registra la orden de análisis en el momento en que el paciente acude a la clínica a realizárselos y posteriormente se almacenarán los resultados, lo que se hará por mediación de un operador del laboratorio (tabla 1)

TABLA 1
Lista simplificada de los requerimientos funcionales del sistema

No.	Requerimientos funcionales
1	Reservar cita para especialidad
2	Registrar un nuevo paciente
3	Registrar orden de análisis de laboratorio
4	Registrar resultados de la consulta o cita
5	Registrar resultados de análisis de laboratorio

Después de todas las entrevistas y estudios necesarios realizados por los analistas (haciendo una simplificación del negocio) se obtiene un listado de requerimientos funcionales del sistema,[1] como se muestra en la tabla 1.

A partir de la tabla 1 se podrá obtener una lista preliminar de clases,[17] que incluya los conceptos que modelan la situación descrita con anterioridad, pero no todos los analistas propondrán desde un inicio todas las posibles clases, dependerá de la experiencia que tengan en este tipo de aplicaciones, por ejemplo, una lista propuesta puede ser la siguiente:

- Paciente,
- Historia Clínica,
- Orden de análisis de laboratorio.
- Médico,
- Especialidad
- Cita reservada.

De la misma forma puede no incluir todas las relaciones que existen o incluir alguna que no sea significativa en la aplicación particular.

El sistema que utiliza el RBC propondrá, una lista preliminar de clases, como la siguiente:

- Paciente,
- Historia clínica,
- Descripción de análisis de laboratorio.
- Orden de análisis de laboratorio.
- Médico,
- Consulta,
- Especialidad
- Agenda
- Cita reservada.

Notar que la descripción de análisis de laboratorio, que contendrá la composición del análisis, es decir, la cantidad de cada compuesto que se necesita para realizar el análisis a paciente no aparece en la solución propuesta por el analista, y, por tanto pueden hacerse consideraciones en el desarrollo del proyecto sin tener en cuenta la existencia de esta clase y las relaciones que ella tiene con otras clases del modelo. Algo similar sucede con las clases agenda y consulta.

El modelo conceptual de UML que mostraría el sistema para este ejemplo sería el de la figura 1.

Es importante contar con un modelo conceptual adecuado para enfrentar las fases sucesivas en la elaboración de proyecto de software y lograr que el producto final se elabore en menor tiempo y con mayor calidad, de forma que se satisfagan los requerimientos de los clientes cumpliendo con las políticas de diseño establecidas por la empresa desarrolladora de software.

Si los analistas no cuentan con un sistema de este tipo llegarán a la solución, pero pueden incluir los nuevos conceptos en otro momento, cuando debía ser considerado desde el principio. Este es un caso muy simple, pero pudieran ser casos con relativa complejidad y actividades muy específicas con las que los desarrolladores de software no estén familiarizados, en estos casos sería mucho más evidente la necesidad de una herramienta con tales facilidades.

Después de la propuesta del sistema, el analista puede hacer ajustes a la solución e incorporar nuevas consideraciones de acuerdo con la situación concreta que está modelando y esa

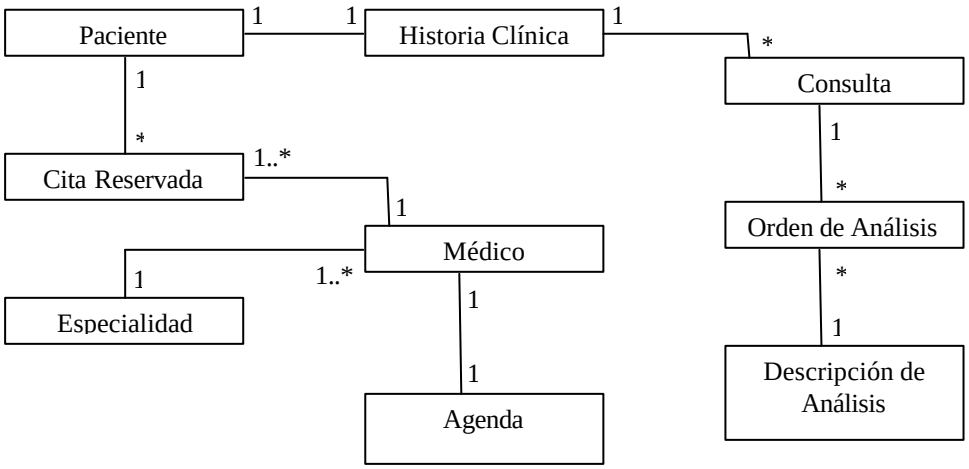

Fig.1 *Modelo conceptual utilizando notación UML para el ejemplo de la clínica.*

nueva solución se almacenará en la base de ejemplos como un nuevo caso para condiciones diferentes y con el que se podrá construir una nueva solución para otro problema del mismo tipo, es decir, la nueva solución será considerada experiencia acumulada a partir de este momento.

La base de ejemplos se enriquecerá y con ella el conocimiento y la experiencia disponible para el desarrollo de nuevos proyectos de software.

CARACTERÍSTICAS DEL SISTEMA EN DESARROLLO

El sistema que se desarrolla en el CEIS para obtener la lista preliminar de clases a partir de los requerimientos candidatos, hace uso del RBC. En la base de casos cada problema será almacenado de acuerdo con el área y el dominio donde clasifique, por ejemplo, el area puede ser medicina y el dominio laboratorio o consulta externa.

En esta aplicación cada caso contiene la siguiente información:

- Área en la que clasifica el sistema.
- Dominio dentro del área.
- Requerimientos funcionales.
- Requerimientos no funcionales.
- Lista preliminar de clases.
- Clases que se corresponden con cada requerimiento.
- Relaciones entre las clases.

Para agilizar la recuperación de los casos se utilizan índices, pues esto reduce el espacio de búsqueda lo que hace posible que la función, utilizada para determinar la similitud entre el caso almacenado y el problema, sea evaluada solo en un conjunto de casos y no en todos los que se encuentran almacenados.

Para obtener los casos que pueden contribuir a la resolución del nuevo problema se utilizan dos índices, uno por área y otro por dominio. Una vez determinado el conjunto de casos se aplica la función de similitud para ordenar los casos encontrados y posteriormente se muestra al usuario un modelo conceptual, que puede ser el resultado de un único caso almacenado en la base de casos o de la combinación de los más parecidos al problema planteado.

La función utilizada para determinar la similitud de un caso almacenado con el problema a resolver es la siguiente:

$$\text{Similitud } (O, A) = P_1 \cdot CRI + P_2 \cdot CRSI$$

donde:

- *CRI*: Cantidad de requerimientos iguales entre el caso actual y el almacenado.
- *CRSI*: Cantidad de requerimientos significativos iguales.
- P_1 y P_2 : Pesos que tendrán asignados los atributos *CRI* y *CRSI* respectivamente, y que serán decididos por el administrador de proyectos cada vez que se introduzca una nueva área o dominio.

Si el administrador de proyectos decide en algún momento que se debe incluir algún otro atributo en la función de similitud, puede hacerlo, especificando su peso respectivo. Los pesos suministrados deben estar entre 0 y 1 y ellos denotan el nivel de importancia de este dentro de la función objetivo, en el momento de seleccionar el modelo conceptual que más se parece a la situación actual, suministrada al sistema por el analista o grupo de proyecto.

El usuario del sistema -el analista- en el momento en que define los requerimientos de su problema debe indicar cuáles de ellos deberán ser considerados como significativos. Los requerimientos significativos son aquellos cuya importancia es medular en el desarrollo del software, en general, son requerimientos funcionales.

El caso más parecido es enriquecido con otros casos cuya evaluación de la función de similitud está en el rango del umbral definido por el administrador de proyectos. El caso más parecido es incorporado de manera íntegra a la solución que será propuesta al usuario y este es ampliado con las funcionalidades que posea el problema planteado y que aparezcan en otros casos de los seleccionados como casos similares, estos casos han sido ordenados por el valor de su función de similitud, lo que garantiza que se tomen las funcionalidades de los casos más parecidos. Una vez que han sido cubiertas todas las funcionalidades del problema el nuevo caso se muestra al analista para que realice las correcciones necesarias y el nuevo caso es incorporado a la base de casos.

El caso más parecido será incorporado de manera íntegra a la solución propuesta por tanto se le proponen al analista algunas funcionalidades que puede no haber considerado y que pueden ser parte de su problema. La decisión de incorporarlas o no al nuevo caso que será almacenado está en sus manos.

Si el caso más parecido fuera idéntico al problema planteado, no es necesario incorporarle nada más. Pudiera quedarse alguna funcionalidad del problema por cubrir porque no existiese un caso en la base de casos con esta funcionalidad, en cuyo caso tendrá que ser completada por el analista antes de que el nuevo caso pueda ser incorporado a la base de casos.

La base de casos es una base de datos relacional, compuesta por las siguientes entidades:

- Áreas.
- Dominios.
- Casos.
- Requerimientos o funcionalidades.
- Clases.
- Relaciones entre clases.

CONCLUSIONES

Este trabajo puede ser un punto de partida para introducir el RBC como ayuda a los desarrolladores de sistemas y contribuir así a lograr software con mayor calidad

aprovechando la experiencia acumulada, de manera que se puedan construir soluciones cada vez más complejas y con la rapidez que exige el creciente ritmo de la tecnología de la información.

De gran utilidad sería este sistema en empresas desarrolladoras de software, pues las dotaría de una base de ejemplos con soluciones dadas en la propia empresa que ayudarían a futuros desarrolladores a familiarizarse en poco tiempo con las características y políticas definidas en el trabajo de esta.

Este tipo de sistema contribuye de manera importante a elevar la calidad del proceso de desarrollo de software y por consiguiente la calidad del producto final, el cumplimiento de los cronogramas previstos y otros aspectos que influyen considerablemente en la eficiencia de las empresas de software en el mundo actual.

REFERENCIAS

1. **JACOBSON, I.:** *El Proceso unificado de desarrollo de software*, Addison Wesley Longman Inc., 2000.
2. **ÁLVAREZ, S.:** "METVISUALE", Centro de Estudios de Ingeniería de Sistemas, Instituto Superior Politécnico José Antonio Echeverría, Ciudad de La Habana, Cuba, 1999.
3. **———.:** "ADOOSIUML", Centro de Estudios de Ingeniería de Sistemas, Instituto Superior Politécnico José Antonio Echeverría, Ciudad de La Habana, Cuba, 2001.
4. **FOWLER, M.:** *UML Destilled*, second edition, Addison Wesley Longman Inc., 2000.
5. **LARMAN, G.:** *UML y Patrones*, Prentice Hall Hispanoamericana SA, 1999.
6. **KOLODNER, J.:** *Case-Based Reasoning*, Morgan Kaufmann Publishers Inc., 1993.
7. **SCHANK, R.:** *Inside Case-Based Explanation,* Lawrence Erlbaum Associates Inc., 1994.
8. **ALTHOFF, K.:** *Case-Based Reasoning. Handbook o Software Engineering and Knowledge Engineering* Fraunhofer Institute for Experimental Software Engineering (IESE), Kaiserslautern, Alemania, 2001.
9. **MANJARES, A.:** *Razonamiento basado en casos* Universidad Nacional de Educación a Distancia Departamento de Inteligencia Artificial, Madrid, España 2001.
10. **CUENA, J.:** *Sistemas inteligentes. Conceptos, técnicas métodos*, Publicación de la Facultad de Informática de la Universidad Politécnica de Madrid, Madrid, España, 1998.
11. **RICH, E.:** *Inteligencia artificial*, 2da. ed., McGraw-Hill Interamericana de España, 1994 .
12. **RIESBECH, CH.:** *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates Inc., 1989.
13. **ALTHOFF, K.:** "Potential Uses of Case-Based Reasoning in the Experience-Based Construction of Software Systems" *Proceedings of the 5th German Workshop in Case-Based Reasoning*, Kaiserslautern Alemania, Centre for Learning Systems and Applications, University of Kaiserslautern, 1997
14. **BERGMANN, R.:** *Developing Industrial Case Based Reasoning Applications. The INRECA Methodology*", Berlín Alemania. Springer-Verlag, 1999.
15. **WATSON, I.:** *Applying Case-Based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufmann Publishers, Inc. 1997.
16. **BOOCH, G:** *The Unified Modeling Language. User Guide* Addison Wesley Longman Inc.,1999.
17. **RUMBAUGH, J.:** *The Unified Modeling Language. Reference Manual*, Addison Wesley Longman Inc.,1999.