risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido / Recibido: 24/10/2010
Aceitação / Aceptación: 03/12/2010

SAM - Sistema Automatizado del Método MECAP para Especificar Casos de Prueba

Edumilis Méndez[1], María A. Pérez[1], Kenyer Domínguez[1], Luis E. Mendoza[1].

{emendez, movalles, kdoming, lmendoza}@usb.ve

[1] Laboratorio de Investigación en Sistemas de Información (LISI), Departamento de Procesos y Sistemas, Universidad Simón Bolívar, Apartado Postal 89000, Caracas 1080-A, Venezuela.

Resumen: Existen cuatro elementos que son relevantes al momento de definir las pruebas: Confiabilidad, Costo, Tiempo y Calidad. El tiempo de desarrollo y el costo del producto se incrementan cuando se desean pruebas confiables y un software de calidad. Pero ¿qué se puede hacer para que los involucrados comprendan que las pruebas deben ser vistas como una red de seguridad? Si la calidad no se contempla antes de comenzar las pruebas, entonces ella no estará cuando se éstas terminen. El objetivo de este artículo es presentar la herramienta, SAM – Sistema Automatizado del Método MECAP que permite especificar Casos de Prueba a partir de Casos de Uso incorporando elementos que promueven la verificación y validación de la trazabilidad entre la Gestión de Requerimientos, el Análisis y Diseño y las Pruebas. SAM soporta el proceso de pruebas de forma automatizada, mejorando la confiabilidad de las mismas.

Palabras clave: Pruebas de Software; Calidad del Software; Casos de Prueba; Herramienta de Pruebas.

Abstract: There are four elements that are relevant when test are defined, Reliability, Cost, Time, and Quality. Development time and product cost increases when you want reliable tests and quality software. But, what can you do to make that stakeholders understand that the tests must be viewed as a security net? If the quality is not considered before starting the tests, then it will not be when they finish. The goal of this paper is to present the tool, SAM - Automated Systems of MECAP Method, which allow to specify test cases from Use Cases incorporating elements that promote the verification and validation of the traceability among Requirement Management, Analysis and Design, and Test. SAM automatically support the testing process, improving the test reliability.

Keywords: Software Test; Software Quality; Test Cases; Test Tool.

1. Introducción

La Disciplina de Pruebas se aborda desde las primeras fases, esto quiere decir que las pruebas se deben comenzar a planificar y además se debe establecer cuál es la estrategia de pruebas a seguir una vez que han aprobado todos los Casos de Uso (CU). Las pruebas del software contraponen dos aspectos importantes: La Verificación y la Validación (V&V). Durante la Verificación se responde a si ¿Se está construyendo el producto correctamente? Así mismo, la Validación establece si ¿Se está construyendo el producto correcto?

El objetivo de la Disciplina de Pruebas es evaluar la calidad del producto a lo largo de todo el ciclo de vida apoyándose en un conjunto de buenas prácticas, entre las que destacan según Leffingwell & Widrig (2006): (a) Encontrar y documentar las fallas en el producto de software: defectos y problemas. (b) Evaluar las suposiciones hechas en las especificaciones de requerimientos y diseño a través de demostraciones concretas. (c) Verificar que el producto de software trabaja según el diseño. (d) Validar que los requerimientos son implementados apropiadamente.

Kruchten (2000), Pressman (2002), Pfleeger (1998) y Sommerville (2000) afirman que el proceso de ejecución de Pruebas debe ser considerado durante todo el ciclo de vida de un proyecto, para así obtener un producto de alta calidad. Su éxito dependerá del seguimiento de una Estrategia de Prueba adecuada. La Estrategia de Prueba de Software integra un conjunto de actividades que describen los pasos que hay que llevar a cabo en un proceso de prueba, tomando en consideración cuánto esfuerzo y recursos se van a requerir, con el fin de obtener como resultado una correcta construcción del software (Pressman, 2002).

Las empresas desarrolladoras de software invierten en las pruebas entre el 30% y 50% del costo total del software (Pressman, 2002). Esto representa un esfuerzo considerable que indica que es una disciplina importante y que los resultados que ella arroje así como su confiabilidad pueden impactar sobre la percepción del cliente o usuario en cuanto a la calidad del software que se le está entregando.

Las pruebas de software consisten en la dinámica verificación del comportamiento de un programa en un conjunto finito de casos de prueba, convenientemente seleccionados de la ejecución de un dominio contra el comportamiento esperado. ¿cómo se evidencia que se efectuaron las pruebas de un software? ¿cuáles fueron los resultados? ¿quién las llevó a cabo? ¿cuándo? ¿se alcanzó el porcentaje de satisfacción esperado? Las respuestas se obtienen a través de la documentación.

El estándar IEEE para Documentación de las pruebas de software (IEEE829-98) proporciona una buena descripción de los documentos de prueba y de su relación entre sí e incluso con el proceso de prueba. SWEBOK (2004) señala que los documentos de pruebas pueden incluir, entre otros, la especificación de los Casos de Prueba (CP). Es por ello que al hacer las pruebas funcionales, hay tres cuestiones claves (Utting y Legeard, 2007): El diseño del caso de prueba, realizar los ensayos y análisis de los resultados, y la verificación de cómo la prueba cubre las necesidades. Cada CP se define por el contexto de prueba, una situación, y algunos criterios de falla o de aprobación. Perry (2006), presenta una guía completa de cómo llevar a cabo un proceso de prueba efectivo. Inclusive, habla de los CP y propone una plantilla para CP, la cual incluye

algunos aspectos que nosotros hemos considerado para nuestro método. Por ejemplo, usar ID (Identificador) para el CP y para el UC. Lewis (2000) propone una plantilla para CPs, donde incorpora condiciones, ambiente, versión y sistema.

Algunos autores como Pinkster et al., 2006 indican que una mejora subsecuente en la calidad de las pruebas es posible si los requerimientos son tomados como la base de las pruebas, denominándola Pruebas basadas en Requerimientos. El probador puede centrarse en los requerimientos de mayor prioridad, y a los problemas se les da la misma prioridad que al requisito relacionado con el CP y el problema que se ha encontrado. De esta forma, los desarrolladores pueden ver los aspectos que necesitan resolverse primero.

Así mismo, el Profile del Unified Modeling Language (UML) para Pruebas (2005) indica que un contexto de la prueba es sólo un CP de alto nivel. Un CP siempre devuelve un veredicto. El veredicto puede ser arbitrado —calculado por el árbitro— o no arbitrado (es decir, siempre provisto por el comportamiento de prueba). Algunos de los beneficios de la matriz de prueba/requerimientos son (Lewis, 2000): correlación de las pruebas y los scripts con los requisitos; facilita el estado de las revisiones y actúa como un mecanismo de trazabilidad a lo largo del ciclo de desarrollo; incluye el diseño y ejecución de prueba. El Plan de Calidad de Planificar-Hacer-Verificar-Actuar (PHVA) se aplica al proceso de pruebas de software. El método propuesto ayuda a aplicar PHVA.

A diferencia de las iniciativas anteriores, el método MECAP – Método para Especificar Casos de Prueba (Méndez, Pérez & Mendoza, 2007 y 2008) incorpora todas las ideas citadas anteriormente con la finalidad de obtener un método que soporte todos los elementos que conforman una estrategia de pruebas. Ejecutar MECAP de forma manual representa un esfuerzo considerable, así lo evidencian resultados preliminares en varios proyectos por lo que surgió la necesidad de automatizarlo a través del uso de una herramienta que garantizara todos los aspectos provistos con el método.

En este sentido, el objetivo de este artículo es presentar la herramienta SAM (Sistema Automatizado del Método MECAP) que facilita la implementación del método MECAP y contribuye con la trazabilidad entre los requerimientos, el análisis y diseño y las pruebas del software.

La metodología con la que se desarrolló SAM fue UP (Unified Process). El tiempo de ejecución del proyecto de desarrollo de SAM fue de 5 meses, a través 5 iteraciones.

Este trabajo consta de 4 secciones, incluyendo la presente introducción. En la sección 2 se presenta el método MECAP. En la sección 3, se muestra la herramienta SAM. Por último, en la sección 4 se presentan las conclusiones y el trabajo futuro.

3. Método para Especificar CAsos de Prueba (MECAP)

El método propuesto consiste en construir CP a partir de CU ya que se parte del supuesto que se debe probar el comportamiento del software en base a las solicitudes o requerimientos.

Un CP es una especificación, usualmente formal, de un conjunto de entradas de prueba, condiciones de ejecución y resultados esperados, identificados con el propósito

de hacer una evaluación de aspectos particulares de un elemento objeto de prueba (Kruchten, 2000):

- Los CP reflejan trazabilidad con los CU (Funcionalidad), ya que éstos muestran una secuencia ordenada de eventos, al describir flujos básicos, flujos alternos, precondiciones y postcondiciones.
- Las especificaciones suplementarias de requerimientos ya que existen otras características de calidad a evaluar, además de la funcionalidad, como Usabilidad, Confiabilidad, Eficiencia, Mantenibilidad y Portabilidad.
- Las especificaciones de diseño del Sistema, ya que se debe verificar que el software fue implementado según el diseño y que los elementos arquitectónicos garantizan la calidad del software.

Esto garantiza que los procedimientos de pruebas sean compatibles con las necesidades y requisitos de los usuarios/clientes. En la práctica se tiende a asumir que un CU en sí mismo es un CP y que el equipo del proyecto trabaje correctamente sobre los CU sin planificar los CP. Cuando se sienta a probar los CU, intuitivamente asume datos y procedimientos de pruebas sin que éstas queden documentadas. Esto es un error ya que el CP extiende o amplía la información contenida en un CU, por ejemplo, en los CU no indicamos valores ni condiciones para la pruebas.

Los CP son esenciales para todas las actividades de pruebas (Kruchten, 2000): (a) Son la base para diseñar y ejecutar los procedimientos de pruebas. (b) La profundidad de las pruebas es proporcional al número de casos de pruebas. (c) El diseño y desarrollo, y los recursos necesarios son gobernados por los casos de pruebas requeridos. (d) Si los CP no son correctos, la Calidad del Sistema se pone en duda y las pruebas dejan de ser confiables.

El moverse de un CU a un CP es un proceso razonablemente amplio y no trivial. Leffingwell & Widrig (2006) señalan cuatro (4) pasos para lograrlo: (a) Identificar los escenarios del CU. (b) Por cada escenario, identificar uno o más CP. (c) Por cada CP, identificar las condiciones que originaran su ejecución. (d) Completar el CP incorporando valores de datos.

Estos pasos indican qué debe hacerse pero no se enseña el cómo de una forma detallada. Tomando como base los pasos que establecen Leffingwell & Widrig (2006) se propone un método para especificar CP a partir de CU, constituido por cuatro fases principales: (1) Identificar Escenarios, (2) Identificar CP, (3) Especificar los CP, (4) Ejecución y Aprobación del CP.

El aporte de este método es que incorpora elementos de trazabilidad de las pruebas con respecto a todo el proceso de desarrollo y a su vez mejora la estrategia de las pruebas al disciplinar este proceso. Así mismo, ayuda a documentar las ideas previas a las pruebas, los CP y cómo estos fueron generados. Este método mantiene los 4 roles que se proponen dentro de la Disciplina de Pruebas: Gerente de Pruebas, Diseñador de las Pruebas, Analista de Pruebas y Probador.

Fase 1: Identificar Escenarios.

El analista de pruebas activa esta fase una vez que se hayan verificado las narrativas de los CU, por lo menos se tengan claras definidas las funcionalidades del sistema, para:

1. Identificar los escenarios tomando como base las narrativas de los CU y considerando cada uno de los escenarios específicos que ocurren para cada CU. El flujo normal, cada flujo alterno o la combinación de ellos es un escenario, que puede ser ejecutado y probado. Esto deriva que siempre el primer escenario sea el que evoca todo el flujo normal de ese CU en particular y que la relación entre CU y escenarios sea de uno a muchos.
2. Presentar gráficamente la secuencia de eventos que se plantea en cada CU: esto permite, como lo muestra la Figura 1 abstraer los eventos que ocurren en un CU: el flujo normal o básico así como, los flujos alternos lo cual sirve de apoyo para visualizar fácilmente las posibles combinaciones que representarían un escenario ya que establece en qué punto del flujo básico ocurre y además qué sucede después que se activa ese flujo alterno: finaliza el CU o retorna al flujo básico.
3. Chequear que se hayan representado gráficamente todos los Flujos alternos con su acción de finalización o retorno.
4. Establecer a través de una tabla (como lo refleja la Figura 2), todos los escenarios asociados a un CU de la Figura 1.
5. Identificar cada escenario del CU indicando el flujo normal y/o el o los flujo(s) alterno(s) que lo componen. La Figura 3 constituye el primero de los 3 artefactos que se generan en MECAP: Tabla de Escenarios por CU. En ésta se puede observar que se incorpora el ID del Escenario con el propósito de establecer un elemento de trazabilidad de las pruebas lo que facilita, a su vez, las actividades de verificación y aprobación de las pruebas. Como se puede observar en la Figura 3, el ID puede estar conformado por el Nro. CU y Nro. Escenario, esto nos indicaría por ejemplo, que si se tiene 02-02, se traduce como el escenario 02 del CU 02. Esta nomenclatura debe ser establecida por la organización. Además, es importante que se agregue un nombre corto durante la identificación de los flujos alternos que representan o conforman un escenario. La información de los escenarios está asociada a la narrativa de un CU (Validar Usuario del Sistema TAI – Tarjeta Académica Inteligente), el cual sirve de ejemplo para explicar el método propuesto.
6. Verificar que se hayan identificado y descrito todos los escenarios posibles para cada CU. En conclusión, cada escenario representa el número de posibilidades o formas a través de las cuales se puede recorrer el CU. Esto evita que solo se prueben algunas de las combinaciones posibles.

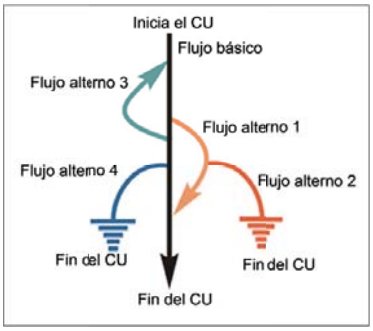

Figura 1. Visualización de los Flujos en un CU (Leffingwell & Widrig, 2006)

Nro. Escenario	Flujo Originario	Flujo alterno	Próximo alterno	Próximo alterno
Escenario 1	Flujo Básico			
Escenario 2	Flujo Básico	Flujo alterno 1		
Escenario 3	Flujo Básico	Flujo alterno 1	Flujo alterno 2	
Escenario 4	Flujo Básico	Flujo alterno 3		
Escenario 5	Flujo Básico	Flujo alterno 3	Flujo alterno 1	
Escenario 6	Flujo Básico	Flujo alterno 3	Flujo alterno 1	Flujo alterno 2
Escenario 7	Flujo Básico	Flujo alterno 4		
Escenario 8	Flujo Básico	Flujo alterno 3	Flujo alterno 4	

Figura 2. Escenarios por CU (Leffingwell & Widrig, 2006)

ID Escenario	Flujo Originario	Flujo alterno	Próximo alterno	Próximo alterno
02-01	Flujo Básico			
02-02	Flujo Básico	Flujo alterno 1: Error Login		
02-03	Flujo Básico	Flujo alterno 1: Error Password	Flujo alterno 3: Presionar Cancelar	
02-04	Flujo Básico	Flujo alterno 2: Error Password		
02-05	Flujo Básico	Flujo alterno 2: Error Password	Flujo alterno 3: Presionar Cancelar	
02-06	Flujo Básico	Flujo alterno 1: Error Login	Flujo alterno 2: Error Password	
02-07	Flujo Básico	Flujo alterno 1: Error Login	Flujo alterno 2: Error Password	Flujo alterno 3: Presionar Cancelar
02-08	Flujo Básico	Flujo alterno 3: Presionar Cancelar		

Figura 3. Tabla de Escenarios para CU002

Fase 2: Identificar los CP

El proceso de pruebas varía de empresa a empresa y de proyecto a proyecto, pero en cada situación un CP documenta un número de elementos comunes. De un escenario de un CU pueden derivarse varios CP. Después que se han identificado todos los escenarios de un CU, se analizan cada uno de ellos en la búsqueda de los posibles CP. Esto se traduce en las siguientes actividades:

1. Se organizan las ideas de pruebas en base los elementos que se desean probar: Funcionalidad (CU), atributos de Calidad, Validaciones de entradas y salidas, Base de datos, interfaces, etc. Esto va a depender del tipo de aplicación, de las restricciones tecnológicas y que maneja el proyecto y de propósito y motivación de las pruebas y experiencia del equipo de pruebas (sobre todo, del probador).
2. Se comienza a llenar la Tabla que se muestra a través de la Figura 4, la cual representa el segundo artefacto del método propuesto y cuyos datos están asociados a considerar los CP para el Escenario 02-02 (Error en Login). Con la información proveniente de las ideas de pruebas se puede decir que para este caso se podría tener un CP para validar "error en login" cuando se introducen caracteres inválidos, un CP cuando el login está en minúsculas, un CP cuando el login es mayor a 10 caracteres, un CP cuando el login está en blanco. Se completa la información asociada a cada CP identificado: ID Caso de Prueba, Nombre del CP, Resultados esperados (valores, mensajes de error, etc.), Nivel de Prueba y Tipo de Prueba. Con respecto al ID del CP se propone incluir en la nomenclatura estándar que determine la organización una estructura que refleje CU-Escenario-CP para así saber por ejemplo que 02-02-01 significa el CP 01 del Escenario 02 del CU 02.
3. Se verifica que se hayan identificado todos los CP para cada escenario. Después de esto, se procede a la siguiente fase.

ID Caso de Prueba	Nombre del Caso de Prueba	Resultados esperados	Nivel de Prueba	Tipo de Prueba
02-02-01	Login con caracteres inválidos	*Mensaje Error 20: Login Inválido*	Sistema/ Aceptación	Función/ Control de acceso
02-02-02	Login con minúsculas	*Mensaje Error 20: Login inválido*	Sistema/ Aceptación	Función/ Control de acceso
02-02-03	Login con longitud mayor a 10 caracteres	*Mensaje Error 20: Login inválido*	Sistema/ Aceptación	Función/ Control de acceso
02-02-04	Login en Blanco	*Mensaje Error 20: Login inválido*	Sistema/ Aceptación	Función/ Control de acceso

Figura 4. Tabla de Casos de Prueba para el Escenario 02-02

Fase 3: Especificaciones de los CP

Uno de los aportes más importantes de esta investigación es el tercer artefacto que se utiliza para especificar con detalle el CP y que se muestra a través de la Figura 5: Tabla de Especificaciones de CP (ECP).
Para cada CP, se deben llevar a cabo las siguientes actividades:

1. Identificar el nombre del Sistema, ID CU, ID de Requerimiento, ID Escenario, ID del CP y versión del CP. Esto permite hacer una traza bidireccional entre estos elementos: por ejemplo, se puede conocer si se especificaron todos los CP para todos los CU y se probaron todos los CU (cobertura de las pruebas).
2. Identificar el nivel y tipo de prueba asociada al CU y cuya información proviene de la Tabla CP por Escenario.
3. Indicar el ambiente de las pruebas. Se puede señalar el nombre de la empresa, si es el ambiente de desarrollo o producción, o si tengo varios ambientes en la empresa se indica en cuál de ellos se ejecutará ese CP en particular.
4. Identificar el nombre del autor del CP y del Probador. Se recomienda que sean personas diferentes las que ocupen estos roles a fin de darle objetividad y confiabilidad al proceso de las pruebas.
5. Señalar la fecha de creación de la versión de ese CP y la fecha en la que fue ejecutado.
6. Identificar las condiciones que deben estar presentes para ejecutar el CP. ¿Cuáles son las condiciones que originan o causan que un usuario ejecute un evento específico o una secuencia de eventos? En la Figura 5, se tiene el CP 02-02-02 en cual se propone el CP para Login con minúsculas. En esta figura se observa que se deben haber implementado todas las funcionalidades asociadas con Validar Usuario, así mismo, los Datos que se utilizaran en este CP hayan sido validados y aprobados por la instancia correspondiente, etc.
7. Describir el procedimiento del CP. Este procedimiento está compuesto de los pasos que se deben hacer para probar el Escenario del CU a través de lo que plantea el CP, de las condiciones particulares que pudieran aplicar para un determinado paso, los valores utilizados, los resultados esperados y los resultados obtenidos. Cabe resaltar, que estos últimos se incluyen en la Tabla ECP una vez que el CP es ejecutado.

ID/Nombre Sistema/Proyecto: SIS-PROY		Nivel de Prueba: Sistema/Aceptación	
ID Caso de Uso: CU-02 Validar Usuario		Tipo(s) de Frueba(s): Función/Control de acceso	
ID Requerimiento: (solo para Caso de Uso No Funcional)		Ambiente de Prueba: AMBIENTE1	
ID/Nombre Escenario: 02-02 Error en Login		Autor del Caso de Prueba: LISI	
ID/Nombre Caso de Prueba: 02-02-02 Login con minúsculas		Nombre del Probador: Probador 1	
Versión del Caso de Prueba: v.1.		Fecha de Creación: 10-01-07	Fecha de Ejecución: 15-03-07
Condición(es) para que se ejecute el Caso de Prueba:			
El usuario desea acceder al sistema. Se hayan implementado todas las funciones que se relacionan con validar usuario. Datos a utilizar para las pruebas hayan sido validados y aprobados. Deben existir usuarios registrados como válidos.			
Para la Ejecución del ID Caso de Prueba:			

Paso	*Condición*	*Valor(es)*	*Resultado Esperado*	*Resultado Obtenido*
Introduce login presiona Ok	intento de acceso < 3	aDM22	Mensaje Error 20 :Login inválido	√
Introduce login presiona Ok	intento de acceso < 3	administrador	Mensaje Error 20 :Login inválido	√
Introduce login presiona Ok	intento de acceso < 3	AdminisTRADOR	Mensaje Error 20 :Login inválido	√

Criterios de Aprobación del Caso de Prueba:	Si se cumplen en un 100% los resultados esperados
Decisión de Aprobación del Caso de Prueba:	Aprobó: X Fallo: ____ (marque con una X el resultado obtenido)
Fecha de Aprobación del Caso de Prueba:	15-03-07

Figura 5. Tabla de Especificación del CP 02-02-02

Sin la incorporación de datos no es posible ejecutar las pruebas y determinar los resultados. Se deben observar las especificaciones suplementarias para encontrar medidas de desempeño (mínimo y máximo), rangos válidos de entrada, protocolos de interfaces, entre otros.

8. Se indica cuál es el Criterio de Aprobación del CP. Como se observa en el ejemplo de la Figura 5. El Criterio es que se deben cumplir en un 100% todos los resultados esperados. También puede indicarse a nivel de pasos cuando el CP involucra pasos diferentes y alguno(s) de ellos son más importante que otros: Si se cumplen en un 100% los resultados esperados del Paso 2, 4 y 5.
9. El analista y el diseñador de las pruebas verifican que todos los CP se hayan especificados correctamente.

Fase 4: Ejecución y Aprobación del CP

Esta fase consiste en poner en práctica el procedimiento de los CP que se encuentran en la Tabla ECP. A continuación se indican las actividades de esta fase:

1. Debe comprobarse que está dado el ambiente y las condiciones para ejecutar los CP. Todos los involucrados en las pruebas deben cooperar en esto.
2. El Gerente y el Diseñador de las pruebas deben autorizar que se active el ciclo de prueba.
3. El Probador ejecuta todos los CP e incorpora los datos de los resultados obtenidos en cada Tabla ECP.
4. El Gerente de Pruebas toma la decisión de si aprobó o falló el CP en base al criterio de aprobación y además, indica la fecha de la aprobación y en algunos casos, coloca su firma.
5. El equipo de pruebas verifica si se cumplió el criterio de salida del ciclo de prueba para decidir si se aprueba el ciclo de prueba o hay que hacer pruebas de regresión y pruebas adicionales a ciertos CP en un próximo ciclo de prueba hasta que se obtengan los criterios de aceptación.
6. El equipo de pruebas guarda todos estos entregables y publica los resultados en

base a los resultados de los ciclos de prueba, cambios, etc. que se dieron hasta completar el proceso de las pruebas.

Puntos de Control

Para garantizar la correcta aplicación del método y cumplir con la estrategia, a continuación se enumeran los puntos de control que se realizan en cada fase.

Fase 1:

- ¿Existe una matriz de Escenarios por cada CU del sistema?
- Revise que todos los escenarios del CU correspondiente se hayan indicado en la matriz de escenarios por CU.
- Revise que los ID del CU y CP estén completos y se correspondan con la nomenclatura propuesta.

Fase 2:

- ¿Existe una matriz de CPs por escenario?
- Para cada matriz de CP por escenario, revisar que están completos y correctos los ID, nombres, resultados esperados, nivel y tipo de prueba.

Fase 3:

- ¿Existe una tabla de Especificación de CP para cada CP identificado en la fase anterior?
- Revisar la trazabilidad entre los ID de los CP, CU, requerimiento y escenario.
- Chequear que todos los puntos indicados en la tabla de especificación de CP estén completos y correctos a excepción de los campos a llenar para la próxima fase (ejecución).
- ¿Se validaron los criterios de aprobación para cada especificación de CP?.

Fase 4:

- ¿Se documentaron los resultados de las ejecuciones de los CP a través del llenado de los campos: resultados obtenidos, aprobó/falló, fecha de aprobación, fecha de ejecución.
- ¿Se indicó en el documento Plan de Pruebas, el criterio de cumplimiento del ciclo de pruebas?
- ¿Los resultados de las pruebas y los cambios durante el proceso de las pruebas fueron entregados y publicados?

En la siguiente sección se presenta la herramienta SAM que automatiza el método propuesto y presentado en esta sección.

4. Herramienta SAM – Sistema Automatizado del Método MECAP

A fin de integrar las disciplinas de Requerimientos con las de Pruebas, se desarrolló una herramienta denominada Workbench UML-UCM que permite convertir un archivo .uml (generado a través de StarUML) en un archivo transformado .ucm (en la herramienta Use Case Maker) para especificar las narrativas de los CU. Una vez finalizadas las especificaciones, se procede a exportar/generar un archivo (.xml) el cual será cargado en la herramienta SAM para luego proceder con la ejecución del método MECAP. Los detalles de Workbench UML-UCM escapan del alcance de este artículo.

SAM fue desarrollado en Software Libre y para su implantación se requiere:

Hardware: El Sistema no requiere de hardware con especificaciones determinadas. Sin embargo se recomienda no utilizar un computador con menos de 1GB de RAM, menos de 1.6GHz de velocidad de procesador, sin puerto de internet o sin conectividad WIFI.

Software: Para la instalación del servidor web: WAMP (Windows Apache 2.2.11, php 5.3 y MySQL 5.1.36) o LAMP ((Similar a WAMP pero con Linux como Sistema Operativo. Software de Soporte: Navegador web: Firefox (desde la versión 3, no menor) o Internet Explorer (IE7, no menor).

A continuación, se muestran algunas de las interfaces de SAM al momento de generar los escenarios y especificar los CP.

En la Figura 6, se muestra la pantalla de inicio de sesión. Después, se puede seleccionar un proyecto.

Figura 6. Pantalla de Inicio de Sesión.

Después de haber seleccionado el proyecto, se debe generar los escenarios a partir de los CU. En la Figura 7, se muestra cómo el Sistema permite importar CU (la narrativa de los CU) en formato XML desde archivos guardados en la computadora. Para utilizar esta opción se presiona el botón de *Cargar Casos de Uso* del menú desplegable de *Casos de Uso*.

Cuando ya se han importado CU al sistema, éste permite consultar los datos o narrativa de cada CU. Para ello, se debe seleccionar el ciclo de prueba asociado a los CU. Para esto se presiona el botón de *Ver casos de uso* del menú desplegable de Casos de Uso o la opción de *Ver casos de uso* del menú interno ubicado del lado izquierdo, como se muestra en la Figura 8.

Cuando ya se han importado los CU, se pueden generar los escenarios para cada CU como se puede apreciar en a Figura 9. Primero se presiona el botón de *Generar Escenarios*, ubicado en la parte inferior de la página de consultar los datos de un CU. Una vez que se selecciona la opción de *Generar Escenarios* se presenta una lista de los escenarios para dicho CU (se pueden mantener todos los escenarios o descartar algunos, dependiendo de los acuerdos a los que llegue el equipo de pruebas) y también se puede ver el grafo asociado al CU, similar a lo que se presenta en la Figura 1.

Para generar los CP, se deben seleccionar los Escenarios como se muestra en la Figura 10. Finalmente en la Figura 11, se muestra la especificación del CP.

Figura 7. Pantalla de Importar Casos de Uso.

Figura 8. Pantalla de Ver Casos de Uso.

Figura 9. Pantalla de Generar Escenarios.

Figura 10. Pantalla de Generar Casos de Prueba.

Figura 11. Pantalla de Especificación de Casos de Prueba.

5. Conclusiones y trabajo futuro

Este artículo presenta la herramienta SAM en una primera versión, la cual permite apoyar la ejecución del método MECAP de forma automatizada, facilitando una mejor gestión del proceso de las pruebas así como la trazabilidad entre diferentes disciplinas de Ingeniería de Software. Así mismo SAM permite apoyar la documentación de las pruebas y mejorar su efectividad.

En estos momentos se están probando el funcionamiento de SAM en el desarrollo de las pruebas de sistemas relacionados con el sector bancario para evaluar la calidad y completitud de los casos de prueba que son elaborados con su soporte. Se espera incorporar a corto plazo algunas funcionalidades que mejoren el Flujo de Trabajo de la Disciplinas de Pruebas como es el caso de los reportes de los ciclos de pruebas así como presentar el Workbench detallado.

6. Referencias bibliográficas

Leffingwell D. & Widrig D. (2006). Managning Software Requirements, a Use Case Approach. Second Edition. Addison-Wesley, Pearson Education.

Kruchten, P. (2000). The Rational Unified Process as Introduction. 2nd Edition. Addison – Wesley.

Lewis W. 2000. Software testing and continuous quality improvement 000 by CRC Press LLC Auerbach is an imprint of CRC Press LLC.

Méndez E., Pérez, M. & Mendoza, L. E. (2008). Improving Software Test Strategy with a Method to Specify Test Cases (MSTC). 10th International Conference on Enterprise Information Systems (ICEIS 2008). Barcelona, España. Junio 2008.

Méndez, E. Pérez, M., & Mendoza, L. E. (2007). Aplicación de un Método para Especificar Casos de Prueba de Software en la Administración Pública. PRIS 2007: Taller sobre Pruebas en Ingeniería del Software. Libro: "Actas de Talleres de Ingeniería del Software y Bases de Datos (JISBD 2007)", Zaragoza, España.

Perry W. 2006. Effective Methods for Software Testing. Wiley. Third Edition.

Pinkster I., Burgt B., Janssen D. and Veenendaal E. 2006. Successful Test Management. Springer and Logicacmg.

Pfleeger, S. (1998). Software Engineering.

Pressman, R. (2002). Ingeniería del Software: Un enfoque Práctico. McGraw Hill.

Sommerville, I. (2000). Software Engineering. Pearson Education.

SWEBOK. 2004. Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE Computer Society.

UML Testing Profile Version 1.0 formal/05-07-07. This is a testing profile for UML 2.0.

Utting M. and Legeard B. 2007. Practical Model-based Testing. Morgan Kaufmann and Elsevier Editorial.