Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

Tipo de artículo: Artículo original
Temática: Ingeniería y Gestión de software
Recibido: 11/03/2013 | Aceptado: 30/04/2013

Experiencias en la ejecución de pruebas automáticas en Segurmática

Automated test execution: experiences in Segurmatica

Emma Torres Orue [1], Jorge Lodos Vigil [1], Ezequiel Sevillano Fernández [1], Guillermo Bernal Felipe [1], Marta Dunia Delgado Dapena[2]

[1] Segurmática, Calle Zanja esq. Soledad No. 651, Centro Habana, La Habana, Cuba. CP.: 10200
[2] Instituto Superior Politécnico José Antonio Echeverría, Calle 114, No. 11901, e/ Ciclovía y Rotonda Marianao, La Habana, Cuba. CP.: 11500

{emma, lodos, ezequiel, gbernal}@segurmatica.com
marta@ceis.cujae.edu.cu

Resumen

La realización de las pruebas de software permite a las empresas entregar productos que satisfagan las necesidades de sus clientes. La automatización de este proceso disminuye la demanda de tiempo y esfuerzo, así como la introducción de errores humanos. En este artículo se describe la experiencia en la ejecución de pruebas de software adquirida en Segurmática. En la empresa se ha llevado a cabo un proceso soportado por herramientas que permiten la ejecución automática de scripts de pruebas. Esta solución ha sido aplicada en proyectos que difieren entre sí en cuanto a arquitectura, lenguaje de programación, tamaño del equipo de desarrollo, complejidad y alcance. Los resultados arrojan mejoras en la realización de pruebas de la empresa en cuanto a organización, rapidez para la ejecución de pruebas y eficacia en la detección de errores.

Palabras clave: Automatización de pruebas, herramientas de pruebas, proceso de pruebas.

Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

Abstract

Performing software testing allows companies to deliver products that satisfy customer requirements. Automating this process reduces the demand of time and effort, and human error introduction. This article describes the experience on implementing software testing in Segurmatica. This company has enforced a tool supported process that allows automatic execution of test scripts. This solution has been applied in projects that differ in terms of architecture, programming languages, development team size, complexity and scope. The results show improvements in organization, tests realization speed and error detection effectiveness.

Keywords: *Test automation, test process, test tool.*

Introducción

La Empresa de Consultoría y Seguridad Informática, SEGURMÁTICA es una empresa estatal cubana perteneciente al Ministerio de Informática y Comunicaciones. Su misión estratégica es brindar los servicios de Seguridad Informática que sean demandados por entidades y particulares radicados en Cuba, sustituyendo importaciones y garantizando la seguridad del país. Durante los últimos 7 años se han realizado investigaciones relacionadas con la automatización de las pruebas (Torres, 2009, 2011) para aumentar la calidad de los productos entregados a los clientes.

En la Unidad Básica de Desarrollo se aborda el enfoque de la ingeniería de línea de productos (Hanssen, 2010, Da Mota, 2011). Se han creado cinco grupos especializados en determinados proyectos. A pesar de tener un objetivo general en común, las aplicaciones de cada grupo difieren entre sí en cuanto a tecnología, lenguaje de programación, complejidad y plataforma de desarrollo.

Dada la diversidad de aplicaciones producidas en Segurmática, se hizo inminente diseñar un proceso que permita organizar las pruebas en los proyectos de software llevados a cabo. Se requería una solución que estuviera orientada a disminuir el esfuerzo y el tiempo requeridos por los especialistas para probar las aplicaciones, reduciendo al máximo la introducción de errores. Para lograr esto, se necesitaba incorporar herramientas de manera tal que se aminorara, al máximo posible, la intervención humana y facilitara la reutilización de los componentes de pruebas. Suplementariamente se deseaba incorporar los sistemas de virtualización para suplir las necesidades de estaciones requeridas para las pruebas.

Diversas instituciones se dedican a la definición de modelos destinados a la calidad de software (ISO/IEC, 2003, Pinheiro, 2009), incluyendo normas dirigidas al trabajo con las pruebas (IEEE, 1987, 2008, 2012). Paralelamente se han confeccionado metodologías y procesos (Abrahamsson, 2010; Barnes, 2007; Pressman, 2006) donde se describen

actividades, roles y artefactos relacionados con la realización de las pruebas dentro del ciclo de vida del software. No obstante tanto la ejecución automática desatendida como los entornos de prueba virtualizados, involucran conceptos y procedimientos que no son descritos por las estrategias mencionadas.

Existen disímiles soluciones para automatizar la generación y ejecución de scripts de pruebas (Edwards, 2010, Schwarzl, 2010; Xie, 2007). Con el propósito de lograr la ejecución desatendida de componentes, a partir de modelos, codificación en lenguajes scripts o gestores de pruebas con interfaces de usuarios amigables (Bouquet, 2008; Davis, 2009; Levinson, 2011). Sin embargo, estas soluciones están concebidas para ejecutar scripts generados por herramientas de prueba específicas. Esto podría implicar el empleo de varias aplicaciones de ejecución desatendida para todas las pruebas automatizadas en la empresa.

Debido a la importancia de emplear amplios y diversos entornos de pruebas, con el fin de validar los artefactos en una línea de productos, se ha decidido incorporar el uso de máquinas virtuales para facilitar la creación y mantenimiento de los laboratorios de pruebas. La integración entre gestores de laboratorios virtuales (Burd, 2011; Matthews, 2008) y herramientas de pruebas (Davis, 2009; Levinson, 2011; Rice, 2012) permite probar aplicaciones en máquinas virtuales a partir de casos de pruebas definidos; grabar y reproducir la ejecución; así como almacenar los resultados y registrar las condiciones que exponen los errores para un posterior seguimiento. IBM *Rational Quality Manager* (Barnes, 2007) y *TestComplete* (Rice, 2012) son soluciones privativas brindadas por las empresas de software IBM *Rational* y SmartBear Software, respectivamente; son ejemplos de esta integración. La limitante de la herramienta de *Rational* radica en que los scripts de prueba solo pueden ser generados por productos comercializados por esta compañía (Barnes, 2007; Davis, 2009). Por otro lado, TestComplete está concebida para máquinas con sistemas operativos de Windows.

Materiales y métodos

En este artículo se describe un proceso que rige la ejecución desatendida de compontes de pruebas sobre laboratorios reales y virtuales. Este proceso es soportado por una herramienta que posibilita la ejecución, sin la intervención humana, de scripts generados por cualquier aplicación de prueba. Posteriormente se plasman los resultados adquiridos a partir de la aplicación del proceso en dos grupos del área de desarrollo en Segurmática.

Automatización de la ejecución de Pruebas

Las diferencias apreciadas en la implementación de las aplicaciones en los grupos de desarrollo, se evidencia en la fase de automatización de las pruebas igualmente. Algunos grupos que implementan en lenguaje C nativo, realizan

pruebas unitarias y funcionales mediante la biblioteca Boost (Reddy, 2011). Mientras otros que programan en la plataforma .NET, utilizan la herramienta *Microsoft Visual Studio Team Edition for Testers* (Torres, 2009) para pruebas unitarias, de base de datos y web. En algunas aplicaciones se realizan pruebas funcionales con la misma herramienta de desarrollo.

A continuación se describe el proceso concebido para organizar y estandarizar la ejecución de pruebas automática para todos los grupos de la empresa. El proceso es soportado por un sistema desarrollado en la empresa llamado QUALITY (Torres, 2011). Esta herramienta permite gestionar toda la información relacionada con las pruebas realizadas en el área de desarrollo; además de habilitar la ejecución de componentes de pruebas sin la intervención humana.

Proceso de ejecución desatendida de pruebas

El proceso está compuesto por tres fases: Generación de Componentes de Pruebas, Creación de Entornos de Pruebas y Ejecución y Obtención de Resultados. Las entradas son los módulos del sistema a probar, la documentación de sus requerimientos y las pruebas unitarias asociadas. Las salidas del proceso constituyen una base de conocimientos donde se almacena toda la información relacionada con la ejecución de las pruebas. Las fases de este proceso se deben realizar frecuentemente, a partir de la incorporación de nuevos módulos o modificaciones al Sistema en desarrollo.

El conjunto de roles del proceso, incluye a los roles convencionales en la etapa de pruebas de un software como son el Gestor de Pruebas, el Ejecutor de Pruebas y el Auditor. Adicionalmente se introducen los roles Experto y Administrador. El primero es el responsable de confeccionar y publicar los artefactos reutilizables para el resto de los especialistas como son las plantillas, sistemas operativos y programas informáticos. El segundo, es quien administra las computadoras y las asigna a desarrolladores y gestores de pruebas, según sus necesidades.

La primera fase comprende la creación de componentes de pruebas, la cual se realiza empleando soluciones para automatizar la ejecución de pruebas disponibles en el mercado (Barnes, 2007; Bouquet, 2008; Levinson, 2011; Xie, 2007). Se han diseñado bibliotecas de comunicación que serán empleadas durante la confección de los scripts para almacenar los resultados obtenidos y eventos, a partir de la ejecución de pruebas. También incluye la inserción de los parámetros de salida; donde se especifica si la ejecución resultó satisfactorio o no. Otras salidas la constituyen otros datos como la fecha y hora de inicio y fin de la ejecución, el usuario que inicia el proceso, el estado de las precondiciones, entre otras.

Para los componentes de pruebas generados por herramientas que no permitan el uso directo de la biblioteca de comunicación, se ha creado un ejecutable con función de mediador. Este binario es un componente de prueba capaz

de ejecutar otro script de prueba y almacenar sus resultados en la base de datos del Sistema QUALITY mediante la biblioteca de comunicación. Los parámetros de entrada del mediador serían los parámetros del script de prueba más el camino donde el componente de prueba se encuentra ubicado. Finalmente el Gestor almacena los scripts de prueba creados en el repositorio e introduce en el Sistema QUALITY sus nombres, parámetros y ubicaciones en el servidor.

Durante la etapa de Creación de Entornos de Pruebas el Administrador introduce en la herramienta QUALITY el nombre, sistema operativo y programas instalados en las computadoras habilitadas para la ejecución de las pruebas; así como los gestores que pueden operar con estas. Por su parte, el Experto registra la información de los sistemas operativos, programas informáticos y plantillas de máquinas virtuales disponibles para la creación de entornos de pruebas. El Gestor de Pruebas guarda los datos de sus máquinas virtuales y las distribuye entre las máquinas reales asignadas a él por el Administrador.

Finalmente, en la tercera etapa, los componentes de pruebas son agrupados en baterías de pruebas para ser ejecutados en un orden determinado. El gestor establece para cada script, la máquina de prueba (virtual o real) donde será ejecutado. Seguidamente, el Sistema realiza una copia del componente desde el servidor hacia la máquina de prueba. Los componentes de prueba dentro de una suite podrán ser ejecutados secuencial o simultáneamente en una o varias máquinas de pruebas. La suite de pruebas en QUALITY es denominada Proceso de Control de Calidad.

Los valores de los parámetros de entrada de los componentes de pruebas podrán variar para la misma suite de pruebas; así como los horarios en que esta puede ser ejecutada. La definición de un juego de valores de los parámetros de entrada de una suite de pruebas, la programación de horario y el grupo del usuario (Administrador, Usuario, Sistema) con que será ejecutada es denominada Instancia de Proceso de Control de Calidad. Este concepto permite que un mismo Proceso definido pueda ser ejecutado en distintos horarios y bajo distintas condiciones establecidas por los valores de los parámetros y las máquinas donde tendrán lugar. En este punto ya queda registrada la documentación necesaria para que un proceso sea ejecutado. La Instancia de Proceso puede ser ejecutada desde la interfaz web por petición del usuario; o según la programación de fecha y hora indicada.

El Auditor puede obtener los resultados y eventos relacionados con las ejecuciones de las Instancias de Procesos desde la interfaz web de QUALITY. Adicionalmente, durante la configuración de los Procesos, los Gestores de Prueba pueden definir direcciones de correos a las que se quieren enviar reportes con eventos y resultados. Estos reportes se pueden enviar al ocurrir diferentes condiciones, como son el inicio de la ejecución de una Instancia de Proceso o de una Prueba, la detección de un error tanto por parte del Sistema como por parte del componente de prueba y al finalizar la ejecución de la Instancia del Proceso o de una Prueba.

Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

Proyectos de desarrollo

El proceso definido ha sido incorporado en dos grupos de la Unidad Básica Empresarial de Desarrollo. Uno de los grupos tiene la responsabilidad de implementar las aplicaciones comercializadas por la empresa a clientes de todo el país. Este grupo ha aplicado la solución para realizar las pruebas en tres proyectos. El segundo equipo de trabajo confecciona herramientas de consumo interno en la empresa, especializados en asegurar y controlar la calidad de software de los productos. Se ha involucrado en el estudio un sistema multicapa, de arquitectura cliente-servidor desarrollado por este grupo. En la tabla se muestran las principales características de los proyectos seleccionados para aplicar el proceso de automatización de pruebas.

Tabla. Proyectos involucrados en la aplicación del proceso.

Características	Proyecto 1	Proyecto 2	Proyecto 3	Proyecto 4
Tipo de proyectos	Bibliotecas	Servicio de Windows	Páginas web	Aplicación web y Servicios Web
Lenguaje de programación	C nativo	C nativo	HTML, Javascript	ASP.NET, C#.NET
Base de datos	No	No	No	MS SQL Server
Tamaño del equipo de desarrollo	2	2	1	3
Etapa en la que se aplica el proceso	Pre-Producción	Planificación	Pruebas	Desarrollo
Alcance del despliegue	Todo el país	Todo el país	Todo el país	Empresa

Los proyectos 1, 2 y 3 se han desarrollado en el grupo de aplicaciones comerciales. Debido a la amplitud de su alcance de despliegue, se desea que estas aplicaciones sean multiplataforma capaces de funcionar en computadoras con bajas prestaciones. La selección de los lenguajes de implementación empleados, responden a estos requerimientos. El proyecto 4 pertenece al grupo de aplicaciones relacionadas con la calidad de software. En este grupo el factor más importante es la agilidad de entrega de las soluciones, lo cual puede ser alcanzado empleando una plataforma de programación de alto nivel.

Los grupos de desarrollo de la empresa son de formato pequeño en su totalidad. Los especialistas están deben desempeñar varios roles durante el proceso de desarrollo. Este hecho se evidencia en los proyectos 3 y 4 donde los desarrolladores, realizan las pruebas de las funcionalidades que implementan. Sin embargo, en los proyectos 1 y 2 se habilitaron probadores que no estuvieron involucrados en la implementación del código.

En la tabla se incluye la característica que indica la etapa del ciclo de desarrollo en la que se encontraba cada proyecto cuando se comenzó la aplicación del proceso propuesto. En el proyecto 1 se había concluido la

Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

implementación y las pruebas de las bibliotecas. En el proyecto 3 se ha aplicado después de finalizada toda la implementación de las páginas web. El proyecto 4 se introdujo en el estudio varios meses después de iniciada la fase de desarrollo. Mientras que en el proyecto 2, según es recomendado por el colectivo de autores de este trabajo, se adoptó la solución propuesta desde la etapa de planificación del proyecto.

Los cuatro proyectos han sido seleccionados en base a sus arquitecturas, alcances y etapas de desarrollo. Con la selección realizada se pretendía demostrar que el proceso es aplicable para distintos proyectos de desarrollo, independientemente de sus características. En el próximo epígrafe se reflejan los resultados obtenidos.

Resultados y discusión

La estandarización y organización sobre la realización de pruebas en el área de desarrollo de Segurmática ha sido el logro principal alcanzado producto de la aplicación de la solución propuesta. En este segmento se hará referencia a los resultados alcanzados en los dos grupos donde se realizó el estudio. La captura y el almacenamiento centralizado de la información relativa a la ejecución de componentes de pruebas permiten obtener estadísticas que demuestran las mejoras alcanzadas referentes a la velocidad y eficiencia en el proceso de pruebas.

El proceso para la ejecución automática de pruebas definido ha propiciado concentrar los scripts de prueba generados en los dos grupos de desarrollo y mantener un registro de sus descripciones, parámetros, ubicación y resultados de sus ejecuciones. Este proceso viabilizó la reutilización de componentes de pruebas; lo cual posibilita la ejecución de scripts de pruebas dentro de diferentes entornos de pruebas y bajo distintas condiciones establecidas a través de sus parámetros de entrada. De esta manera se logró aumentar el alcance de las pruebas. En la figura 1 se muestra una gráfica que contiene el número de componentes de pruebas confeccionados para los proyectos vinculados en el estudio realizado.

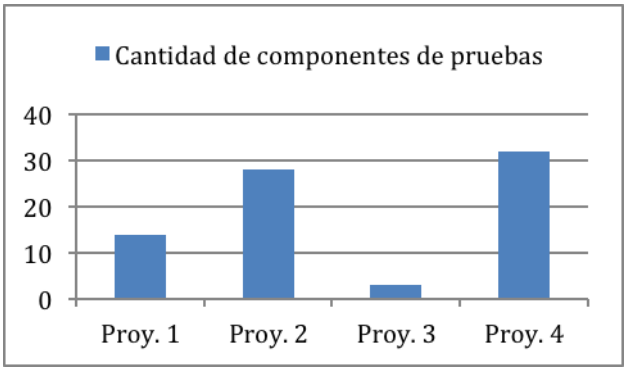

Figura 1. Cantidades de componentes de pruebas para los proyectos del estudio.

Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

Un beneficio importante incorporado en el ciclo de vida de los proyectos ha sido la detección temprana de defectos. Previamente a la aplicación de esta solución, las pruebas eran ejecutadas en las estaciones de trabajo de los desarrolladores, por lo tanto los errores relacionados con los ambientes de funcionamiento del software, podrían ser detectados por los clientes en la fase de producción. La posibilidad de realizar pruebas de componentes, de integración y funcionales en varias máquinas de pruebas ha permitido revelar errores vinculados a los entornos en etapas tempranas del desarrollo de la aplicación.

La ejecución semanal de las pruebas, llevada a cabo a través de la herramienta que da soporte al proceso, ha propiciado la manifestación de errores en pocos días, en ocasiones horas, de diferencia respecto a la fecha de su introducción en el código. Por otro lado, la ejecución programada ha brindado mejoras orientadas a la realización de pruebas de integración y de regresión. Los desarrolladores no solo han podido estar al tanto del buen funcionamiento de su código, sino de que la integración de este con el resto de la aplicación sea satisfactoria; manteniendo un control del desarrollador que ha podido introducir el error. En la siguiente figura se plasma para cada proyecto el número de defectos encontrados a partir de pruebas funcionales y de integración ejecutadas automáticamente.

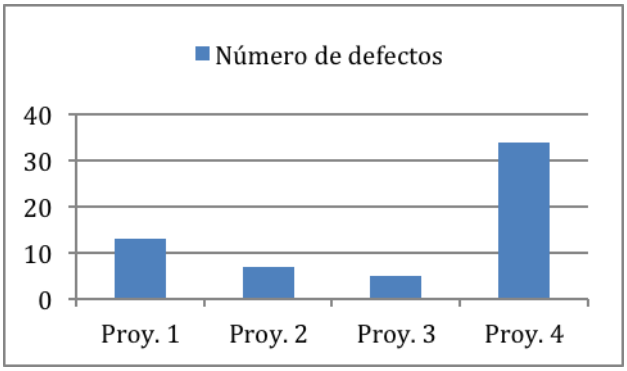

Figura 2. Cantidades de defectos encontrados en los proyectos del estudio.

La solución propuesta ha permitido registrar los tiempos empleados en las ejecuciones de scripts de prueba. Dicha información no se archivaba previamente por los especialistas que ejecutaban las pruebas de forma manual. Por lo tanto no era posible controlar y por ende, planificar los tiempos necesarios para la ejecución de las pruebas de un proyecto dado. En la figura 3 se muestra la suma de los tiempos promedio de ejecución de todas las baterías de pruebas configuradas para los proyectos seleccionados. Estos tiempos fueron obtenidos solo de las ejecuciones que resultaron satisfactorias, no se tuvieron en cuenta las ejecuciones interrumpidas por causa de errores.

Revista Cubana de Ciencias Informáticas
Vol. 7, No. 2, Abril-Junio, 2013
ISSN: 2227-1899 | RNPS: 2301
Pág. 145-156
http://rcci.uci.cu

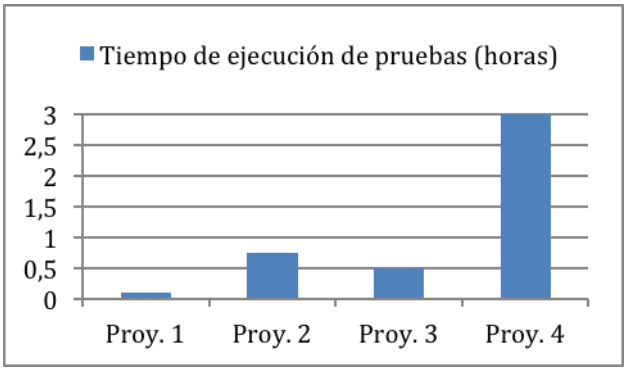

Figura 3. Cantidad de horas empleadas en la ejecución de las baterías de pruebas de los proyectos del estudio.

Una mejora relacionada con la organización del proceso de pruebas manifestada ha sido la centralización y publicación de los resultados. De esta manera se han mantenido informados sobre el progreso del desarrollo de las aplicaciones a jefes de proyecto, comerciales, administrativos y todo el personal interno de la empresa vinculado con los proyectos. Especialmente se han visto reflejados las dificultades encontradas y aquellas a las que se les he encontrado solución.

A pesar de las diferencias que presentan los proyectos seleccionados, en los cuatro proyectos se aprecian mejoras en cuanto a organización, control y detección de errores. Aunque no se haya llevado un registro formal del tiempo empleado en las ejecuciones de pruebas antes de la puesta en práctica del proceso, es sabido a partir de encuestas y entrevistas con los especialistas de los grupos, que se ha ahorrado significativamente en tiempo y esfuerzo por parte de ellos.

Conclusiones

En el presente trabajo se ha definido un proceso destinado a estandarizar la ejecución desatendida de componentes de pruebas en Segurmática. Esta empresa está enfocada en el desarrollo de una línea de productos de seguridad informática. El proceso cuenta de tres fases: Generación de componentes de pruebas, Preparación de entornos de pruebas y Ejecución. El proceso propone el registro y control de los scripts de pruebas y de las máquinas que integran los laboratorios de prueba, asumiendo la inclusión de máquinas virtuales. Se ha implementado una herramienta para dar soporte al proceso descrito, la cual facilita la generación de artefactos y permite la ejecución desatendida de componentes de pruebas. La solución ha sido aplicada en cuatro proyectos desarrollados en dos grupos área de Desarrollo de la empresa. Estos proyectos se diferencian entre sí en cuanto a complejidad de la arquitectura, lenguaje de programación, alcance de despliegue y tamaño de los equipos de trabajos. A partir de las estadísticas obtenidas con

la herramienta de soporte se ha podido demostrar que el proceso ha mejorado la realización de pruebas a las aplicaciones de software seleccionadas en cuanto a disponibilidad de la información, eficiencia en la detección de errores y velocidad en la ejecución de componentes de pruebas.

Agradecimientos

A los especialistas del área de Desarrollo de Segurmática que intervinieron en la puesta en práctica de la solución planteada, por su trabajo y sus opiniones que ayudaron a mejorar el proceso diseñado.

Referencias

- ABRAHAMSSON, P., OZA, N., SIPONEN M. T. Agile Software Development Methods: A Comparative Review. En: DINGSØYR, T. et al. (editores.). Agile Software Development Current Research and Future Directions. Berlin: Springer, p. 31-53, 2010.
- BARNES, J. Implementing the IBM® Rational Unified Process® and Solutions: A Guide to Improving Your Software Development Capability and Maturity. Upper Saddle River NJ, IBM Press, p. 216, 2007.
- BOUQUET, F. et al. A Test Generation Solution to Automate Software Testing. En: Proceedings of 3rd International Workshop on Automation of Software Test (AST'2008). New York: ACM, p. 45 – 48, 2008.
- BURD, S. D. et al. Virtual Computing Laboratories Using VMware Lab Manager. En: Proceedings of 44th Hawaii International Conference on System Sciences (HICSS' 2011), Washington: IEEE Computer Society, p. 1 – 9, 2011.
- DA MOTA SILVEIRA, P.A.; RUNESON, P. et. al. Testing Software Product Lines, IEEE Software, Vol. 28 (5): p.16 – 20, 2011
- DAVIS, C. et al. Software Test Engineering with IBM Rational Functional Tester: The Definitive Resource. Upper Saddle River NJ, IBM Press, p. 696, 2009
- EDWARDS, A., TUCKER, S., DEMSKY, B. AFID: an automated approach to collecting software. Automated Software Engineering, vol. 17 (3): p. 347 – 372, 2010.

- HANSSEN, G. K.: Opening Up Software Product Line Engineering. En: Proceedings of Workshop on Product Line Approaches in Software Engineering. 32nd International Conference on Software Engineering. New York: ACM, p. 1–7, 2010.
- IEEE. Standard for Software Unit Testing. IEEE 1008. IEEE Computer Society. Nueva York. EUA. 1987.
- IEEE. Standard for Software Test Documentation. IEEE 829. IEEE Computer Society. Nueva York, EUA, 2008.
- IEEE. Standard for Software Verification and Validation. IEEE 1012. IEEE Computer Society. Nueva York, EUA. 2012.
- ISO/IEC. Software Engineering - Product Quality - Part 1: Quality Model. ISO/IEC 9126-1. ISO/IEC Office. Ginebra. Suiza. 2001.
- LEVINSON, J. Software Testing with Visual Studio® 2010. Addison-Wesley Professional, 2011. 336 p.
- MATTHEWS, J. N.; DESHANE, T. *et al.* Running Xen: A Hands-On Guide to the Art of Virtualization. Upper Saddle River, NJ, Prentice Hall, 2008. 624 p.
- PRESSMAN, R. Ingeniería del software, un enfoque práctico. Ciudad de México, McGraw-Hil Interamericana, 2006. 958 p.
- PINHEIRO, R.; OLIVEIRA K. M. y PEREIRA W. Evaluating the service quality of software providers appraised in CMM/CMMI, Software Quality Journal, 2009, Vol. 17 (3): p.283-301.
- REDDY, M.: Chapter 10: Testing. En: API Design for C++. Burlington: Morgan Kaufmann, 2011, p. 218 - 328.
- RICE, N., y TREFETHEN, S. TestComplete Version 8 Made Easier: Keyword Testing. California, Falafel Software Inc, 2012. 337 p.
- SCHWARZL, C. y PEISCHL, B. Generation of Executable Test Cases Based on Behavioral UML System Models. En: Proceedings of 5th Workshop on Automation of Software Test (AST'2010). New York: ACM, 2010, p. 31 - 34.
- TORRES, E.; HERNÁNDEZ F.; LODOS J. *et al.* Aseguramiento de la Calidad para un Sistema Ciente - Servidor en la plataforma .NET. En: Memorias del XIII Convención y Feria Internacional

de Informática. IV Taller de Calidad en las Tecnologías de la Información y las Comunicaciones. La Habana: 2009.

- TORRES E.; SEVILLANO, E. y LODOS J. Herramienta para la ejecución de componentes de pruebas. En: Memorias de la XIV Convención y Expo Internacional de Informática. V Taller Internacional de Calidad en las Tecnologías de la Información y las Comunicaciones. La Habana: 2011.
- XIE, Q. y MEMON, A. M. Designing and comparing automated test oracles for GUI-based software applications. ACM New York, 2007, vol. 16 (1), p. 4-art.