VI ENCUENTRO CHILENO DE COMPUTACION

Antofagasta, 9 al 13 de Noviembre de 1998

SOCIEDAD CHILENA DE CIENCIA DE LA COMPUTACION

UNIVERSIDAD CATOLICA DEL NORTE

Jornadas Chilenas de la Computación

Actas del VI Encuentro Chileno de Computación

Antofagasta, 9 al 13 de Noviembre de 1998, Chile

Organizan:

Sociedad Chilena de Ciencia de la Computación

Universidad Católica del Norte

Editor:

Carlos Pon Soto (Universidad Católica del Norte)

Mensaje del Presidente del Comité de Programa

Por primera vez, el Departamento de Ingeniería de Sistemas y Computación de la Universidad Católica del Norte es responsable de organizar el VI Encuentro Chileno de la Computación, junto a el XI Simposio Internacional en Aplicaciones de Informática y las I Jornadas Empresariales sobre Aplicaciones de Tecnologías de Información, y además es responsable por la implementación del XVIII International Conference of the Chilean Computer Science Society en conjunto con los representantes de la Universidad de Chile y de la Pontificia Universidad Católica de Chile, a realizarse entre los días 9 al 13 de Noviembre de 1998, en la ciudad de Antofagasta, Chile.

Para el VI Encuentro Chileno de la Computación se han seleccionado 31 trabajos de una recepción total de más de 50 resúmenes, provenientes tanto de autores nacionales como extranjeros, los que han sido revisados por distinguidos profesionales y académicos.

Con mucho agrado deseo dar la bienvenida a todos los participantes de la academia, científicos, y profesionales de diversos lugares del mundo, los que intercambiarán sus experiencias e ideas a través de las ponencias de trabajos en una amplia gama de tópicos de la informática. Es nuestra esperanza que la información aquí contenida sea un gran aporte a los conocimientos adquiridos en diferentes disciplinas, los que podrán ayudar a desarrollar nuevas ideas y permitirán a contribuir un mejor bienestar de la sociedad de nuestro mundo para enfrentar con optimismo al nuevo milenio.

En nombre de la Universidad Católica del Norte, deseo agradecer sinceramente la participación a todos los integrantes del comité editorial y a todos los autores de trabajos nacionales y extranjeros por su colaboración brindada. Disfruten al máximo su estadía en esta ciudad nortina.

Carlos Pon Soto
Presidente del Comité de Programa
VI Encuentro Chileno de Computación
Antofagasta, Chile, 1998

VI Encuentro Chileno de Computación

9 al 13 de Noviembre de 1998

Antofagasta, Chile

Comité de Programa

- Carlos Pon Soto, Universidad Católica del Norte, Presidente
- Héctor Beck Fernandez, Universidad de Tarapacá
- Alvaro Campos Ulloa, Pontificia Universidad Católica de Chile
- Marcos Chait Bollo, Universidad Católica del Norte
- Yusef Farrán Leiva, Universidad de Concepción
- Margarita García, Universidad de la Serena
- Luis Lobos Flores, Universidad Católica del Norte
- José Piquer Gardner, Universidad de Chile
- Ibar Ramírez Varas, Universidad de Tarapacá
- Oscar Saavedra Rodríguez, Universidad Técnica Federico Santa María
- Héctor Sosa Pollman, Universidad Católica del Norte
- Jorge Tabilo Alvarez, Universidad Católica del Norte
- Marcello Visconti Zamora, Universidad Técnica Federico Santa María

Comisión Organizadora

Juan Alfaro Toledo	**Empresa Minera de Mantos Blancos S.A.**
Mauricio Bravo Céspedes	**Ferrocarril Antofagasta a Bolivia**
Carlos Contreras Bustos	**Minera Michilla S.A.**
Hector Jorge Díaz Alvarado	**Codelco Chile, División Chuquicamata**
Patricio Peralta Varela	**Codelco Chile, División Radomiro Tomic**
Carlos Pizarro Quispe	**Minera Escondida Ltda**
Pablo Alvarez	**Industria Nacional de Cemento S.A.**
Ante Vrsalovic Ostojic	**SQM Nitratos S.A.**
Juan Carlos Reinoso Herrera	**Empresa Eléctrica de Antofagasta S.A.**
Oscar Carmona Morales	**Corporación de Capacitación de la Construcción**

Contenido

Bases de Datos

Pág.

Skip List, Arboles Comunes y Aboles Aleatorizados 3
Patricia Garcés M, Enrique Grájeda Ch, Marcelo Núñez del Prado (Universidad Mayor de San Simón, Bolivia)

Measurement for Database Fourth-Generation Languages 13
Antonio Martínez Hernández, Mario Platinni (Universidad de Castilla, España)

Algoritmo híbrido basado en FastMap y Kd-tree para búsqueda en espacios métricos 23
Karina Olmos Joffre, Rodrigo Villega Fiengo,Richard Jiménez Velasco (Universidad Mayor de San Simón, Bolivia)

Búsqueda Aproximada y General sobre Indices Invertidos 34
Corina Flores V. y Rosemary Torrico B. (Universidad de San Simón, Bolivia)

Una Propuesta para la Mantención de un Modelo de Datos en Tercera Forma Normal, Utilizando Forma Normal de Dominio y Clave 45
Angélica Urrutia Sepúlveda, Leoncio Jiménez Candia, Leonardo Chamorro M. (Universidad Católica del Maule, Chile)

Consultas Flexibles sobre Documentos 55
Ana Isabel Aguilera Faraco, Leonid José Tineo (Universidad de Carabobo, Venezuela)

Redes Neuronales

Experiencias en Programación Paralela de Redes Neuronales 69
Marcelo Puelles, Fidel Pérez, Pedro Robles, Carlos Pon, Marcos Chait (Universidad Católica del Norte, Chile)

Una estructura Jerárquica de Expertos para la Predicción Metereológica Utilizando Redes Neuronales 77
Isaac Moro, Rogelio Fullan, Benjamin Kuchen, Luis Alonso (Universidad de Valladolid, España)

Paralelismo y Sistemas Distribuidos

Pág.

Pre-Procesamiento Paralelo de Imágenes en Tiempo Real 85
Claudia Cordero, Luisa Yañez, Carlos Pon, Marcos Chait (Universidad Católica del Norte, Chile)

Construcción de Arreglos de Sufijos Distribuidos Basados en el Algoritmo de Manber Y Myers 95
Jaime Guillén R, José Reyes F,Luis Coronado M. (Universidad Mayor de San Simón, Bolivia)

Selección de Algoritmo de Scheduling según Clasificación de Aplicaciones 104
Mauricio Solar, Marc Feeley (Universidad de Montereal, Canadá)

A New General Framework Congestion Control Architecture for BISDN/ATM 113
Gerinal Isern, Giuseppe Sena, Dalila Megherbi (College of Computer Science, Northeastern University, USA)

Ingeniería de Software

Nuevos Requerimientos en Herramientas de Gestión de Configuraciones 125
José Antonio Gutiérrez de Mesa, Juan Antonio Rodrigo Yáñez (Universidad de Alcalá, España)

Validación del Proceso de Estudio del Dominio de un Modelo de Proceso de Software 137
Cecilia María Lasserre, Silvia Acuña, Viviana Quincoces (Universidad Nacional de Jujuy, Argentina)

Marco Metodológico para el Desarrollo de Modelos de Proceso de Software Centrados en la Cultura 147
Silvia Teresita Acuña y Graciela Barchini de Giménez (Universidad Nacional de Santiago del Estero, Argentina)

The Construction of a Software Capability Evaluation Tool 157
Alejandro Alan Silberman (Argentina)

Biblioteca de Reuso de Software 166
Olga Altamirano Loayza (Pontificia Universidad Católica de Chile)

Sistema Inteligente de Ayuda a la Optimización en la Asignación de Recursos 173
Fernando García Pérez, Vicente Martínez Orga (Universidad Politécnica de Madrid, España)

MANTEMA: Una Metodología para la Gestión Integral del Mantenimiento de Software 184
Francisco Ruiz González, Mario Gerardo Piattini V, Macario Polo U. (Universidad de Castilla, España)

Modelo de Estimación para Proyectos Cliente/Servidor Basado en el Método de Puntos de Función 194
Lautaro Guerra G, Cristián Vildósola A (Universidad Técnica Federico Santa María, Chile)

Herramientas de Desarrollo para Arquitecturas 204
Coral de la Barra, Gustavo Zurita
Universidad Mayor de San Simón, Bolivia

Teoría Informática y Lenguajes

Filtros para Búsqueda Aproximada en Texto 217
M. Leticia Blanco Coca, Iván Félix Fernández (Universidad Mayor de San Simón, Bolivia)

Análisis y Prueba de Algoritmos que Resuelven el Problema de la Cápsula Convexa 228
Coral de la Barra, Gustavo Zurita (Universidad Mayor de San Simón, Bolivia)

Construcción de un Preprocesador para el Lenguaje Java 238
Juan León Oros, Luis Mateu Brulé (Universidad de Chile)

Sistemas y Tecnologías de Información

Un Sistema de Gestión Docente Universitario: Una Experiencia en Acción 243
Yanko Ossandón N, Sabino Rivero F, Rodolfo Schmal S (Universidad de Tarapacá, Chile)

Las Potencialidades de la Lógica Difusa en Gestión de Empresas: Una aplicación para el Cálculo del VAN 252
Héctor Acevedo, Luis Ramirez (Universidad Técnica Federico Santa María, Chile)

Aplication of an Unbalance Growth Economic Model to Analyze the Information Technology Industry 260
Alejandro Alan Silberman (Argentina)

Modelización y Simulación de Sistemas de Producción Mediante Agentes 269
Alberto Gómez M, Josep Casanovas G, Luis Álvarez Ll, José González S. (Universidad de Extremadura y Universidad Politécnica de Cataluña, España)

Multimedia

Películas como Instrumentos de Evaluación del Autoaprendizaje 281
Julia Córdova, Paola Cifuentes, Juan Estanovich, Yanko Ossandón (Universidad de Tarapacá, Chile)

Multimedia Virtual Classroom 289
José González S, Alfonso Gazo C, Antonio Plaza M, Alberto Gómez M, Marisol Sánchez A (Universidad de Extremadura, España)

Metodología para Desarrollo de Sistemas con Tecnología Multimedia 299
Oscar Saavedra R, Rodrigo Montaner G.(Universidad Técnica Federico Santa María, Chile)

MANTEMA: Una metodología para la gestión integral del mantenimiento del software. [1]

Francisco Ruiz González (*)
Mario Gerardo Piattini Velthuis (*)
Macario Polo Usaola (*)
(*) Grupo ALARCOS. Departamento de Informática. Universidad de Castilla-La Mancha.
Escuela Superior de Informática. Ciudad Real, 13071. España.
{fruiz,mpiattin,mpolo}@inf-cr.uclm.es
tlf: +34-926295300, fax: +34-926295354

Resumen:

El mantenimiento se ha convertido en la etapa más costosa en el ciclo de vida del software. En muchos casos supone hasta el 80% del total de los recursos, imposibilitando de esta manera el desarrollo de nuevos sistemas. A pesar de ello, existen pocos métodos, herramientas y técnicas desarrollados exclusivamente para el mantenimiento. En este trabajo presentamos el proyecto MANTEMA cuyo objetivo es desarrollar una metodología específica para el mantenimiento del software basada en el estándar ISO 12207. Este proyecto está siendo desarrollado en colaboración con ATOS-ODS, multinacional europea de servicios informáticos. Por último, comentaremos algunos aspectos sobre la medida de la mantenibilidad de productos software que han sido abordados en el citado proyecto.

Tópico: Ingeniería del Software.

Palabras Claves: Mantenimiento del Software, Mantenibilidad, ISO 12207, Metodologías de Mantenimiento.

1. Introducción

1.1 Los costes del mantenimiento

Múltiples estudios señalan que el mantenimiento es la parte más costosa del ciclo de vida del software. Estadísticamente está comprobado que el coste de mantenimiento de un producto software a lo largo de toda su vida útil supone más del doble que los costes de su desarrollo [Schach, 1992]. La tendencia es creciente con el paso del tiempo, tal como puede observarse en la tabla 1.

---

[1] Este trabajo ha sido parcialmente financiado por el proyecto MANTEMA que se está desarrollando en colaboración con la empresa ATOS ODS S.A. (dentro de la iniciativa ATYCA de la Dirección General de Tecnología y Seguridad Industrial del Ministerio de Industria y Energía de España) y por el proyecto MANTICA (CICYT-FEDER nº 1FD97-0168).

Referencia	Fechas	% Mantenimiento
[Pressman, 1993]	años 70	35%-40%
[Lientz y Swanson, 1980]	1976	60%
[Pigoski, 1996]	1980-1984	55%
[Pressman, 1993]	Años 80	60%
[Rock-Evans y Hales, 1990]	1987	67%
[Schach, 1990]	1987	67%
[Pigoski, 1996]	1985-1989	75%
[Frazer, 1992]	1990	80%
[Pressman, 1993]	Años 90 (prev.)	90%

Tabla 1. Evolución de los costes del mantenimiento según diversos autores.

Algunos autores [Frazer, 1992] estiman que la situación puede llegar a ser casi insostenible. Existen empresas que se acercan a porcentajes del 95% de los recursos dedicados al mantenimiento, con lo cual se hace imposible el desarrollo de nuevos productos software. Esta situación se conoce como *Barrera de Mantenimiento.*

Son varias las causas de que se requiera mucho trabajo de mantenimiento [Osborne y Chikofsky, 1990]. En primer lugar, una gran cantidad del software que existe actualmente ha sido desarrollado hace más de 10 años y se construyó con restricciones de tamaño y espacio de almacenamiento y se desarrolló con herramientas tecnológicamente desfasadas. En segundo lugar, estos programas han sufrido una o varias migraciones a nuevas plataformas o sistemas operativos. Y por último, han experimentado múltiples modificaciones para mejorarlos y adaptarlos a las nuevas necesidades de los usuarios. El resultado de todo ello es la existencia de sistemas software con una baja calidad (diseño pobre de las estructuras de datos, mala codificación, lógica defectuosa y documentación escasa) pero que tienen que seguir funcionando en la actualidad.

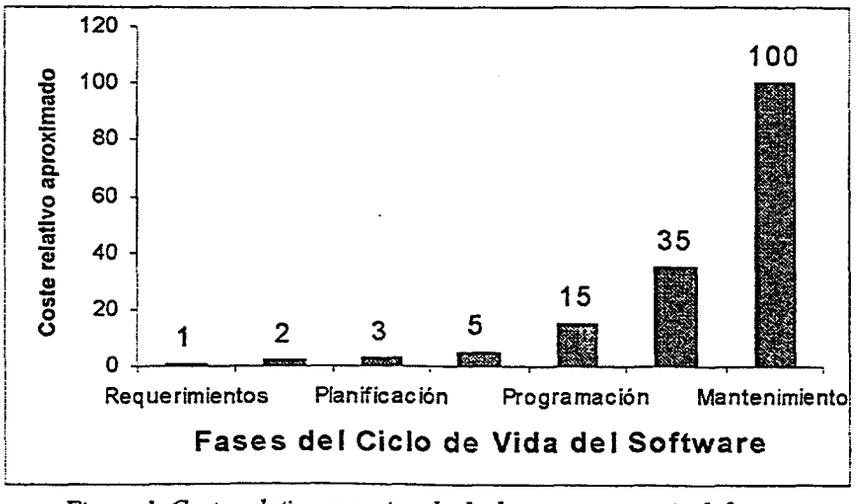

Figura 1. Coste relativo aproximado de detectar y corregir defectos.

Una causa directa de los grandes costes del mantenimiento es que el coste relativo aproximado de reparar un defecto aumenta considerablemente en las últimas etapas del ciclo de vida del software [Boehm, 1981] de forma que la relación entre el coste de detectar y reparar un defecto en la fase de análisis de requisitos y en la fase de mantenimiento es de 1 a 100 respectivamente (ver figura 1).

1.2 Gestión del mantenimiento

En la figura 2 se muestran las necesidades del ciclo de vida del software en las fases de desarrollo y mantenimiento según un estudio desarrollado a nivel europeo por ATOS ODS [Piattini y otros, 1998b]. Se observa que son prácticamente divergentes y distintas. Por tanto, se hace necesario abordar el problema del mantenimiento dándole la importancia que le corresponde y teniendo en cuenta sus características especiales y diferenciadas con las de la fase de desarrollo.

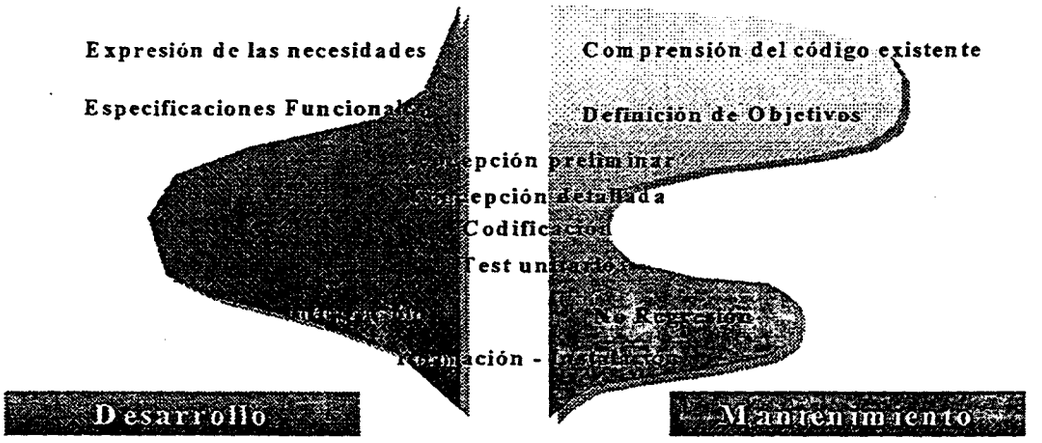

Figura 2. Necesidades en la fase de desarrollo vs fase de mantenimiento.

Para reducir los importantes costes del mantenimiento y abordar sus características particulares es importante emplear una gestión estructurada y organizada del proceso de mantenimiento del software. Este *Mantenimiento Estructurado* [Pressman, 1993] o *Gestión del Mantenimiento* aparece como resultado de la aplicación de una metodología de ingeniería del software a esta etapa del ciclo de vida del software. Cuando no se procede de esta manera, se sufren las consecuencias de la falta de metodología: dolorosa evaluación del código (muchas veces poco legible), complicada comprensión del sistema por la pobre documentación interna (desconocimiento de la estructura del programa, las estructuras de datos globales, las interfaces y otros requisitos de diseño y/o rendimiento), dificultad para descubrir las consecuencias de los cambios en el código y, por último, imposibilidad de realizar pruebas de regresión (repetición de pruebas anteriores) al no existir ningún registro de pruebas.

En los últimos años se ha realizado un considerable esfuerzo en la comunidad internacional para elaborar estándares para el ciclo de vida del software. Fruto de estos trabajos ha sido la publicación de diversos estándares por parte de IEEE y de ISO que tienen relación directa o indirecta con el problema del mantenimiento del software:

- Para los procesos del ciclo de vida del software: ISO 12207 [ISO/IEC, 1995] e IEEE 1074 [IEEE, 1995].
- Para la calidad del software y sus métricas: ISO 9126 [ISO/IEC, 1998] e IEEE 1061 [IEEE, 1992].
- Para el mantenimiento del software: IEEE 1219 [IEEE, 1993]. [2]

Todos estos estándares son de utilidad para elaborar una metodología para la gestión integral del mantenimiento del software. Los estándares para los procesos del ciclo de vida del software nos permiten encajar y asociar el proceso de mantenimiento con los demás procesos existentes para el software. Los estándares de calidad del software interesan en mantenimiento del software porque los factores de calidad del software (especialmente la complejidad y la mantenibilidad) inciden directamente sobre el esfuerzo de mantenimiento necesario.

2. Metodología MANTEMA

Aunque existen muchas metodologías para abordar el desarrollo del software, no existe ninguna específica para la fase del mantenimiento. Acabamos de ver que sus características y necesidades son diferentes, por tanto, se justifica totalmente la necesidad de crear metodologías para el mantenimiento del software. Además, esta es una necesidad manifestada por las empresas de servicios informáticos relacionados con el mantenimiento (*outsourcing*, externalización, etc.).

La metodología que presentamos en este trabajo está adaptada al estándar ISO 12207, de forma que el proceso de mantenimiento consta de las siguientes actividades:

1. Implementación del proceso: en esta actividad se desarrollan los planes correspondientes para llevar a cabo las actividades y tareas del mantenimiento. También se deben definir los procedimientos necesarios para la gestión de problemas y petición de modificaciones (empleando el proceso de resolución de problemas), e implementar el proceso de gestión de configuración [Piattini y otros, 1998b] para gestionar las modificaciones al sistema existente.

2. Análisis de problemas y modificaciones: esta actividad consiste en analizar los problemas o peticiones de modificación con el fin de evaluar su impacto en el sistema y la organización existentes, determinando el tipo de modificación (preventiva, correctiva, etc.), su alcance (tamaño, coste, tiempo, etc.) y su criticidad (rendimiento,

[2] En las fechas de elaboración de este trabajo está en fase de construcción la propuesta de estándar ISO para el mantenimiento del software.

seguridad, etc.). La organización encargada del mantenimiento debe también verificar el problema, elaborar distintas opciones para implementar las modificaciones, documentar el problema o la petición de modificación, así como los resultados del análisis y las opciones de implementación. Por último, deberá obtener la aprobación para la opción seleccionada.

3. Implementación de las modificaciones: en esta actividad se incluyen todas las tareas relativas a determinar qué documentación, unidades de software y versiones deben modificarse y se utiliza el proceso de desarrollo para implementar las modificaciones. Pero los requisitos del proceso de desarrollo deben complementarse de la siguiente manera:
   - Se deberán definir y documentar los criterios de evaluación y prueba para probar y evaluar las partes del sistema (unidades, componentes y elementos de la configuración) modificadas y no modificadas.
   - Se deberá asegurar la completa y correcta implementación de los requisitos nuevos y modificados, y que no se vean afectados los requisitos originales no modificados. También se deberán documentar los resultados de las pruebas.

4. Revisión y aceptación del mantenimiento: esta actividad consiste en la revisión de la integridad del sistema modificado, que llevarán a cabo la organización encargada del mantenimiento junto con la organización que autorizó la modificación. La organización encargada del mantenimiento deberá obtener también la aprobación de terminación satisfactoria de la modificación.

5. Migración: en el caso de una migración (cambio a un entorno diferente), se debe desarrollar un plan de migración en el que se especifiquen al menos las siguientes cuestiones:
   - análisis de requisitos y definición de la migración,
   - desarrollo de herramientas de migración,
   - conversión del software y de los datos,
   - ejecución de la migración,
   - verificación de la migración, y
   - soporte del entorno antiguo en el futuro.

6. Retirada de software: es necesario desarrollar y documentar un "plan de retirada", que aborde cuestiones como las siguientes:
   - cese de soporte total o parcial después de un cierto tiempo,
   - archivo del producto software y su documentación asociada,
   - responsabilidad sobre cuestiones de soporte residual futuro,
   - transición al nuevo producto, y
   - accesibilidad de copias de datos.

Cada actividad está formada por varias tareas. En la tabla 2 se muestra un resumen de todas ellas.

ACTIVIDADES	*TAREAS*
IMPLEMENTACIÓN DEL PROCESO	DESARROLLAR LOS PLANES DE MANTENIMIENTO
	DEFINIR LOS PROCEDIMIENTOS DE PETICIÓN DE MODIFICACIÓN
	IMPLEMENTAR LA GESTIÓN DE CONFIGURACIONES
ANÁLISIS DE PROBLEMAS Y MODIFICACIONES	EVALUAR EL IMPACTO
	VERIFICAR EL PROBLEMA
	ELABORAR ALTERNATIVAS
	DOCUMENTAR EL PROBLEMA
	OBTENER LA APROBACIÓN
IMPLEMENTACIÓN DE LAS MODIFICACIONES	DETERMINAR LOS OBJETOS A MODIFICAR
	DESARROLLAR LAS MODIFICACIONES
REVISIÓN Y ACEPTACIÓN DEL MANTENIMIENTO	REVISAR LA INTEGRIDAD
	OBTENER LA APROBACIÓN
MIGRACIÓN	ASEGURAR AJUSTE A LA NORMA
	DESARROLLAR PLAN
	NOTIFICAR LA FUTURA MIGRACIÓN
	EJECUTAR EN PARALELO
	NOTIFICAR MIGRACIÓN
	REALIZAR REVISIÓN POST-OPERACIÓN
	ARCHIVAR DATOS DEL ENTORNO ANTIGUO
RETIRADA	DESARROLLAR PLAN
	NOTIFICAR FUTURA RETIRADA
	EJECUTAR EN PARALELO
	NOTIFICAR RETIRADA
	ARCHIVAR DATOS DEL ENTORNO ANTIGUO

Tabla 2. Principales actividades y tareas del mantenimiento.

Además del estándar ISO 12207, la metodología MANTEMA también tiene en cuenta otros estándares que pueden ser útiles para el problema del mantenimiento del software, como por ejemplo (Software Process

Improvement and Capability Determination) [El Emam y otros, 1997] , que está siendo desarrollado por ISO tomando como fuente de inspiración el "modelo de madurez de capacidad" (CMM) desarrollado por el SEI (Software European Institute). Las indicaciones de SPICE se utilizan para valorar el proceso de mantenimiento de una organización, cuyo propósito es "*gestionar la modificación, migración y retirada de componentes del sistema (como hardware, software, operaciones manuales y redes) en respuesta a peticiones del usuario*".

La metodología MANTEMA también se basa en la experiencia propia de ATOS ODS a lo largo de muchos años de dar servicio de mantenimiento software a grandes clientes. Por esta razón, entre las herramientas que se están diseñando, se incluye una versión ampliada de MANUTEN CONDUCT, la herramienta para la gestión integral del mantenimiento de la citada empresa.

3. Mantenibilidad del software

La mantenibilidad, o facilidad de mantenimiento, es el atributo de calidad del software que más directamente influye en los costes y necesidades del mantenimiento. Por esta razón, el proyecto MANTEMA aborda también la definición de métodos y herramientas para medir la mantenibilidad, lo que, a su vez, nos permitirá realizar una evaluación de los costes del mantenimiento. Para ello, en MANTEMA se sigue el modelo de calidad del software propuesto en la última versión del estándar ISO 9126 [ISO/IEC, 1998] y definimos la mantenibilidad como la capacidad de un producto software para ser modificado. Estas modificaciones pueden incluir correcciones, mejoras o adaptación del software a cambios en el entorno, en los requerimientos o en las especificaciones funcionales.

La mantenibilidad se considera dividida en seis subcaracterísticas:

a) *Analizabilidad*: Capacidad del producto software de diagnosticar sus deficiencias o causas de fallos, o de identificar las partes que deben ser modificadas.
b) *Cambiabilidad*: Capacidad del producto software de permitir implementar una modificación especificada previamente. La implementación incluye los cambios en el diseño, el código y la documentación.
c) *Estabilidad*: Capacidad del producto software de minimizar los efectos inesperados de las modificaciones.
d) *Facilidad de prueba*: Capacidad del producto software de permitir evaluar las partes modificadas.
e) *Conformidad*: Capacidad del producto software de satisfacer los estándares o convenciones relativas con la mantenibilidad.

Utilizar estas medidas durante el desarrollo ayuda a determinar la cuantía en que se está incorporando en el software el objetivo de mantenibilidad. Una vez el software ha sido desarrollado, las medidas pueden guiar durante el proceso de mantenimiento, bien para evaluar el impacto de un cambio (mantenibilidad de la nueva configuración

obtenida), o para realizar un análisis comparativo entre varias propuestas o aproximaciones diferentes para realizar una modificación requerida por los usuarios.

Las herramientas de medida de la mantenibilidad que se están definiendo en el proyecto MANTEMA incluyen un conjunto de métricas internas y externas a partir de las indicaciones de la norma ISO 9126. También se está teniendo en cuenta la experiencia en casos reales de mantenimiento externo y *outsourcing* de la empresa ATOS ODS. Esto nos permite una validación de las métricas y herramientas en entornos reales totalmente significativos. Algunos de estos casos reales donde se están verificando o se verificarán dichas herramientas, junto con la propia metodología MANTEMA, son:

- Argentaria (Corporación Bancaria de España): 11403 programas; 4757000 LDC's.
- Fiat Auto (Italia): 400000 horas/año de mantenimiento.
- Ayuntamiento de Valencia (España): 16500 objetos Natural; 1990000 LDC's.

La mantenibilidad del software es una característica que debe establecerse como objetivo en las fases iniciales del ciclo de vida para reducir las posteriores necesidades de mantenimiento [Pigoski, 1996]. También se debe tener como objetivo durante la fase de mantenimiento para reducir los efectos laterales y otros inconvenientes ocultos que pueden producir los conocidos efectos de "bola de nieve" (un cambio en el software obliga a cambios posteriores que a su vez obligan a otros cambios y así sucesivamente). Por esta razón, en MANTEMA se exige realizar un plan de mantenibilidad para controlar el proceso del cambio del software. Dicho plan está orientado a controlar las acciones a realizar con un doble fin: disminuir el riesgo de introducir errores en cada intervención de mantenimiento que se realice; y controlar el aumento del esfuerzo necesario cada vez que se introduce un cambio.

En suma, se trata de conocer la situación en la que se encuentra el producto software para poder decidir mejor cómo gestionar cada petición de mantenimiento.

4. Conclusiones y líneas de trabajo futuras

El mantenimiento supone una parte fundamental de los costes a lo largo del ciclo de vida de un producto software. Además no existen metodologías diseñadas teniendo en cuenta las necesidades y características particulares del mantenimiento, que son diferentes de las que se producen en las fases de desarrollo. Por tanto, se hace necesario disponer de metodologías para el mantenimiento del software, así como herramientas que las soporten y faciliten su utilización.

En este trabajo presentamos la metodología MANTEMA que estamos desarrollando en el proyecto del mismo nombre. Esta metodología está basada en los estándares actuales de ciclo de vida y calidad del software. También hemos justificado la gran importancia que en estas metodologías tiene la característica de mantenibilidad, y lo interesante que es disponer de herramientas para poder medirla y poder predecir y/o reducir los costes de mantenimiento futuros.

El proyecto MANTEMA comenzó en septiembre de 1997 y durará hasta octubre de 1999. Hasta ahora se han realizado los estudios de metodologías, normas y herramientas existentes y se ha analizado su utilidad para los objetivos del proyecto. Se ha definido la primera versión de la metodología y se está empezando a validar empíricamente en casos reales. También se ha comenzado a estudiar el problema de cómo medir la mantenibilidad de los productos software, especialmente de las bases de datos, área que ha sido tradicionalmente descuidada [Polo y otros, 1998], [Piattini y otros, 1998a].

En las siguientes etapas del proyecto se refinará la metodología en base a los resultados experimentales y se desarrollará un conjunto de herramientas para utilizar y aplicar la metodología. Por último, se realizará la adaptación a varios entornos específicos, como COBOL, ORACLE FORMS, C++, Java y sistemas de gestión de bases de datos objeto-relacionales (SQL3).

Referencias:

[Boehm, 1981] Boehm, B.W., "*Software Engineering Economics*". Ed. Prentice-Hall, USA. 1981.

[El Emam y otros, 1997] El Emam y otros, *SPICE. The Theory and practice of Software Process Improvement and Capability Determination*. IEEE Computer Society, Los Alamitos, CA. 1997.

[Frazer, 1992] Frazer, A., "Reverse Engineering- hype, hope or here?". En: *Software Reuse and Reverse Engineering in Practice*. Ed. Chapman & Hall, 1992.

[IEEE, 1992] IEEE, std 1061: "*Standard for a Software Quality Metrics Methodology*". IEEE Computer Society Press. Estados Unidos, 1992.

[IEEE, 1993] IEEE, std 1219: "*Standard for Software Maintenance*". IEEE Computer Society Press. Estados Unidos, 1993.

[IEEE, 1995] IEEE, std 1074: "*Standard for Developing Software Life Cycle Processes*". IEEE Computer Society Press. Estados Unidos, 1995.

[ISO/IEC, 1995] ISO 12207: "*Information Technology-Software Life Cycle Processes*". Suiza, 1995.

[ISO/IEC, 1998] ISO 9126: "*Software Quality Characteristics and Metrics*". Canadá, 1998.

[Lientz y Swanson, 1980] Lientz, B.P. y Swanson, E.F., "*Software Maintenance Management*". Ed. Addison-Wesley, 1980.

[Osborne y Chikofsky, 1990] Osborne, W.M., Chikofsky, E.J., "Fitting Pieces to the Maintenance Puzzle". En *IEEE Software*. Enero 1990, pgs. 10-11.

[Piattini y otros, 1998a] Piattini, M., Calero, C., Polo, M. y Ruiz, F., "Maintainability in Object-Relational Databases". En *European Software Measurement Conference*. Amberes. Mayo 1998, pgs. 223-230.

[Piattini y otros, 1998b] Piattini, M., Ruiz, F., Polo, M., Bastanchury, T., Fernández, I., Martínez, M.A. y Villalba, J.. "*Mantenimiento del Software. Conceptos, Métodos, Herramientas y Outsourcing*". Ed. RA-MA. Madrid, 1998.

[Pigoski, 1996] Pigoski, T.M., "*Practical Software Maintenance. Best Practices for Managing Your Investment*". Ed. John Wiley. Estados Unidos, 1996.

[Polo y otros, 1998] Polo, M., Calero, C., Ruiz, F. y Piattini, M. "Métricas de Calidad y Complejidad de Bases de datos". *III Jornadas de Ingeniería del Software*. Murcia, España. 1998.

[Pressman, 1993] Pressman, Roger S. "*Ingeniería del Software, un enfoque práctico*". 3ª edición. Editorial McGraw-Hill, 1993.

[Rock-Evans y Hales, 1990] Rock-Evans, R. y Hales, K., *Reverse Engineering: Market, Methods and Tools*. 1990.

[Schach, 1990] Schach S.R., *Sofware Engineering*. Ed. Irwin & Aksen. USA 1990.

[Schach, 1992] Schach, S.R., *Practical Sofware Engineering*. Ed. Irwin & Aksen. USA 1992.