Ensayos

Métodos formales aplicados en la industria del software.

Resumen

Uno de las ideas más difundidas sobre el uso de métodos formales es su nula aplicación en la industria sotware. Si bien es cierto que su adopción ha sido lenta, existen casos de importantes empresas que han tenido éxito al aplicarlos en proyectos reales. En el presente artículo se mencionan algunos de estos ejemplos donde se han aplicando distintos métodos formales, tanto en la industria de software y hardware, incluyendo la situación en nuestro país y las perspectivas a futuro.

Abstract

One of the most widely spread ideas about the use of formal methods is that its application is nullified in the software industry. While it is true that its adoption has been slow, there are cases of important businesses that have successfully applied these methods in real projects. This article will examine some of these examples where different formal methods have been applied in both the software and hardware industries, including the situation in our country and its future perspectives.

Résumé

Une des idées les plus généralisées sur l'utilisation de méthodes formelles est sa non-application dans l'industrie du software. S'il est certain que son adoption a été lente, il existe des cas d'entreprises importantes qui ont remporté le succès en appliquant ces méthodes dans des projets réels. Dans cet article on mentionnera quelques uns des exemples pour lesquels ont été appliquées des méthodes formelles, dans l'industrie du software et du hardware, en y incluant la situation dans notre pays et les perspectives dans le futur.

* Carlos A. Fernández y Fernández, Miriam Ambrosio, Gabriel Andrade, Galileo Cruz, Martín José, Román Ortiz, Hernán Sánchez.

Palabras clave:
Métodos formales.

1. Introducción

De acuerdo con Hall (1990), la escasa adopción de métodos formales en la industria software es solamente un mito. La existencia de este mito se debe principalmente a que hasta hace algunos años existían pocos casos documentados en la industria. Con el objetivo de que las empresas consideren la adopción de tales métodos, es importante que conozcan la existencia de casos de éxito que muestren las ventajas de su aplicación. Sin embargo, no basta con saber que la aplicación de métodos formales se da de manera exitosa en proyectos en la industria del software. Es necesario además que éstos puedan ser adaptados al contexto de las empresas mexicanas si se desea que consideren su adopción en sus propios proyectos. Por tal razón en el presente artículo se muestra una recopilación de casos de éxito de la aplicación de métodos formales en proyectos reales de la industria de software y hardware, incluyendo una breve introducción de la situación en nuestro país.

* Universidad Tecnológica de la Mixteca.

2. ¿Qué son los métodos formales?

Un método formal es una técnica basada en matemáticas, usada para describir sistemas de hardware o software, Wing, Jeannette M. (1990). Los métodos formales permiten representar la especificación del software, verificación y diseño de componentes mediante notaciones matemáticas. El uso de métodos formales permite plantear de manera clara la especificación de un sistema, generando modelos que definen el comportamiento en términos del "qué debe hacer" y no del "cómo lo hace". Gracias al correcto proceso de especificación, se pueden verificar propiedades derivadas de cada módulo mediante técnicas de razonamiento asociadas a los modelos formales, como probadores de teoremas y verificadores de modelos Hall (1996). Para los procesos de especificación se reconoce las siguientes clasificaciones:

1. Lenguajes basados en modelos y estados. Permiten especificar el sistema mediante un concepto formal de estados y operaciones sobre estados. Los datos y relaciones/ funciones se describen en detalle y sus propiedades se expresan en lógica de primer orden. La semántica de los lenguajes está basada en la teoría de conjuntos. Ejemplos: VDM, Z, B, OCL.
2. Lenguajes basados en especificaciones algebraicas. Proponen una descripción de estructuras de datos estableciendo tipos y operaciones sobre esos tipos. Para cada tipo se define un conjunto de valores y operaciones sobre dichos valores. Las operaciones de un tipo se definen a través de un conjunto de axiomas o ecuaciones que especifican las restricciones que dében satisfacer las operaciones. Algunos ejemplos: Larch, OBJ, TADs.
3. Lenguajes de especificación de comportamiento:
   - Métodos basados en álgebra de procesos: modelan la interacción entre procesos concurrentes. Esto ha potenciado su difusión en la especificación de sistemas de comunicación (protocolos y servicios de telecomunicaciones) y de sistemas distribuidos y concurrentes. Algunos ejemplos son: CCS, CSP, Pi Calculus y LOTOS.
   - Métodos basados en Redes de Petri: una red de Petri es un formalismo basado en autómatas, es decir, un modelo formal basado en flujos de información. Permiten expresar eventos concurrentes. Los formalismos basados en redes de Petri establecen la noción de estado de un sistema mediante lugares que pueden contener marcas. Un conjunto de transiciones (con pre y post condiciones) describe la evolución del sistema entendida como la producción y consumo de marcas en varios puntos de la red.
   - Métodos basados en lógica temporal: se usan para especificar sistemas concurrentes y reactivos. Los sistemas reactivos son aquellos que mantienen una continua interacción con su entorno respondiendo a los estímulos externos y produciendo salidas en respuestas a los mismos, por lo tanto el orden de los eventos en el sistema no es predecible y su ejecución no tiene por qué terminar.

En cuanto al proceso de verificación se tienen básicamente dos enfoques:

1. Verificación de modelos: trabajan mediante una búsqueda exhaustiva en los estados posibles de un modelo para encontrar errores en la especificación.
2. Prueba de teoremas: donde a partir de un conjunto de axiomas se trata de probar si la especificación, extendida con algunos teoremas, es válida.

3. Métodos formales aplicados en la industria

3.1. Aplicación de métodos formales en proyectos de software: CICS IBM

El proyecto CICS (Customer Information Control System) 1 fue diseñado en los laboratorios Hursley Park de IBM, para ofrecer acceso a datos, comunicaciones, integridad y servicios de seguridad. Es uno de los sistemas de software más exitosos en el mundo, cuenta con más de 30,000 licencias y la mayoría de las principales compañías del mundo lo utilizan, Woodcock, Jim

and Davies, Jim (1996). Entre las aplicaciones más conocidas del proyecto CICS se incluyen sistemas de compensación bancaria, control de almacenes, reservaciones aéreas y sistemas ATM, y cuenta con miles de licencias corporativas en el mundo Freitas et al. (2009).

Especificación del proyecto CICS

En las primeras implementaciones de CICS, el API incluye bloques de control y un conjunto de macros del lenguaje, alentando a los programadores a conocer los detalles internos sobre los bloques de control utilizados en la implementación del sistema.

El API contaba con más de 90 comandos con 300 opciones aproximadamente y fue originalmente dividido en diferentes grupos de comandos. Tiempo después se decidió usar estos grupos para estructurar la especificación formal dentro de 15 módulos CICS. Los módulos son:

1. Iniciación automática de transacciones.
2. Soporte a cartografía básica.
3. Control de descargas.
4. Manipulación de condiciones excepcionales.
5. Control de archivos.
6. Control de intervalos
7. Control periódico.
8. Programa de control.
9. Control de almacenamiento
10. Control de tareas.
11. Control de almacenamiento temporal.
12. Control de terminales.
13. Control de trazos.
14. Transacciones y servicios principales.
15. Transitoria de control de datos.

De acuerdo a estos módulos se generaron también quince tablas de control asociadas a la funcionalidad de cada módulo. Una descripción completa de estas tablas se puede encontrar también en el sitio oficial de CICS. Posteriormente IBM decidió incorporar la especificación formal usando el lenguaje formal Z en una nueva versión de IBM CICS, y de igual forma hacer la especificación formal del API de CICS basado en módulos. Esta estrategia surgió por la necesidad de:

1. Proporcionar una interfaz de comandos básicos para todas las versiones del CICS.
2. Descubrir cualquier comportamiento accidental que no fuera parte de la intención original del diseñador.
3. Hacer explícito qué comportamientos fueron efectivamente garantizados.

Una evaluación reportó que el resultado de esta estrategia se tradujo en una mejora en la calidad y la fiabilidad del código entregado. En junio de 1989 se desarrolló el primer producto utilizando la notación Z llamado CICS / ESA versión 3. Con el desarrollo de los primeros proyectos se pudo observar que la especificación formal con el mecanismo de la notación Z brindó una provechosa experiencia la cual trajo a la luz no sólo errores cometidos en el pasado, sino también a las nuevas orientaciones del lenguaje Z en sí.

3.2. La combinación de UML y métodos formales en la seguridad de un aeropuerto

La seguridad en la aviación civil se rige por normas internacionales y prácticas recomendadas. Un elemento clave en la seguridad de la aviación es la seguridad del aeropuerto, que impide que puedan llevarse a bordo de un avión armas y otros objetos peligrosos. La calidad de la seguridad de los aeropuertos depende de:

1. La calidad de estas normas, y
2. La conformidad de un aeropuerto dado a estas normas

El proyecto EDEMOI 2 fue patrocinado desde 2003-2006 por ACI Sécurité Informatique. El objetivo de EDEMOI es la aplicación de técnicas de modelado de la comunidad de Ciencias de la Computación para abordar estos dos problemas. El enfoque EDEMOI se basa en dos tipos de modelos como se muestra en la Figura 1:

1. Modelos gráficos, tales como diagramas de clases UML.
2. Modelos formales, es decir, B y lenguaje Focal.

En el enfoque EDEMOI se presentaba como principal problema la validación del modelo formal por parte de las autoridades de certificación en aviación civil, las cuales no tuvieron ningún problema al validar

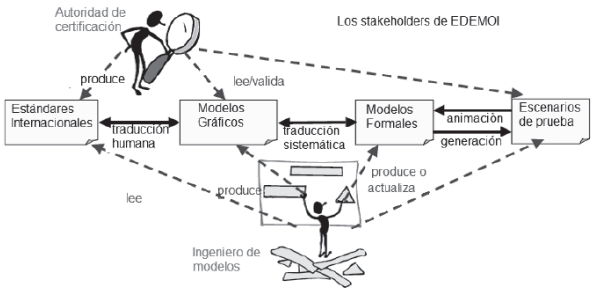

FIG. 1: STAKEHOLDERS DE EDEMOI

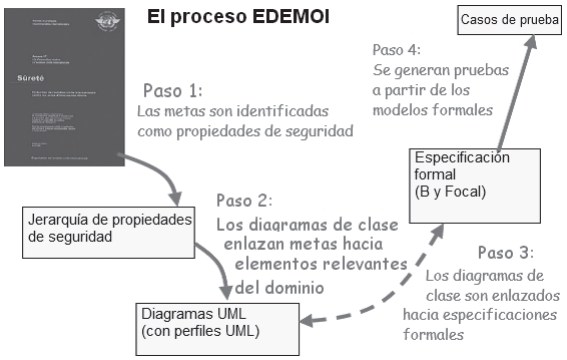

FIG. 2: EL PROCESO EDEMOI

los diagramas de clases y escenarios que se generaron durante el proyecto.

EDEMOI se centra en el modelado de las normas de todos los aeropuertos internacionales de la aviación comercial. Se propuso un proceso que se muestra en la Figura 2, para la identificación de las propiedades de seguridad, la producción de gráficas (UML) y los modelos formales (B y Focal), y se inició una actividad de generación de pruebas desde el modelo formal.

Se dedicaron esfuerzos especiales en los dos primeros pasos, con el fin de mejorar la validación del modelo gráfico. Lo cual condujo a la definición de un enfoque de ingeniería de requerimientos dedicado a la seguridad de la aviación.

Resultados

Estas actividades se dirigieron a los dos objetivos mencionados anteriormente: el modelado de las normas para aeropuertos internacionales llevó al descubrimiento de ambigüedades del lenguaje natural e hipótesis ocultas. El análisis más detallado de los modelos formales ayudó a evaluar la calidad de la norma mientras que la generación de pruebas abordó el problema de conformidad. Los principales resultados científicos de EDEMOI fueron:

1. El enfoque de modelado que conduce a la identificación de objetivos y la producción de los modelos gráficos;
2. los modelos formales de las normas para aeropuertos internacionales; y
3. la generación de pruebas de conformidad a las normas.

Los productos derivados de este proyecto incluyen el desarrollo de herramientas que permiten volver de especificaciones B a los diagramas UML y la experimentación del lenguaje Focal como un lenguaje de modelado formal.

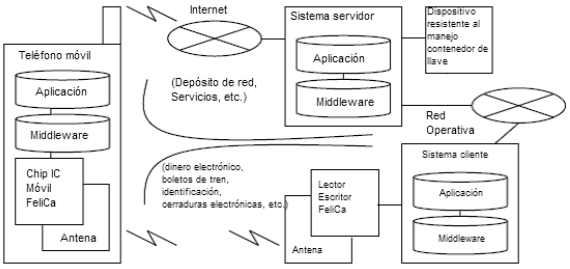

FIG. 3: DIAGRAMA DE MOBILE FELICA SYSTEM KURITA ET AL. (2008)

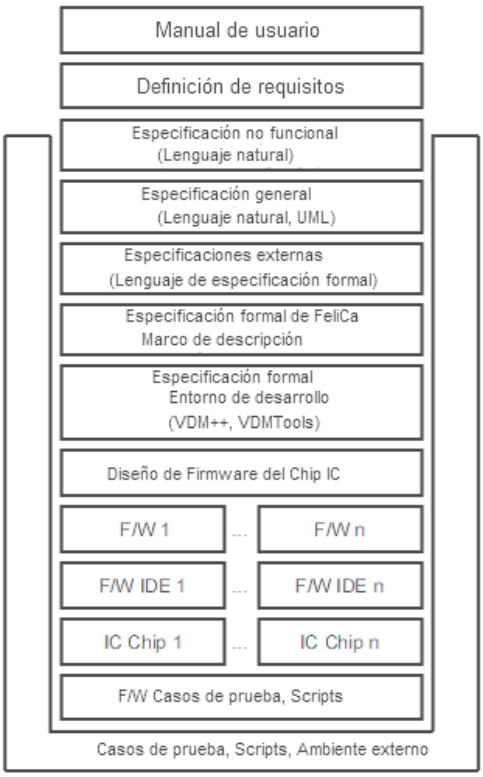

FIG. 4: DIAGRAMA OBTENIDO DE LA ESPECIFICACIÓN DEL SISTEMA UTILIZANDO VDMTOOLS

El proyecto EDEMOI ha demostrado que las técnicas dedicadas al modelado, análisis y pruebas de la seguridad, podrían utilizarse provechosamente en otras áreas. Este proyecto es un caso de éxito de la conjunción de UML con métodos formales para modelar estándares que regulan la seguridad de un aeropuerto. Utiliza la producción de un modelo UML, para soportar la actividad de validación y un modelo formal para su verificación. Para este proyecto, se utilizó la herramienta Rational Rose RoZ Martínez, Christian A. (2007).

3.3. Chip de firmware para la incrustación en teléfonos móviles Mobile FeliCa

FeliCa es una tecnología de tarjetas sin contacto IC ampliamente utilizado en Japón, desarrollado y promovido por Sony Corporation3 . FeliCa es utilizado por IC Mobile, que está integrado en un teléfono móvil conocido como "Osaifu Keitai" (medio de monedero electrónico) por NTT DoCoMo, Inc., y hoy en día estos teléfonos móviles pueden utilizarse como monedero electrónico, tickets para el tren, identificación, llaves, etc. El sistema móvil FeliCa que se muestra en la Figura 3 tiene un sistema de archivos de seguridad, un protocolo de comunicaciones y un firewall que controla los múltiples servicios antes mencionados.

Uno de los aspectos más importantes es asegurar la máxima calidad del software con el fin de evitar graves problemas sociales relacionados con la infraestructura y el servicio que reciben los clientes no se vería afectado.

Lenguaje Utilizado

Debido a la exigencia del proyecto se determinó utilizar un método formal para el proyecto, bajo los siguientes objetivos:

1. Describir rigurosamente las especificaciones y definición de funciones.
2. Desarrollar y aplicar un sistema y los procesos de desarrollo de la especificación, la aplicación y pruebas del firmware.
3. Verificar detalladamente las especificaciones formales para el conjunto de procesos de desarrollo de software.
4. Mejorar la comunicación entre los ingenieros y desarrolladores.

Una vez definido los objetivos, se decidió utilizar el lenguaje de especificación formal VDM++ Dürr e van Katwijk (1992) y VDMTools4 , debido a que el apoyo sobre la descripción y la ejecución de los modelos era demasiada.

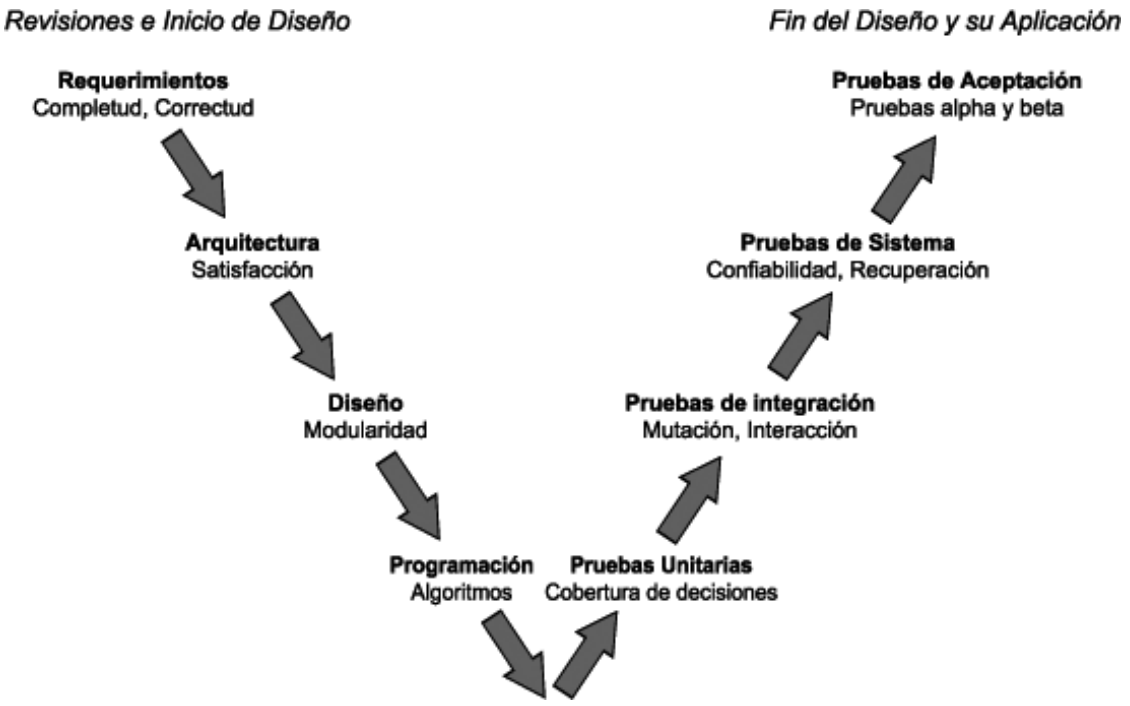

FIG. 5: ETAPAS DEL MODELO V LEÓN-CARRILLO, LUIS VINICIO (2005A)

Resultados Obtenidos

Los resultados obtenidos durante el proceso se muestran en la Figura 4. Se generó el desarrollo y las especificaciones externas usando VDM++.

El proceso de la especificación del proyecto es la siguiente:

1. Discutir los requerimientos con los stakeholders y escribir las especificaciones generales en un lenguaje natural con varios diagramas basándose en notación UML, diagramas de transición de estado y diagramas de secuencia.
2. Modelar el sistema de archivos de FeliCa y diseñar e implementar un firmware en VDM++.
3. Describir comandos y especificaciones de seguridad usando el firmware .
4. Hacer especificaciones de pruebas usando un firmware de pruebas unitarias.

Los resultados relacionados con las especificaciones son los siguientes:

1. 383 páginas de un protocolo manual escritas en un lenguaje natural
2. 677 páginas de un pliego de condiciones externas escritas en un lenguaje de especificación formal

Las especificaciones formales tienen alrededor de 100,000 pasos, que incluyen casos de prueba y comentarios escritos en un lenguaje natural.

La aplicación del método formal fue un éxito para los sucesos y esquemas del proyecto. El método formal contribuyó en la calidad de los procesos y en la implementación de seguridad de la comunicación. Además sirvió para identificar la necesidad de establecer una buena especificación de requisitos para obtener una buena calidad en el software.

3.4. Uso de métodos formales para dispositivos de firma digital de seguros

En 1997, el Gobierno de la República Federal de Alemania estableció la Ley de Información y Servicios de Comunicación. Ésta contiene la Ley de Firma Digital. El objetivo es la definición de requerimientos que permitan la equivalencia de la firma manual y digital vinculada a cuestiones jurídicas, autenticidad e integridad de documentos electrónicos en el futuro.

Las SmartCards son los dispositivos adecuados para la generación de firmas digitales. La organización alemana de estandarización (DIN) desarrolló una especificación de interfaz para SmartCards con aplicación/función de firma. El objetivo de esta norma es garantizar la interoperabilidad. Se basa en la norma ISO/IEC 7816[5] y la define los objetos de los datos dentro de una estructura de archivos, así como una colección de comandos de la interfaz que se utilizará para la generación y validación de firmas digitales.

Seguridad de la SmartCard

En preparación del desarrollo del modelo formal de política de seguridad se realizó un análisis de amenazas el cual no tomó en cuenta al hardware o el sistema operativo de la SmartCard. El examen se limitó a la aplicación de software embebido y de datos del procesador de operaciones de firma de datos (SigCard), se revelaron las siguientes amenazas Langenstein et al. (2000):

1. Extracción de la clave secreta de la firma de la SigCard por la lectura de los datos o el resultado de análisis computacional.
2. Uso de la aplicación de firma digital de la SigCard sin tener el permiso del titular de la tarjeta.
3. Imperceptible modificación de los datos firmados.
4. Presentación de una SigCard forjada a un dispositivo de interfaz pública sin que el dispositivo de interfaz pública pueda darse cuenta de eso.
5. Presentación de la SigCard a un dispositivo de infertaz pública forjado sin que el titular pueda darse cuenta de eso.

Se recurrió a la especificación formal para definir las políticas de seguridad de la SigCard. Esta especificación formaliza la importancia de los principios de seguridad y los relaciona con los objetivos de seguridad descritos en el objetivo general de seguridad. El modelo formal fue desarrollado usando “Verification Support Environment” (VSE) Baur et al. (1992), la cual es una herramienta de apoyo a los métodos formales en el ciclo de vida completo del software.

Resultados y Beneficios

El objetivo general de seguridad y el acompañamiento de un modelo formal de política de seguridad para SigCards fue desarrollado en forma paralela. El proceso de formalización de la política de seguridad empezó con un primer borrador del objetivo de seguridad genérico (GST) describiendo sólo algunas

partes de las características de seguridad de SigCard Langenstein et al. (2000).

Cada extensión del objetivo de seguridad genérico causó un nuevo análisis tratando de incluir las características adicionales en un estilo formal en el modelo. Algunas inconsistencias y contradicciones en el borrador GST fueron descubiertos durante el análisis y proceso de formalización como vulnerabilidades en la política de seguridad de la SmartCard

El modelo formal de la política de seguridad define lo esencial del comportamiento de seguridad de la SigCard y suposiciones necesarias en el medio ambiente en una descripción abstracta. El beneficio para la tarea de evaluación es que el evaluador puede verificar el verdadero producto de TI de manera muy eficaz contra el comportamiento de seguridad de la especificación formal. Normalmente el trabajo de evaluación completo se realiza después del desarrollo del producto, éste también incluye la construcción del modelo. El uso de métodos formales en fases tempranas del desarrollo del producto dió una buena asistencia al diseñador del sistema. Como sabemos, pequeñas modificaciones pueden causar serios errores si no se detectan a tiempo. Estos errores pueden no ser fácilmente reconocidos si se especifican en lenguaje natural.

Documentación_del_Proceso →
proceso Identificador() ;
;
Cuerpo
Cuerpo →
Subprocesos
Secuencia_de_Actividades
Subprocesos → proceso Identificador() ;
;
Cuerpo ; Resto1
Resto1 → Subprocesos | λ
Secuencia_de_Actividades →
inicio Actividades fin
Actividades → Actividad Resto2
Resto2 → Actividades | λ
Actividad →

Llamada_a_subproceso → Identificador() ;
Alternación_de_Actividades →
si (

FIG. 6: FORMATO DEL LENGUAJE DE ESPECIFICACIÓN DE PRUEBAS DE E-QUALLITY LEÓN-CARRILLO, LUIS VINICIO (2005A)

4. Métodos Formales en México

La situación en nuestro país es aún precaria en cuanto a información sobre la adopción de Métodos Formales. Esto se debe principalmente a la poca disponibilidad de casos de éxito documentados que permitan establecer con exactitud el panorama actual. Sin embargo, esto no significa que en México no se apliquen métodos formales en los procesos de desarrollo de software. Ejemplo de esto es el caso de estudio documentado en León-Carrillo, Luis Vinicio (2008) que menciona la aplicación de enfoques formales para realizar pruebas de software para la empresa SIAC Software6 realizadas por una empresa especializada en pruebas de software y firmware 7 enfocado en definir jerarquías de procesos .

El método utilizado para realizar las pruebas se basa en aplicar el modelo V-Model (ver Figura 5) con el objetivo de realizar un diseño de pruebas en etapas tempranas del ciclo de desarrollo, utilizando para dichas pruebas lenguajes formales en su definición y aplicación. El método utilizado es una variante de un Lenguaje de Definición de Procesos (PDL por sus siglas en inglés) desarrollado por la misma compañía León-Carrillo, Luis Vinicio (2005b) (ver Figura 6). El caso de estudio compara los resultados obtenidos de un proyecto que realiza pruebas solamente en la etapa final del desarrollo y uno que realiza pruebas desde etapas tempranas del proceso.

Entre las conclusiones obtenidas el autor enfatiza los beneficios de aplicar el lenguaje de especificación ya que ayuda a simplificar los procesos de prueba, y para ello enfatiza la importancia de mantener un lenguaje simple y pequeño para la definición de las pruebas.

Como se mencionó anteriormente, existen pocos casos documentados en nuestro país, similar al expuesto lo que dificulta establecer con exactitud la situación actual. En el área de pruebas de software, esto se justifica debido a la situación de nuestro país. De acuerdo con León-Carrillo, Luis Vinicio (2006), se debe a lo siguiente:

1. "En México, se ve una industria poco estructurada, con muy pocos servicios de alto valor agregado (muy concentrados en la renta de testers), sin fuerte respaldo teórico.
2. Las pocas empresas que ofrecen esos servicios no son especializadas, por lo que suelen ser poco efectivas y poco productivas; lo que es peor: suelen ofrecer también el

servicio de desarrollo de software, lo que les hace perder credibilidad, pues se colocan como juez y parte."

Adicionalmente, de los casos de estudio internacionales analizados, se puede decir que en el uso más conocido de los métodos formales son en proyectos para grandes aplicaciones, como en el caso del proyecto EDEMOI, llevados a cabo por empresas grandes como el caso de CICS e IBM, o para modelos de hardware o software empotrado como los casos de FeliCa y las SmartCards. Sin embargo, en general las empresas que se dedican a actividades relacionadas con Tecnologías de la Información en nuestro país son las llamadas Pequeñas y Medianas Empresas PYMES (ver Tabla 1), que se dedican mayormente a la elaboración de software a la medida, de software empaquetado e implementación de sistemas. Esto implica que pocas empresas tienen los recursos para capacitar a sus empleados en su utilización, adicionalmente, las PYMES esperan que las herramientas adoptadas cumplan ciertas características, Scott et al. (2001):

1. Los métodos utilizados deben ser efectivas y producir buenos resultados (ciclos de vida más cortos, detección temprana de errores, menor cantidad de errores después de liberados los productos).
2. Los métodos utilizados deben ser incrementados de forma que puedan ser implementados con una inversión relativamente corta, y ser extendidos de manera continua.
3. Los métodos utilizados deben dar resultados rápidos y tangibles de manera que pueda justificarse su continuación.
4. Los métodos utilizados deben hacer uso de tecnologías existentes para obtener el máximo beneficio del trabajo ya hecho en la empresa.

Las investigaciones realizadas por Hall (1990) y Bowen e Hinchey (1995) sobre la aplicación de métodos formales en la industria muestran que los métodos formales son ideales para cumplir con los requisitos antes mencionados, dando como conclusión que los métodos formales pueden ser aplicados de manera exitosa, especialmente por los casos que muestran las ventajas de su aplicación y la posibilidad de aplicarlos en etapas específicas del desarrollo, lo que permitiría a las empresas adaptarlos de manera gradual y aprovechando las herramientas ya existentes.

5. Conclusiones

Analizando la situación actual de las empresas de nuestro país y sus exigencias con respecto a la aplicación de nuevos métodos y procedimientos, se puede concluir que los métodos formales pueden ser aplicados en las empresas para mejorar sus procedimientos y productos. Desafortunadamente, el personal de las empresas cuenta con pocos fundamentos teóricos, esto agregado a la poca difusión de casos de éxito de métodos formales aplicados en nuestro país, origina que pocas empresas tengan conocimiento de su existencia y aún menos se atrevan a implementarlos en sus empresas. Por esta razón, los primeros pasos a dar para lograr su adopción en nuestro país serían los siguientes:

1. Se debe dedicar más tiempo y recursos a la investigación y difusión de la existencia de métodos formales, de manera que casos de éxito documentados e información sobre los métodos sean accesibles a las empresas y de esta manera puedan aplicarlos de acuerdo a sus necesidades.
2. Es necesario que el personal formado cuente con el suficiente respaldo teórico en el área, de tal forma que tenga la capacidad suficiente para aplicar los métodos en sus empresas de manera exitosa. Este respaldo

Tamaño	Número de empleados	Resultados	Porcentaje
Micro	1 – 10	48	39.0
Pequeña	11 – 50	53	43.0
Mediana	51 – 100	12	9.8
Grande	101 en adelante	10	8.2
Total		123	100

TAB. 1: CLASIFICACIÓN TOTAL POR TAMAÑO DE EMPRESA SECRETARÍA DE ECONOMÍA (2004)

debe darse desde la etapa de formación, lo que significa que debe fortalecerse la enseñanza de los métodos formales en las carreras afines a Tecnologías de la Información impartidas en nuestro país. Esto también propiciará su difusión no solamente en la industria, sino también en el ámbito académico fomentando la investigación sobre los métodos y su aplicación en el contexto de empresas como las PYMES que predominan en nuestro país.

Asimismo es importante recordar que, como en el caso de cualquier método de desarrollo que se desee aplicar en las empresas, es importante contar con el respaldo de la alta dirección si se quiere que la implementación de los métodos se lleve a cabo de manera exitosa. Así mismo, se debe contar con el apoyo de las instituciones encargadas en nuestro país que den respaldo y fomenten a las empresas a la investigación y aplicación de métodos que permitan mejorar su situación en el contexto de la economía nacional y también en el contexto de la competencia global a la que se enfrentan las empresas de nuestro país hoy en día.

6. Anexo: ¿Cuál es el costo de añadir el uso de métodos formales a los prácticas tradicionales?

Enrique Vázquez y Miguel Pérez Cerviño e Tumbeiro (2008) aplican dos métodos de desarrollo: el convencional y utilizando métodos formales en un caso práctico con el objetivo de hacer una comparación en el esfuerzo requerido y muestran también la calidad del software desarrollado con las dos metodologías. El software utilizado es un Sistema de Seguridad Compilable y tiene como funcionalidad: grabación de video digital, control automático de puertas, detección de intrusos, supervisión de guardias. Para este caso práctico, se utiliza VDM-SL Plat e Larsen (1992) para el desarrollo de un módulo del sistema y se hace partiendo de los mismos requisitos que en el desarrollo convencional. Se utiliza también IFDA con soporte de VDM-SL y C++.

Entre los principales beneficios se encuentra la reducción del código de 12000 líneas del código original a 5500 líneas. La especificación fue introducida a IFDA, un generador de código C++ sobre la plataforma Solaris. Al probar la calidad del software, se hicieron de una a siete pruebas para cada requisito. La calidad fue medida en función de qué tanto se cumplen los requisitos. El software desarrollado con el método formal cumplió en un 82% los requisitos, mientras que con el método convencional, se cumplió sólo el 62%.

El esfuerzo se midió en función del tiempo que se llevó para el desarrollo de ese módulo. Con el método convencional se ocuparon 8 semanas para la obtención de los requisitos y 20 semanas para el diseño e implementación, haciendo un total de 28 semanas. En cambio, con el método formal se llevaron 18 semanas para la especificación abstracta, 5 semanas para la especificación ejecutable, 5 semanas para la validación y media semana para la generación de código en C++, haciendo un total de 28 semanas y media.

La conclusión de la investigación fue que el incremento en esfuerzo para realizar el módulo resultó poco significativa, en comparación con el incremento en la calidad, el cual, si fue considerable.

Referencias

Baur, Plasa, Kejwal, Drexler, Reif, Stephan, Wolpers, Hutter, Sengler E Canver

1992 *«The Verification Support Environment VSE»*, in *IFA Symposium on Safety, Security and Reliability of Computers*.

Bowen, J. P. E Hinchey, M. G.

1995 *Seven More Myths of Formal Methods*, IEEE Software, vol. 12 (4), p. 34—41.

Cerviño, E. V. E Tumbeiro, M. F. P.

2008 *Métodos formales VS Métodos convencionales*.

Dürr, E. E Van Katwijk, J.

1992 «VDM++: a formal specification language for object-oriented designs», in *TOOLS 7: Proceedings of the seventh international conference on Technology of object-oriented languages and systems*, p. 63—77, Prentice Hall International (UK) Ltd., Hertfordshire, UK, UK.

Freitas, L., Woodcock, J. E Zhang, Y.

2009 *Verifying the CICS File Control API with Z/Eves: An experiment in the verified software repository*, Sci. Comput. Program., vol. 74 (4), p. 197—218.

Hall, A.

1990 *Seven Myths of Formal Methods*, IEEE Software, vol. 7, p. 11—19.

Hall, A.
1996 *Using Formal Methods to Develop an ATC Information System*, IEEE Softw., vol. 13 (2), p. 66—76.

Kurita, T., Chiba, M. E Nakatsugawa, Y.
2008 «Application of a Formal Specification Language in the Development of the "Mobile FeliCaÏC Chip Firmware for Embedding in Mobile Phone», in *FM '08: Proceedings of the 15th international symposium on Formal Methods*, p. 425—429, Springer-Verlag, Berlin, Heidelberg.

Langenstein, B., Vogt, R. E Ullmann, M.
2000 *The Use of Formal Methods for Trusted Digital Signature Devices, in Proceedings of the Thirteenth International Florida Artificial Intelligence Research Society Conference*», p. 336—340, AAAI Press.

León-Carrillo, Luis Vinicio
2005a *El Proceso de la Prueba de Software. Lenguajes de Definición de Procesos*, Software Gurú, p. 46—47.

León-Carrillo, Luis Vinicio
2005b *La especificación de un "Process Definition Language* (Reporte Interno), E-Quallity.

León-Carrillo, Luis Vinicio
2006 «La Industria de Prueba de Software en México», Software Gurú.

León-Carrillo, Luis Vinicio
2008 «The Impact of Software Testing in Small and Medium Settings», in OKTABA, H. e PIATTINI, M., curatori, «Software Process Improvement for Small and Medium Enterprises: techniques and case studies», cap. 4, p. 94—108, Information Science Reference.

Martínez, Christian A.
2007 «Combinación de UML y Métodos Formales para el Éxito de los Sistemas», Journal de Investigación. Escuela de Graduados e Innovación, ITESM Campus Puebla, p. 32—35.

Plat, N. E Larsen, P. G.
1992 «An overview of the ISO/VDM-SL standard», SIGPLAN Not., vol. 27 (8), p. 76—82.

Scott, L., Jeffery, R., Carvalho, L., D'ambra, J. E Rutherford, P.
2001 «Practical software process improvement - the IMPACT project», in «Software Engineering Conference, 2001.», p. 182—189.

Secretaría De Economía
2004 *Estudio del nivel de madurez y capacidad de procesos de la industria de tecnologías de información*.

Wing, Jeannette M.
1990 *A Specifier's Introduction to Formal Methods*, Computer, vol. 23 (9), p. 8—23.

Woodcock, Jim And Davies, Jim
1996 *Using Z: specification, refinement, and proof*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

1. http://www-01.ibm.com/software/htp/cics/
2. http://vasco.imag.fr/EDEMOI/
3. http://www.felicanetworks.co.jp/index.html
4. http://www.vdmtools.jp/en/
5. http://www.iso.org/iso/catalogue_detail.htm?csnumber=35168
6. http://www.siac.com.mx/siacframeset.htm
7. http://www.e-quallity.net/index.php