Tecnologías emergentes de la información y desarrollo de software

Felipe Antonio Román Albores

- Tecnología
- 01.09.2014
- 19 minutos de lectura

Resumen

En la actualidad se viven constantes cambios en el ámbito de las Tecnologías de la Información; una de las ramas que más cambios tiene es la Ingeniería de Software. Como describe Román (2011) la Web ha evolucionado de manera significativa pasando por 3 etapas hasta ahora. La primera fue la Web 1.0 las cuales estaban accesibles a personas especializadas que eran las encargadas de crear y mantener el contenido para ser consultado en la Web. Cuando se desarrollaron plataformas para que cualquier persona creara contenido Web sin conocimientos especializados o técnicos y se crearon las primeras redes sociales se hablaba de la evolución de la Web 1.0 a la denominada Web 2.0. Actualmente se habla de una nueva evolución llamada Web 3.0 o Web Semántica que trata de dotar de significado a los recursos existentes en la Web.

Para llevar a cabo los desarrollos tecnológicos que se requieren en esta evolución e innovación constante la Ingeniería en Tecnologías de la Información y Desarrollo de Software se apoyan de otras áreas derivadas como son los Sistemas Embebidos, Cómputo Móvil y Ubicuo, Interfaces Humano-Computadora, Sistemas Distribuidos y Bases de Datos No Relacionales, Redes de Próxima Generación, Arquitectura de Software, Patrones de diseño, Calidad en el Desarrollo de Software y Desarrollo Dirigido por Pruebas, Desarrollo de Software Orientado a Servicios, Modelos y Metodologías de Desarrollo de Software como base para converger y crear las aplicaciones y dispositivos electrónicos necesarios para responder a la constante demanda de información y conocimiento que las personas requieren. A continuación se abordan las tecnologías más relevantes que se enfocan en el área de Tecnología de la Información y Desarrollo de Software.

1. Sistemas Embebidos

Según Calva et. al. (2012) los sistemas embebidos son aquellos sistemas informáticos que forman parte de sistemas más grandes y que llevan a cabo algunos de los requisitos necesarios para la operatividad de estos sistemas. Los sistemas integrados cubren una amplia gama de sistemas informáticos de pequeños dispositivos basados en computadoras de ultra a grandes sistemas de vigilancia y control de procesos complejos.

La inmensa mayoría de los sistemas informáticos pertenece a los sistemas integrados. La mayoría de estos sistemas integrados se caracterizan también como sistemas de tiempo real, lo que significa que las propiedades de tiempo real, tales como el tiempo de respuesta, tiempo de ejecución son importantes problemas de diseño. El aumento de la complejidad de los sistemas embebidos de tiempo real conduce a la creciente demanda en relación con la ingeniería de requisitos, diseño de alto nivel, la detección temprana de

errores, la productividad, la integración, verificación y mantenimiento, lo que aumenta la importancia de una gestión eficiente de las propiedades del ciclo de vida tales como la mantenibilidad, portabilidad y capacidad de adaptación así como la calidad implementada en los procesos de desarrollo de software.

De acuerdo con Toro y Cardona (2010) se han considerado un sin número de procesos para ser automatizados, en su mayoría mediante el control de un software o en su defecto por dispositivos que integren software embebido para su manipulación. Es por ello que cada vez más la Ingeniería de software está trabajando en mejorar la calidad de los productos y procesos para hacer que dichos sistemas de control y automatización no contengan errores.

2. Cómputo Móvil y Ubicuo

Hoy en día tiene una relevancia cada vez mayor el uso de dispositivos electrónicos inmersos en nuestras vidas. Esto se debe en parte a los diferentes dispositivos que se interconectan en nuestro entorno ya sea en la oficina, escuela, hogar, o en el transporte público. Según Zapata (2012), La tecnología ubícua permite a los individuos aprender allí donde estén, y contar para ello con los componentes de su entorno social. En entornos de aprendizaje en línea es importante acceder a los recursos de manera tal que los individuos no tengan que preocuparse por la forma o dispositivos que se requieren para conectarse y consumir los objetos de aprendizaje de las plataformas en línea.

Una de las principales características del cómputo móvil y ubicuo es acceder a los recursos a los que se tienen acceso mediante los dispositivos conectados a internet sin siquiera pensar de qué manera estamos logrando la comunicación entre los equipos o dispositivos. Esta área de conocimiento de las Tecnologías de la Información está cobrando cada vez más importancia ya que continuamente se están mejorando las plataformas y tecnologías de interconexión o redes de próxima generación.

La computación móvil y ubicua está usándose junto a los sistemas distribuidos, sistemas embebidos entre otros para hacer realidad un nuevo concepto tecnológico denominado: Internet de las Cosas (del Ingles Internet of Things) donde Haller (2010) la cual define como una red de datos que permite la consulta de la información acerca de los objetos del mundo real por medio de un identificador único llamado Código Electrónico de Producto y un mecanismo de resolución. El internet de las cosas utiliza técnicas de la web semántica para dotar de significado a los objetos y datos obtenidos de ellos y al hablar de objetos se entiende que son dispositivos electrónicos.

3. Redes de próxima Generación

La sociedad en la que actualmente vivimos demanda cada vez más conectividad de los dispositivos electrónicos para acceder a los medios de información, recursos disponibles en internet así como la interconectividad entre sus pares. Es por ello que ahora se habla de la sociedad de la información y del conocimiento de acuerdo con Quiroz (2005) es el proceso que se realimenta a si mismo donde las nuevas tecnologías facultan a la sociedad en el manejo de grandes volúmenes de información, las cuales a su vez, generan más conocimiento en un círculo virtuoso ascendente de progreso.

La organización UIT-T Y (2014) define a estas redes en una red basada en paquetes que permite prestar servicios de telecomunicación y en la que se pueden utilizar múltiples tecnologías de transporte de banda ancha propiciadas por la QoS (Quality of Service), y en la que las funciones relacionadas con los servicios son

independientes de las tecnologías subyacentes relacionadas con el transporte. Permite a los usuarios el acceso sin trabas a redes y a proveedores de servicios y/o servicios de su elección.

De acuerdo con IHS (2006) la diferencia de las redes especializadas en proveer un servicio específico como las actuales, las NGN son una red multiservicio. Las redes de la próxima generación (NGN) anuncian el paso al enfoque de muchos servicios a través de una sola red, es básicamente una red que unifica voz, datos y video bajo la plataforma IP convirtiéndola en una red en la cual es posible prestar diferentes servicios, Restrepo (2009).

4. Interfaces Humano Computadora

La interacción Humano-Computadora se encarga de estudiar la relación de la interacción entre el Hombre y las computadoras, buscando hacerla entendible y fácil de usar según Abud (2006). Con esta definición se infiere que cualquier dispositivo electrónico que interactúe con un humano ya sea mediante Hardware o Software. Para Pérez (2014) el objetivo primordial de la interacción de usuarios de dispositivos electrónicos sea eficiente debe de minimizar los errores, incrementar la satisfacción del usuario, disminuir la frustración referente al uso del dispositivo. En resumen debe hacer más eficientes y productivas las tareas rutinarias y de trabajo de las personas con los dispositivos electrónicos.

Para el desarrollo de nuevas tecnologías o plataformas, lenguajes de programación o software es importante tomar en cuenta la Interfaces que estarán en contacto con los usuarios finales ya que de ello depende en gran medida el éxito o fracaso del sistema de software o hardware que interactúa con el usuario final. Cada vez más las empresas de software se están fijando en dos conceptos que se relacionan entre si los cuales son la usabilidad y accesibilidad. La usabilidad se refiere al alcance en el que un producto puede ser utilizado por usuarios específicos para alcanzar metas específicas y la accesibilidad es la posibilidad de que un producto o servicio pueda ser accedido y usado de forma independiente de las limitaciones propias del individuo o de las derivadas del contexto de uso Martínez (2014). La usabilidad en una aplicación cumple con su función principal siempre que se utilice con efectividad, eficiencia y satisfacción en un contexto específico de uso y la accesibilidad debe de hacer posible usar esa aplicación sin importar las capacidades diferentes del individuo ya sea visual, motriz o alguna otra.

5. Sistemas Distribuidos y Bases de Datos No Relacionales

Las aplicaciones tienen que almacenar una vasta cantidad de información, esto se debe a la evolución constante de la Web. Al comenzar las redes sociales y la Web 2.0 se comenzaron a diseñar aplicaciones basadas en sistemas Distribuidos ya que la velocidad de respuesta debe ser mayor a la que ofrece un servidor central de aplicaciones. Los sistemas distribuidos ahora son muy utilizados sobre todo por el nuevo concepto BigData el cual según Barranco (2012) es la tendencia en el avance de la tecnología que ha abierto las puertas hacia un nuevo enfoque de entendimiento y toma de decisiones, la cual es utilizada para describir enormes cantidades de datos (estructurados, no estructurados y semiestructurados) que tomaría demasiado tiempo y sería muy costoso cargarlos a un base de datos relacional para su análisis.

De tal manera el concepto de Big Data aplica para toda aquella información que no puede ser procesada o analizada utilizando procesos o herramientas tradicionales es por ello que muchas aplicaciones utilizan Bases de Datos No Relacionales para almacenar la información.

Existen plataformas como Hadoop (2009), es un marco de referencia que permite el procesamiento distribuido de grandes conjuntos de datos a través de grupos de computadoras que utilizan modelos de programación simples. Es decir que no requiere que la arquitectura o hardware sea muy costosa o de última generación. En cuanto a las bases de datos no relacionales Hadoop se integra perfectamente con Cassandra (2008) la cual es una distribución de un almacén de datos clave-valor altamente escalable y estructurada. Entre las dos aplicaciones dan soporte y crecimiento horizontal a las necesidades de las aplicaciones distribuidas como es el BigData para tener disponible la información a diferentes empresas tales como Facebook entre otras.

6. Arquitectura de Software

De acuerdo con Cervantes (2010) la Arquitectura de Software se refiere a la forma de cómo se estructura el diseño de un sistema, este se crea en etapas tempranas del desarrollo. La finalidad de estructurar los componentes o módulos así como el diseño del sistema tiene los propósitos de: satisfacer atributos de calidad en cuanto a desempeño, seguridad, mantenibilidad, y servir como guía en el desarrollo del proyecto. El objetivo de la arquitectura de Software consiste en desarrollar sistemas de software grandes de forma eficiente, estructurada y con capacidad de reutilización.

La Arquitectura de Software es una de las áreas los Ingenieros de Software integran en los diseños de software que realizan. Esto se debe a que la experiencia y el tiempo que tienen desarrollado proyectos les demandan tener un diseño arquitectónico de referencia. En la figura 1 se observa los pasos a seguir para definir una Arquitectura de Software:

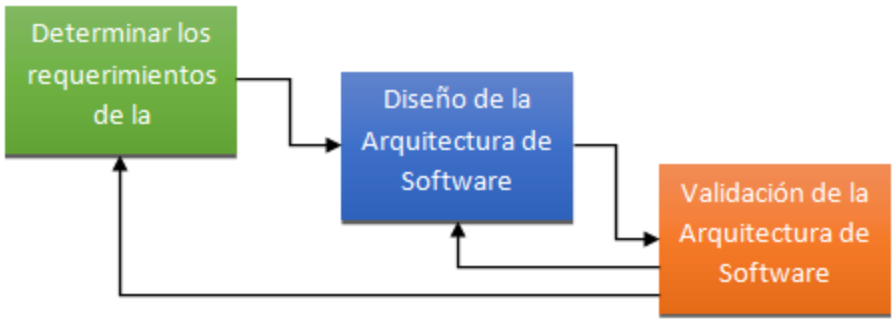

Figura 1. Etapas del diseño de una arquitectura de Software, Barraza (2014).

Determinar los requerimientos de la Arquitectura de Software: involucra crear un modelo desde el levantamiento de requerimientos que guiaran el diseño de la arquitectura basado en atributos de calidad esperado.

Diseño de la Arquitectura de Software: definir la estructura y las responsabilidades de los componentes que comprenderán la Arquitectura.

Validación de la Arquitectura de Software: básicamente se prueba la arquitectura pasando a través del diseño contra los requerimientos actuales y cualquier requerimiento futuro.

En Camacho et al. (2004), dice que en la medida que los sistemas de software crecen en complejidad, bien sea por número de requerimientos o por el impacto de los mismos, se hace necesario establecer medios para el manejo de esta complejidad. Las Arquitecturas de Software más comunes van desde Aplicaciones Monolíticas, Arquitectura Cliente-Servidor, Arquitectura Cliente-Servidor Mejorada, Arquitectura de 3 niveles, Arquitectura de N-niveles y Arquitectura Orientada a Servicios (SOA).

Los Ingenieros de Software que comienzan a desarrollar plataformas tecnológicas robustas y escalables necesitan implementar diferentes técnicas para mejorar su codificación. Una de ellas es la identificación de problemas repetitivos en la programación por lo que se emplean patrones de diseño de software para dar solución a los problemas que se necesitan solventar.

7. Patrones de Diseño

Los patrones de diseño de software son una parte importante en el diseño de la Arquitectura y en el Desarrollo de software en específico de la codificación de las aplicaciones. Estos patrones tienen su origen en el diseño de sistemas con el paradigma de la Programación Orientada a Objetos y a problemas que estos no pueden resolver y que son muy recurrentes en la programación de software. De acuerdo con Gamma et al. (1995) un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular.

Los patrones de diseño ayudan a mantener un código reutilizable y tener mayor control sobre los problemas que son recurrentes, de ahí que sean una solución factible para su implementación en múltiples aplicaciones.

Estos patrones son parte fundamental para el desarrollo de soluciones modulares, mantenibles y escalables. Los patrones de diseño se integran junto al modelado de sistemas con UML para identificar claramente los patrones empleados en cada componente, módulo o clase del sistema que se desarrolle. Este modelado forma parte de la Arquitectura de Software.

Se tienen 3 categorías de patrones de diseño que se aplican dependiendo de los problemas identificados en el desarrollo de software: patrones creacionales, patrones estructurales y patrones de comportamiento.

8. Desarrollo de Software Orientado a Servicios

Este tipo de Arquitectura de Software, es una de las arquitecturas más usadas para el desarrollo de soluciones de software robustas, escalables y de alto desempeño. De acuerdo con Torre et al. (2010) las aplicaciones orientadas a servicios permiten a una aplicación ofrecer su funcionalidad como un conjunto de servicios para que sean consumidos por clientes y se especializan en proporcionar un esquema basado en mensajes de nivel de aplicación. Las características principales de este tipo de aplicaciones es su alta disponibilidad debido en parte a que cada servicio es autónomo y no afecta a la aplicación en su totalidad si llegará a fallar. Los clientes y servicios son autónomos y pueden ser consumidos de manera remota a través de la red. El mantenimiento de sistemas robustos no es complicado ya que se actualizan los servicios o se crean nuevos sin afectar directamente a los demás o la aplicación cliente que los consuma. Según Somerville (2005) existen 3 estándares fundamentales para la comunicación o publicación de servicios Web los cuales son:

- SOAP (del Ingles Simple Object Access Protocol): este estándar define una organización para el intercambio de datos estructurados entre servicios Web.
- WSDL (del Ingles Web Services Description Language): este protocolo define cómo representar las interfaces de servicios Web.
- UDDI(del Ingles Universal Description, Discovery and Integration): este estándar de búsqueda define como se puede organizar la información de descripción de servicios, usada por los solicitantes de los servicios para encontrar servicios.

Para los Ingenieros de Software es importante tener las habilidades necesarias para desarrollar plataformas tecnológicas que implementen la Arquitectura Orientada a Servicios (SOA) que tiene una gran penetración en el mercado de las aplicaciones empresariales, por sus bondades y características de alta disponibilidad.

9. Calidad en el Desarrollo de Software y Desarrollo Orientado a Pruebas

Uno de los propósitos de la Ingeniería de Software según Kendall y Kendall (2005) son el aseguramiento de la calidad de los productos Software diseñando los sistemas en un enfoque modular; documentar el software con las herramientas adecuadas y por último probar, mantener y auditar el Software.

Una de las técnicas más utilizadas e implementadas en los últimos años es el desarrollo de Software dirigido a pruebas (del inglés TDD) donde Blé et al. (2010) propone toda una técnica para disminuir los problemas relacionados con el desarrollo de software tradicional en especial a los métodos tradicionales como el de cascada. Esta técnica se enfoca en 3 puntos principales:

- Implementación de funciones justas que el cliente necesita y no más.
- Minimización de número de defectos que llegan al Software en fase de producción.
- Desarrollar Software modular, altamente reutilizable y preparado para las adecuaciones.

El desarrollo de Software Dirigido por Pruebas es un nuevo paradigma que ofrece la facilidad de pensar en las Pruebas de Software antes que en la codificación, logrando con minimizar lo más posible los errores generados en etapas de desarrollo.

La calidad en el Desarrollo de Software esta intrínsecamente relacionado con la calidad en el proceso de Software para crearlo. A través de la implantación de procesos de desarrollo probados, bien documentados e institucionalizados, una compañía es capaz de incrementar la precisión de sus planificaciones, dando la posibilidad de establecer compromisos con sus clientes que antes no podía o no era capaz de asumir Jiménez (2012).

10. Modelos y Metodologías de Desarrollo de Software

Al hablar de la implementación de un modelo o una metodología para el desarrollo de software estamos haciendo referencia a la calidad de los productos de software. La obtención de un Software con calidad según Fernández, García, & Beltrán (1995), implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba del software que permitan unificar el trabajo, tanto para la labor de desarrollo como para el control de la calidad del software. Esto se debe a que los modelos tienen definidas las áreas de proceso a realizar para llevar a cabo la gestión de proyectos de

software con los diferentes roles que intervienen. Existen multitud de modelos para la gestión de la calidad del software y otros sistemas y normas de gestión que se han aplicado sobre estos procesos para su evaluación Martín-Avil (2010).

En este apartado se hablará de la metodología ágil para la administración de proyectos de Software SCRUM y el modelo mexicano para el desarrollo de Software MoProSoft y el modelo internacional CMMI.

La metodología ágil SCRUM de acuerdo con Palacio & Ruata (2011) es aplicado al desarrollo de Software donde se emplea el principio ágil del desarrollo iterativo e incremental llamado comúnmente sprint a cada una de las iteraciones de desarrollo. En la figura 2 se observa el modelo de SCRUM así como los componentes que lo integran.

Figura 2. Metodología SCRUM.

Ventura & Peñaloza (2006) define a MoProSoft como un modelo de procesos para la industria de software nacional (México), que fomenta la estandarización de su operación a través de la incorporación de las mejores prácticas en gestión e ingeniería de software.

Los beneficios de MoProSoft son los siguientes:

- Las empresas logran un mayor control sobre su desempeño en el mercado al cubrir el modelo desde áreas de dirección hasta la operación.
- El costo de personal se reduciría si se enfoca a la educación y a la capacitación sobre un modelo.
- Las empresas pequeñas, al seguir procesos similares, podrían asociarse con mayor facilidad para afrontar proyectos de mayor envergadura.
- La exportación de servicios de software de las empresas mexicanas gracias a que MoProSoft considera las prácticas reconocidas en el ámbito internacional.

Cabe señalar que la norma MoProSoft ha ayudado a la industria de software en México a estandarizar procesos y madurar en cada uno de los niveles que tiene los cuales son 5 si no se cuenta el nivel 0.
El modelo Internacional (del Inglés Capability Maturity Model Integration) se ha convertido mundialmente en un requisito para acceder a la exportación de servicios de Software. La norma CMMI-DEV (2010) provee

una guía para implementar una estrategia de calidad y mejorar los procesos de una organización que se dedica al desarrollo y/o mantenimiento de software. Dispone de un esquema de certificación creado sobre organismos privados donde el SEI (del Ingles Software Engineering Institute) creo la norma para la estandarización de áreas y procesos para lograr una mejor integración de los desarrollos de las empresas que lo manejan. Al ser una norma Internacional hace más fácil la integración de procesos de una empresa a otra aunque se encuentren en diferentes países o continentes. CMMI tiene 5 niveles los cuales certifica a 4 de ellos.

Conclusión

En este artículo se abordan diferentes tecnologías de la información, metodologías y paradigmas de desarrollo de software y como se complementan entre sí para beneficio de las necesidades que tiene constantemente la sociedad de la información y el conocimiento.

La evolución de nuevas plataformas tecnológicas brinda una oportunidad importante para que diferentes áreas de las Tecnologías de la Información converjan para un fin común que es desarrollar una solución tecnológica robusta, escalable, mantenibles y sobre todo empleando calidad en su proceso desarrollo que se manifiesta en la calidad del producto desarrollado.

Es por ello que los ingenieros, desarrolladores y especialistas en Tecnologías de la Información y Desarrollo de Software deben estar en constante actualización para solventar las necesidades de las empresas, organizaciones o entidades que lo soliciten.

Bibliografía

- Abud, Ma. Antonieta. (2006).”Diseño de Interfaces Humano-Computadora en Aplicaciones de Software Educativo”. Revisado el 19 de agosto de 2014. [En línea] **http://www.repositoriodigital.ipn.mx/bitstream/handle/123456789/5334/41-42-3.pdf?Seque-nce=3**.
- Barranco, Ricardo. (2012).”¿Qué es BigData?”. IT Specialist for Information Management, IBM Software Group México. México D.F. 18 de Junio de 2012. Revisado el 22 de agosto de 2014. [En línea] **http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data**.
- Barraza, Fernando. (2014). “Modelado y Diseño de Arquitectura de Software”. Conceptos de Modelado. Revisado 17 de agosto de 2014. [En línea] **http://cic.puj.edu.co/wiki/lib/exe/fetch.php?media=materias:s2_conceptosdemodelado.pdf**.
- Blé, Carlos et al. (2010). “Diseño Ágil con TDD” Primer Edición enero de 2010, www.iExpertos.com Licencia Creative Commons. ISBN: 978-1-4452-6471.4.
- Camacho, Erika. et al. (2004). “Arquitectura de Software”. Guía de Estudio. [En línea] **http://prof.usb.ve/lmendoza/Documentos/PS-6116/Guia%20Arquitectura%20v.2.pdf**.
- Cassandra. (2008). “Cassandra Wiki”. Revisado el 22 de agosto de 2014. [En línea] **http://wiki.apache.org/cassandra/FrontPage**.
- Calva, C. A et al. (2012). “Design of a Low Cost Electronic System for Automotive Steering Controlling” International Journal of Computer and Communication Engineering, Vol. 1, No. 4,

Noviembre 2012. Revisado el 22 de agosto de 2014. [En línea] **http://www.ijcce.org/papers/81-T0023.pdf**

- Cervantes, Humberto. (2010). "Arquitectura de Software". Un vistazo al ADN de la industria de Software en México. Editorial Software Gurú. México DF. Número. 27. Año 2010, pág.32. ISSN 1870-0888.
- CMMI-DEV, V1.3 (2010). "Mejora de los procesos para el desarrollo de mejores productos y servicios". CMMI® para Desarrollo, Versión 1.3, Equipo del Producto CMMI. Revisado el día 23 de agosto de 2014.
- Fernández, Oscar M., García, Delba & Beltrán, Alfa (1995). "Un Enfoque Actual Sobre la Calidad del Software" .ACIMED. Revisado el 22 de agosto de 2014. [En línea] **http://eprints.rclis.org/5424/1/aci05395.htm**
- Gamma et al., (1995) "Design Patterns. Elements of Reusable Object-Oriented Software". Editorial: Addison-Wesley. 1995. ISBN: 0-201-63361.Westford, Massachusetts, USA.
- Hadoop. (2009). "Welcome to Apache™ Hadoop®!". Revisado el 22 de agosto de 2014. [En línea] **http://hadoop.apache.org/**
- Haller, Stephan. (2010). "The Things in the Internet of Things". SAP Research Center Zurich. Internet of. Things Conference 2010, Tokyo, Japan. Revisado el 19 de agosto de 2014. [En línea] **http://www.iot-a.eu/public/news/resources/TheThingsintheInternetofThings_SH.pdf**.
- IHS. (2006). "Las Redes de la Próxima Generación Comienzan a Transformar las Comunicaciones". IHS Company. Revisado 19 de agosto de 2014. [En línea] https://www.ihs.com/expertise/thought-leadership.html.
- Jiménez, Luis. (2012). "El Reto de la Calidad en el Desarrollo de Software (II)". El Blog de Desarrollador de IBM Rational España. Revisado el 22 de agosto de 2014. [En línea] **https://www.ibm.com/developerworks/community/blogs/rationalspain/entry/el_reto_de_la_calidad_en_el_desarrollo_de_software_ii4?lang=en**.
- Kendall, E., Keneth y Kendall, E., Julie (2005) "Análisis y diseño de sistemas" Sexta Edición, Editorial: Pearson Educacion, Mexico 2005 ISBN: 970-26-0577-6. Área: Computación.
- Martin-Avil, Toni. (2010). "Certificaciones y Normativas de Calidad en Software". Calidad en Software. Revisado el 22 de agosto de 2014. [En línea] **http://ocw.unican.es/ensenanzas-tecnicas/ingenieria-del-software-i/materiales-de-clase-1/is1_t06_Patrones.pdf**
- Martínez, Guillermo M. (2014). "Usabilidad y Accesibilidad en Web". Revisado 19 de Agosto de 2014. [En línea] **http://www.semac.org.mx/archivos/6-11.pdf**.
- Mora, Juan. (2011). "Arquitectura de software para aplicaciones Web". Tesis de Maestría en Ciencias de la Computación. Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), Unidad Zacatenco, Departamento de Computación. México DF. Revisado el 15 de agosto de 2014. [En línea] **http://delta.cs.cinvestav.mx/~pmalvarez/tesis-tahuiton.pdf**
- Pérez, Sergio L. (2014). "Interfaces de usuario [Interacción humano-computadora]". UAM Cuajimalpa. México, DF. Revisado el 11 de agosto de 2014. [En línea] **http://computacion.cs.cinvestav.mx/~sperez/cursos/ui/Interaccion.pdf**.
- Palacio, Juan & Ruata, Claudia. (2011)."Scrum Manager – Gestión de Proyectos". Licencia Creative Communs.
- Quiroz, Francisco J., (2005). "Sociedad de la información y del conocimiento". Boletín de los Sistemas Nacionales Estadístico y de Información Geográfica. Vol. 1 Numero 1, Mayo – Agosto 2005.

Revisado el 19 de agosto de 2014. En línea: **http://www.inegi.gob.mx/prod_serv/contenidos/espanol/bvinegi/productos/integracion/especiales/BoletinSNEIG/2005/bolsneig1.pdf**

- Restrepo, Ana M. (2009). "Visión General de las Redes de Próxima Generación (NGN)". Universidad Pontificia Bolivariana, Escuela de Ingenierías, Facultad de Ingeniería Eléctrica y Electrónica. Medellín, Colombia, 2009. Revisado 19 de agosto de 2014. [En línea] **http://repository.upb.edu.co:8080/jspui/bitstream/123456789/173/1/TRABAJO%20DE%20GRADO.pdf**.
- Román, Felipe. (2011). "Extracción de Información Basada en Técnicas de Alineamiento de Ontologías". Tesis de Maestría en Ciencias en Ciencias de la Computación. Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET). Cuernavaca Morelos a 28 de Octubre de 2011.
- Somerville, Ian. (2005). "Ingeniería del Software". Editorial: Pearson Educación SA. Séptima Edición, ISBN: 84-7829-074-5.
- Torres, Cesar et al. (2010). "Guía de Arquitectura N-Capas Orientada al Dominio con .Net 4.0". Editorial: Krasis Consulting S.L. España Marzo de 2010. ISBN: 978-84-936696-3-8
- Toro Alonso y Cardona, Lorena (2010). "Estado del Arte de la Ingeniería del Software en el Ámbito Nacional e Internacional de Acuerdo a Organizaciones que Tratan la Disciplina". Universidad Católica Popular del Risaralda, Facultad de Ciencias Básicas e Ingenierías, Ingeniería de Sistemas y Telecomunicaciones. Pereira, Colombia. Revisado 13 de agosto de 2014. [En línea] **http://ribuc.ucp.edu.co:8080/jspui/bitstream/handle/10785/1507/CDMIST18.pdf?sequence=1**
- UIT-T Y. (2014). "ITU-T's Definition of NGN". UNION INTERNACIONAL DE TELECOMUNICACIONES. Revisado 19 de agosto de 2014. [En línea] **http://www.itu.int/en/ITU-T/gsi/ngn/Pages/definition.aspx**.
- Ventura, Teresa & Peñaloza, Marcela. (2006). "MoProSoft: Modelo de Procesos de Software Hecho en México". Entérate en línea, Internet Cómputo y Telecomunicaciones. Año 5, Número 47, Marzo de 2006. Revisado el 22 de agosto de 2014. [En línea] **http://www.enterate.unam.mx/Articulos/2006/marzo/moprosoft.htm**.
- Zapata, Miguel. (2012). "Calidad en entornos ubicuos de aprendizaje". Universidad de Alcalá. RED. Revista de Educación a Distancia, Número 31. ISSN 1578-7680. Revisado el 22 de agosto de 2014. En línea: **http://www.um.es/ead/red/31/zapata_ros.pdf**.