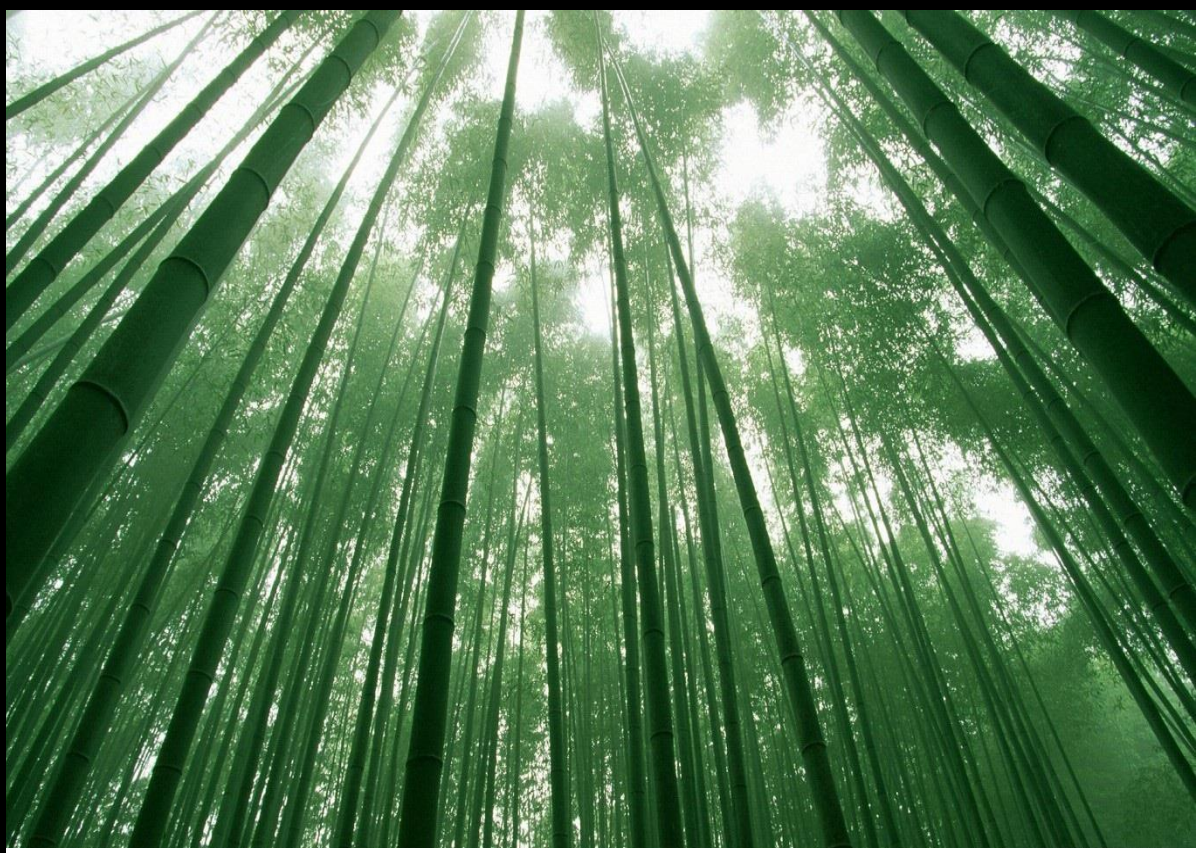
LIBRO BLANCO
DE LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA



Editor
Edgar Serna M.

RedLatinaIS

Instituto Antioqueño de Investigación

Grupo de Investigación en Ciencias Computacionales e Ingeniería de Software

LIBRO BLANCO
DE LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA

EDITOR
Edgar Serna M.

© Instituto Antioqueño de Investigación, IAI
Medellín, Antioquia, Colombia
2013

Serna M., Edgar (Ed.)
Libro Blanco de la Ingeniería de Software en América Latina / Edgar Serna M. --1a ed.
Medellín: Editorial IAI, 2013.
145 p. (Investigación Científica).

Copyright © 2013 Instituto Antioqueño de Investigación (IAI)™

Except where otherwise noted, content in this publication is licensed under the
Creative Commons Attribution, NonCommercial, ShareAlike 3.0 Unported License.
Available at http://creativecommons.org/licenses/by-sa/3.0.

Primera edición, diciembre 2013

Global Publisher: *Instituto Antioqueño de Investigación* (IAI)
Cover Designer: *Instituto Antioqueño de Investigación* (IAI), *Medellín, Antioquia, Colombia.*

Red Latinoamericana en Ingeniería de Software (RedLatinaIS) is trademarks of the *Instituto Antioqueño de Investigación* (IAI). All other trademarks are property of their respective owners.

The information, findings, views, and opinions contained in this publication are responsibility of the authors and do not necessarily reflect the views of *Instituto Antioqueño de Investigación* (IAI), and does not guarantee the accuracy of any information provided herein.

Diseño, edición y publicación: *Instituto Antioqueño de Investigación* (IAI)

Instituto Antioqueño de Investigación
http://fundacioniai.org/index.html
contacto(AT)fundacioniai.org

RedLatinaIS
http://fundacioniai.org/red.html
redlatinais(AT)lacrest.org

© 2013 Editorial IAI
Medellín, Antioquia, Colombia

ISBN: 978-958-46-3302-6

CONTENIDO

PRÓLOGO 6
INTRODUCCIÓN 7
AGRADECIMIENTOS 9

CAPÍTULO I. CONTEXTO DE LA INICIATIVA "LIBRO BLANCO" 10
PRESENTACIÓN 10
1. LATIN AMERICAN CONGRESS ON REQUIREMENTS ENGINEERING & SOFTWARE TESTING 11
1.1 Objetivo general 12
1.2 Objetivos específicos 12
1.3 Premio LACREST 13
1.4 Logo LACREST 13
2. RED LATINOAMERICANA EN INGENIERÍA DE SOFTWARE (RedLatinaIS) 14
2.1 Objetivos 15
2.2 Logo RedLatinaIS 16
3. DEFINICIÓN DE TÉRMINOS 16
3.1 Ingeniería 16
3.2 Ingeniero 17
3.3 Sistema 17
3.4 Software 17
3.5 Informática 17
3.6 Computación 18
3.7 Ciencias Computacionales 18
3.8 Sistemas de Información 19
3.9 Tecnologías de la Información 19
3.10 Ingeniería de Sistemas 19
3.11 Ingeniería Informática 20
3.12 Ingeniería de Software 20
3.13 Desarrollador 21
3.14 Programador 21
REFERENCIAS 22

CAPÍTULO II. LA INGENIERÍA DE SOFTWARE EN LA HISTORIA 23
INTRODUCCIÓN 23
1. HISTORIA DE LA INGENIERÍA DE SOFTWARE 24
1.1 La Ingeniería de Software posterior a las conferencias 26
1.1.1 El desarrollo del concepto *Ciclo de Vida 27*
1.1.2 Evolución del software 29
1.1.3 El desarrollo de la industria del software 32
2. EL DESARROLLO DE SOFTWARE EN AMÉRICA LATINA 36
3. LÍNEA DEL TIEMPO DE LA INGENIERÍA DE SOFTWARE 41
REFERENCIAS 44

CAPÍTULO III. LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA 46
INTRODUCCIÓN 46
1. ARGENTINA, BRASIL, CHILE, PARAGUAY, URUGUAY 47
1.1 Políticas 47
1.1.1 Software público 48
1.2 Formación y capacitación 48
1.1.1 Medios de acceso 48

1.1.2 Valoración general 50
1.3 Programas relacionados con la Ingeniería de Software 51
1.4 Reglamentaciones relacionadas con la Ingeniería de Software 54
2. RESTO DE SURAMÉRICA 55
2.1 VENEZUELA 55
2.1.1 Reglamentaciones relacionadas con la Ingeniería de Software 57
2.2 ECUADOR 58
2.2.1 Reglamentaciones relacionadas con la Ingeniería de Software 60
2.3 PERÚ 60
2.3.1 Reglamentaciones relacionadas con la Ingeniería de Software 63
2.4 BOLIVIA 64
2.4.1 Reglamentaciones relacionadas con la Ingeniería de Software 66
2.5 COLOMBIA 66
2.5.1 Reglamentaciones relacionadas con la Ingeniería de Software 68
3. CENTROAMÉRICA 69
3.1 COSTA RICA 69
3.1.1 Reglamentaciones relacionadas con la Ingeniería de Software 71
3.2 CUBA 71
3.2.1 Reglamentaciones relacionadas con la Ingeniería de Software 73
3.3 GUATEMALA 73
3.3.1 Reglamentaciones relacionadas con la Ingeniería de Software 74
3.4 MÉXICO 75
3.4.1 Reglamentaciones relacionadas con la Ingeniería de Software 76
3.5 PANAMÁ 77
3.5.1 Reglamentaciones relacionadas con la Ingeniería de Software 79
REFERENCIAS 80

CAPÍTULO IV. RETOS Y REALIDADES DE LA INGENIERÍA DE SOFTWARE EN EL SIGLO XXI 81
INTRODUCCIÓN 81
1. VISIÓN GLOBAL DE LA INGENIERÍA DE SOFTWARE 82
2. PANORAMA DE LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA 84
2.1 La formación académica 84
2.2 Atributos de los productos software 85
2.3 Responsabilidad ética y profesional 86
3. PROSPECTIVA DE LA INGENIERÍA DE SOFTWARE 86
3.1 Retos y realidades de la Ingeniería de Software 86
3.1.1 Calidad del software 87
3.1.2 Servicios y modelos de negocio 87
3.1.3 Integración de sistemas y componentes 88
3.1.4 Dispositivos móviles y conectividad 88
3.1.5 Recurso humano competente 88
4. TENDENCIAS 88
4.1 Métodos ágiles e incrementales 88
4.2 Dominio de los productos software 89
4.3 Ingeniería de Requisitos en entornos globales 89
4.4 La prueba del software 89
5. PROSPECTIVA DE LAS NECESIDADES DE LA SOCIEDAD ACTUAL 89
5.1 Gestión del conocimiento 90
REFERENCIAS 94

CAPÍTULO V. LA FORMACIÓN EN INGENIERÍA DE SOFTWARE 95
INTRODUCCIÓN 95
1. GUÍA PARA EL CUERPO DE CONOCIMIENTO SWEBOK 97

2. LA FORMACIÓN EN INGENIERÍA DE SOFTWARE 98
2.1 Pregrado 98
2.2 Posgrado 101
3. OTROS NIVELES DE FORMACIÓN EN INGENIERÍA DE SOFTWARE 103
4. NIVEL DE MADUREZ DE LA INGENIERÍA DE SOFTWARE COMO PROFESIÓN 104
4.1 Elementos que conforman una profesión 104
4.2 Nivel de madurez actual 105
4.2.1 Formación profesional inicial 106
4.2.2 Acreditación 106
4.2.3 Desarrollo de habilidades 106
4.2.4 Certificación 106
4.2.5 Concesión de licencias 107
4.2.6 Desarrollo profesional 108
4.2.7 Sociedades profesionales 108
4.2.8 Código de ética 108
4.3 Evolución de la profesión 109
5. CONCLUSIONES 109
REFERENCIAS 111

CAPÍTULO VI. LA RUTA DE LA PROFESIONALIZACIÓN DE LA INGENIERÍA DE SOFTWARE 113
PRESENTACIÓN 113
INTRODUCCIÓN 115
1. LA INGENIERÍA DE SOFTWARE COMO PROFESIÓN 117
1.1 Desarrollar software es ciencia y es arte 118
1.2 Profesiones y profesionales 119
1.3 Profesionalizar 121
1.4 Responsabilidad profesional de los ingenieros de software 123
1.5 Recomendaciones para alcanzar la profesionalización 125
2. SITUACIÓN ACTUAL DE LA PRÁCTICA PROFESIONAL 125
2.1 Naturaleza del proceso del software 126
2.2 Las prácticas en la Ingeniería de Software 127
3. LA FORMACIÓN DEL INGENIERO DE SOFTWARE 128
3.1 Competencias del ingeniero de software 131
3.2 Roles del ingeniero de software 132
3.3 Prácticas del ingeniero de software 134
REFERENCIAS 135

CAPÍTULO VII. OFERTA DE PROGRAMAS EN DESARROLLO O INGENIERÍA DE SOFTWARE 137
PRESENTACIÓN 137

PRÓLOGO

Cada vez que se escucha acerca de un Libro Blanco, lo primero que se piensa es que se trata de un informe gubernamental acerca de una cuestión de cobertura o de desarrollo en alguna política. Esta es una imagen que nos ha quedado desde que se descubrieron las bondades de este tipo de publicaciones, y desde que la política los empezó a utilizar para publicitar su desempeño. Pero nada que ver con lo que se presenta en este libro, porque es un texto de valor incalculable para quienes estamos embebidos en el mundo de la Ingeniería de Software, o para aquellos que comienzan su formación en esta área.

La calidad de la presentación, lo organizado del contenido y el maravilloso aporte de la literatura que contiene, nos brindan un recorrido por la historia, los sinsabores, los éxitos, los fracasos y el futuro de un área que, como se descubre a medida que se avanza en la lectura, está llamada a ser la de mayor impacto en la Sociedad de la Información y Conocimiento. Porque esta Sociedad es software-dependiente, y sus problemas, en alto grado complicados, necesitan de profesionales bien formados para que les presenten soluciones, pero a un nivel de simplicidad que los remedien sin muchos contratiempos y mediante herramientas atractivas y fáciles de operar para las personas.

El futuro estará siempre gobernado por los desarrollos tecnológicos, los mismos que no pueden funcionar sin el componente del software. Hoy no es posible concebir una nave espacial, un avión, un auto, un teléfono o un reloj, sin un programa que les permita a las personas operarlo y utilizarlo. Esta sociedad se ha acostumbrado a estos desarrollos, y la mayoría no concibe un mundo diferente. Es por esto que se tiene dependencia de la tecnología y, del mismo modo, del software. Este aditamento no se puede separar de su contrapartida, el hardware, porque sería un mundo lleno de máquinas que no se pueden utilizar.

Ahora bien, Latinoamérica es una región que se ha caracterizado por ser consumidora de los desarrollos que provienen de los países industrializados, y eso le ha generado un consumismo que muchas veces supera lo racional. Pero, aunque no se desarrolle tecnología, esta región puede competir con talento humano capacitado. Por tanto, contar con ingenieros de software bien formados, profesionales idóneos, éticos y comprometidos, le brinda una posibilidad única frente a las otras regiones del mundo. Porque si bien en las demás se desarrolla hardware, aquí se puede desarrollar el software para que funcione. Pero esto debe ser un programa de amplia envergadura, en el que se deben comprometer el Estado, la academia y la industria, de tal forma que esos profesionales sean apetecidos en todo el mundo por sus conocimientos y habilidades, y porque desarrollan productos software de alta calidad.

Este libro es un llamado a analizar esa situación, y a re-pensar los programas, mallas y contenidos curriculares porque, como se dará cuenta el lector, a esta región le falta mucho trabajo para ser competitiva en esta área. La diversidad de oferta en programas relacionados con el software, que en muchos casos confunden a los estudiantes sin haberlos tomado, es una de las primeras tareas que se pueden desprender de este texto. Es una labor a corto plazo y con metas inmediatas. En resumen, es un libro con amplias posibilidades de uso, y que viene a cubrir un vacío que se había detectado pero que no se había hecho mayor cosa por llenarlo.

Michael Presmann T. (MSTE)
Redmond, Washington

INTRODUCCIÓN

Mucho se ha dicho y escrito acerca de la Ingeniería de Software. Por desgracia, gran parte de esa producción se ha escrito desde la academia con una perspectiva que no siempre responde a las preocupaciones y necesidades de las partes interesadas e involucradas en los procesos del desarrollo de software. Debido al continuo incremento en la complicación de los problemas que se deben solucionar desde esta área del conocimiento, y con la dependencia cada vez mayor de la sociedad en la tecnología, se necesitan cambiar la visión de esta ingeniería y pensarla como una profesión, y a sus practicantes como profesionales. Ese es en parte el objetivo de este libro, porque sus diferentes capítulos guían al lector por caminos de sensibilización y comprensión del papel de esta área en el siglo XXI.

El contenido de este Libro Blanco es producto del trabajo de un equipo de investigadores preocupados por darle a la Ingeniería de Software el protagonismo que se merece en la sociedad y en los procesos formativos actuales. Un papel activo y cuya necesidad es apremiante de acuerdo con el volumen de problemas que el software debe solucionar actualmente. Cada capítulo presenta un contenido estructurado con objetivos específicos, pero que se enmarcan en una general: describir la situación actual de la Ingeniería de Software en América Latina. Para lograrlo, el texto se distribuye de la siguiente forma:

Capítulo I. *Contexto de la iniciativa Libro Blanco*. Se presenta la historia y el contexto que dieron origen al libro, y se involucran los principios generales de componentes importantes del proceso, como LACREST, RedLatinaIS y el Manifiesto por la profesionalización del desarrollo de software, iniciativas del Instituto Antioqueño de Investigación (IAI), que en conjunto se alinean con el objetivo del libro. También se presenta una definición a los términos comunes para el resto del contenido.

Capítulo II. *La Ingeniería de Software en la Historia*. Un interesante recorrido por los orígenes de esta área del conocimiento, en el que se detallan cuestiones clave para el surgimiento del término, y se describen la evolución del software hasta la actualidad. Además, se hace una detallada descripción de la situación de la industria del software y de la ingeniería misma en Latinoamérica, ofreciendo un punto de vista de las necesidades y prospectivas para su afianciamiento y desarrollo.

Capítulo III. *La Ingeniería de Software en América Latina*. En este capítulo se describe la situación de esta ingeniería en la región, pero desde una perspectiva de las políticas, los programas, la industria y la normativa involucrada. Se hace un recorrido por los países del continente, describiendo cada una de estas perspectivas, y ofreciendo un completo resumen de la situación de la Ingeniería de Software en cada uno de ellos.

Capítulo IV. *Retos y realidades de la Ingeniería de Software en el siglo XXI*. Este capítulo contiene una exposición sustentada en dos ejes fundamentales, a través de los que se aborda la temática propuesta para el desarrollo del contenido, y con base en dos enfoques que, a juicio de los autores, constituyen una perspectiva que le permitirá al lector formarse una idea global acerca de la realidad actual y de los futuros desafíos que debe afrontar la Ingeniería de Software en los albores de este milenio, para luego centrarse en el contexto de América Latina en particular.

Capítulo V. *La Formación en Ingeniería de Software*. La creciente importancia del software en la vida diaria de las personas ha generado en los últimos años una fuerte y creciente demanda mundial de

ingenieros de software cualificados, que ayuden a producir software de calidad, en el plazo y dentro del presupuesto especificado. La Ingeniería de Software es una disciplina relativamente nueva, pero ha madurado rápidamente debido a estas mismas exigencias. Este capítulo presenta los distintos esfuerzos que se han realizado para desarrollar y mejorar la formación en esta área del conocimiento.

Capítulo VI. *La ruta de la profesionalización de la Ingeniería de Software*. El propósito de este capítulo es describir la situación, los objetivos y el alcance de una iniciativa para lograr la profesionalización del ejercicio de los ingenieros de software. Aquí se aclara la comprensión que se tiene, desde la RedLatinaIS, acerca del significado de la Ingeniería de Software, ya que es un término con amplio uso pero con diferentes aceptaciones para cada persona y situación. En pocas ocasiones la visión global de esta ingeniería ha sido completamente positiva, por lo que es un área temática y una profesión que sufre de una sobrecarga de tecnología y de falta de rigurosidad histórica.

Capítulo VII. *Oferta de programas en desarrollo o Ingeniería de Software*. Luego de una de una amplia búsqueda en cada uno de los organismos que centralizan esta información en los países de la región, con el objetivo de determinar mediante el nombre del programa su relación con la Ingeniería de Software, en este capítulo se relaciona el listado de programas que ofrecen las instituciones Latinoamericanas, cuyo nombre permite relacionarlos directamente con el desarrollo profesional de software.

AGRADECIMIENTOS

El Instituto Antioqueño de Investigación (IAI) agradece a todos los investigadores que atendieron el llamado para escribir los capítulos que conforman este libro. Porque gracias al alto nivel de compromiso con el que se dieron a la tarea de investigar, analizar, resumir y redactar el contenido que aquí se presenta, fue posible compilar y editar el texto que presentamos. También agradecemos a la RedLatinaIS, por la difusión y el apoyo a esta tarea, porque sin esa colaboración tampoco habríamos logrado el maravilloso producto que ofrecemos hoy.

Esperamos que este libro se convierta en una guía para pensar en modificar el estado de la Ingeniería de Software en América Latina, porque como se expresa en el prólogo, tenemos mucho trabajo por realizar si queremos ser competitivos globalmente. Gracias a todos los que directa o indirectamente colaboraron para lograr este producto, porque sólo con trabajo en equipo es que se llevan a cabo estos proyectos. Gracias Latinoamérica.

CAPÍTULO I

CONTEXTO DE LA INICIATIVA "LIBRO BLANCO"

Edgar SERNA M.; Alexei SERNA A.
Instituto Antioqueño de Investigación (IAI)
Medellín, Antioquia, Colombia

PRESENTACIÓN

La iniciativa "Libro Blanco" tuvo su origen en el año 2010, en el marco del Congreso Ingeniería 2010 Argentina, que se llevó a cabo en la ciudad de Buenos Aires. En una de las reuniones de trabajo se planteó una discusión acerca de los diferentes enfoques con los que se ofrecen los programas en Ingeniería de Sistemas en el mundo, y de las similitudes que algunos de ellos tienen con los de Ingeniería de Software. En el auditorio se interrogó a los participantes por sus puntos de vista acerca de este tema, y cada uno presentó la visión que desde su conocimiento reflejaba la situación en su país. Entre los 22 participantes, provenientes de Estados Unidos, Colombia, Argentina, Chile, Brasil, España, Inglaterra, Nigeria, Suráfrica, Francia y Arabia, quedó el sin sabor de no encontrar unanimidad de criterios para llegar a conclusiones finales acerca de la situación de estas ingenierías. En definitiva, y para concluir el taller, se tomó la decisión de crear capítulos de trabajo por regiones representativas con el objetivo de investigar acerca de esta cuestión. Se organizaron las comisiones para África, Europa, Norte América, Asía y Latinoamérica.

Esta última comisión quedó conformada por los representantes de Colombia, Argentina, Chile y Brasil, y se encargó de hacer un estudio a la situación de ambas ingenierías en este lado del mundo. Debido a diferentes situaciones no fue posible iniciar el trabajo al año siguiente, y la iniciativa entró en un proceso de replanteamiento. Para finales de 2011 se retomó la idea, pero inmersa ahora en la estructura que se propuso, desde el Instituto Antioqueño de Investigación (IAI), para organizar el Latin American Congress on Requirements Engineering & Software Testing (LACREST). Para este evento, cuya primera versión se llevó a cabo en la ciudad de Medellín, Colombia en 2012, se decidió incluir la conformación de una red que se dedicara a trabajar en el área de la Ingeniería de Software. Este fue el origen de la Red Latinoamericana en Ingeniería de Software (RedLatinaIS), cuyo objetivo es realizar actividades de I+D+i, orientadas al reconocimiento de la Ingeniería de Software como profesión, y a trabajar por la profesionalización y mejoramiento del desarrollo de software.

En LACREST 2012 se concretó la idea de la red, a la vez que se plantearon actividades para atraer a un número mayor de integrantes. Entre las iniciativas se propusieron dos proyectos específicos: 1) redactar el *Manifiesto por la Profesionalización del Desarrollo de Software* y 2) escribir el *Libro Blanco de la Ingeniería de Software en América Latina*, el cual reactivó la idea inicial surgida en Argentina 2010. Luego de LACREST 2012 se conformaron los equipos de trabajo que escribirían cada uno de los capítulos en que se subdivide el presente libro, y se hizo invitación abierta para participar en la redacción del Manifiesto. En el segundo trabajo participaron cerca de 100 académicos, industriales y representantes gubernamentales de la región, que iniciaron el trabajo desde comienzos de 2013. Paralelamente, se conformaron los equipos que redactarían los capítulos del Libro Blanco, que trabajaron en el mismo periodo de tiempo, y que además también hacen parte de la RedLatinaIS.

1. LATIN AMERICAN CONGRESS ON REQUIREMENTS ENGINEERING & SOFTWARE TESTING (LACREST)

LACREST es un Congreso académico-científico que nació como una iniciativa de extensión y difusión del Instituto Antioqueño de Investigación (IAI), para atender cuestiones clave alrededor de la proyección de la Ingeniería de Software como profesión. En la Figura 1 se observa su estructura administrativa. El lema inicial, "*por la profesionalización de la Ingeniería de Software*", se determinó como respaldo al resultado de investigaciones realizadas desde las instituciones participantes, y en apoyo a iniciativas como SWEBOK y otras que patrocinan ACM, IEEE y algunas universidades nacionales e internacionales. El objetivo central del congreso es reunir a un grupo de investigadores, con trabajo multidisciplinar, que propenda por estos intereses.

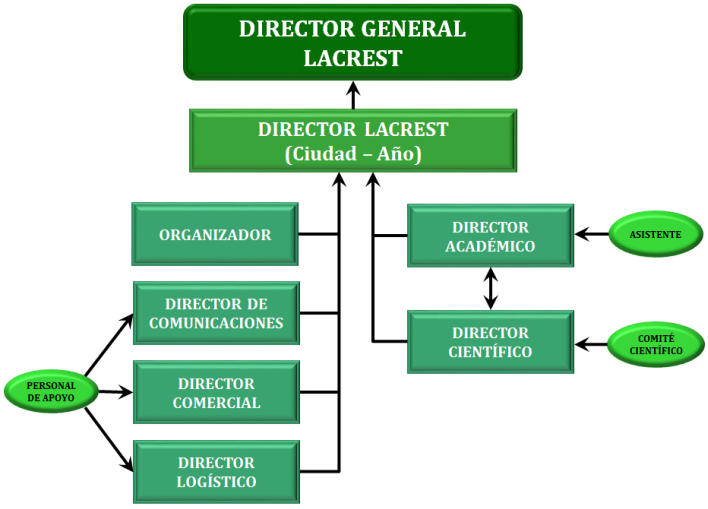


Figura 1. Estructura administrativa del Congreso LACREST

Los organizadores de cada año son las Instituciones de Educación Superior y empresas del sector del software con amplio reconocimiento, que comparten la necesidad de trabajar en las áreas de interés del congreso. El objetivo es estructurar y realizar un evento de carácter ciudad, que reúna a la región y a la nación alrededor de una causa exclusivamente académico-científica, e irrigar los resultados que se obtengan en la academia, la industria, el Estado y la sociedad en general. Para la realización del evento se busca el apoyo de organizaciones estatales del orden regional y nacional, quienes asumen al congreso como un evento de trascendencia, y le dan la importancia que se merece por su impacto cultural, académico, científico, social y turístico, y para el progreso, el desarrollo y la imagen de cada ciudad y país. También se busca el apoyo incondicional de empresas nacionales e internacionales relacionadas con las temáticas que se aborda, y otras que, sin estar directamente relacionadas con el software, valoran el esfuerzo, la dedicación y el compromiso del grupo humano que integra los diferentes comités.

Como evento académico-científico, LACREST es un Congreso que busca crear comunidad alrededor del lema establecido. Se parte de la idea de que en Latinoamérica se cuenta con una industria del software reconocida internacionalmente por su calidad, profesionalismo y responsabilidad, y una academia que investiga esta temática con rigor científico y proyección social y global. Cualidades que se deben valorar y exaltar y, en la medida que se realice el diálogo internacional para conocer los procesos que otros países e investigadores están llevando a cabo, se

podrán estrechar lazos científicos de cooperación y de investigación para proyectar los adelantos. Los beneficios para los diferentes actores que participan en la organización anual del evento son:

- *La academia y los investigadores* podrán lograr acuerdos de integración y de cofinanciación para estructurar y ejecutar proyectos interdisciplinares con pares en diferentes países.

- *La industria del software* re-conocerá el trabajo que se realiza en el mundo en su campo, y podrá fortalecer sus procesos con ideas frescas, además, podrá concretar negocios que le permitan expandirse nacional e internacional para fortalecer su modelo de negocio.

- *La ciudad y el país* se darán a conocer como un espacio donde la ciencia se dialoga y se fortalece desde la comunidad Universidad-Empresa-Estado, de forma transparente y sin interese particulares.

- Lo más importante del evento es el *beneficio social*, porque el sector del turismo logra posicionamiento, porque se mostrará el progreso y el desarrollo físico y humano de cada ciudad, y porque se generarán ingresos económicos para diversos sectores productivos.

En este contexto, la Ingeniería de Software es un área importante y actual, porque el software es un producto del intelecto humano que cubre todos los ambientes sociales. Poder contar cada vez con programas de mejor calidad es una necesidad social apremiante, por lo que espacios como el que brinda LACREST son una ventaja competitiva para la región. El congreso representa oportunidades en el área de las temáticas cubiertas para compartir experiencias, discutir avances, establecer alianzas académico-investigativas, intercambiar ideas, visitas científicas, pasantías, proyectos científicos comunes, entre otras. Para los estudiantes y profesores es una oportunidad de discutir sus investigaciones con pares latinoamericanos, y de otras regiones del planeta. Además, se proponen esfuerzos cooperativos y planes para reorganizar y reagrupar actividades futuras de la comunidad latinoamericana, para continuar investigaciones de forma dinámica, y para compartir conocimientos, experiencias y facilidades experimentales.

1.1 Objetivo general

- Realizar un evento académico-científico en torno a la Ingeniería de Requisitos, las Pruebas de Software y otras áreas relacionadas, que sirva de punto de encuentro y de actualización para los actores académicos y la industria del software: 1) Academia: Formación, 2) Empresa: Negocios/asociatividad y 3) Estado: Políticas.

1.2 Objetivos específicos

- Realizar transferencia y gestión de conocimiento a partir de conocer y compartir los procesos académico-investigativos alrededor de las temáticas cubiertas, y hacer visible y proyectar internacionalmente el trabajo de académicos, científicos y empresarios en Ingeniería de Software.

- Propiciar un espacio de acercamiento entre la industria, la universidad y el Estado, con el propósito de establecer estrategias para fortalecer estos tópicos en Latinoamérica. Asimismo, discutir el futuro de la ingeniería de Software, de cara a los planes de desarrollo Nacionales, y Regionales, en los que se reconoce su importancia como factor de progreso y de mejoramiento de la calidad de vida de la sociedad.

- Contribuir a cerrar la brecha entre las necesidades empresariales y los resultados del proceso académico-investigativo, en relación con el capital humano, la formación y el conocimiento, en las temáticas que se cubren en cada versión.

- Darle continuidad a la Red Latinoamericana en Ingeniería de Software (RedLatinaIS), involucrando en la discusión al Estado, a través de los entes responsables de políticas de formación, como agente que proporciona la dirección e infraestructura normativa para establecer y mantener los currículos relacionados con el desarrollo de software, y los principios que rigen el trabajo de la Red.

1.3 Premio LACREST

En cada versión del congreso se entrega el PREMIO LACREST, que nació con el objetivo de reconocer y exaltar la labor de los investigadores y de los empresarios que trabajan en pro de la profesionalización de la Ingeniería de Software, y se entrega cada año durante la realización del congreso. El PREMIO es un estímulo moral y de reconocimiento para aquellas personas cuyo trabajo científico y empresarial hace aportes sustanciales para mejorar la calidad del desarrollo de software, y para buscar de alguna manera su profesionalización. En la fase final de la organización del congreso se abrirá la convocatoria para presentar los candidatos, cuyas hojas de vida serán analizadas por el comité organizador del evento, y la decisión se dará a conocer en la clausura del mismo.

1.4 Logo LACREST

En la Figura 2 se muestra el logo del Congreso LACREST, en el que, en fondo blanco, tres ruedas se entrelazan mediante ejes sólidos conformando una triada, similar al diseño de los sistemas de locomoción de los vehículos interplanetarios. La mecánica de la imagen refleja el principio ingenieril de un sistema en movimiento, y estructurado para superar con éxito las dificultades del entorno. El color blanco del fondo y el verde de las ruedas representan los colores de la bandera antioqueña, símbolo del tesón y de la pujanza de personas que siempre miran hacia adelante. Los ejes son de mármol blanco y gris para representar la perduración en el tiempo del mecanismo. Las ruedas están diseñadas en 3D para representar solidez y resistencia, y los ejes, también en 3D, tienen una ligera separación hacia el centro, para representar la independencia de acción entre los principios bajo los cuales se direcciona LACREST: Ciencia, Investigación, Desarrollo e Innovación.

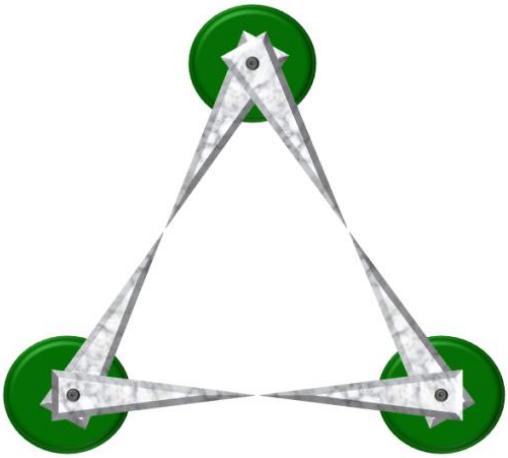


Figura 2. Logo LACREST

2. RED LATINOAMERICANA EN INGENIERÍA DE SOFTWARE (RedLatinaIS)

La RedLatinaIS es un proyecto del Instituto Antioqueño de Investigación (IAI) que se materializó en LACREST 2012. Como parte de la iniciativa del Congreso se decidió darle vida a la red para que se dedicara a trabajar en el área de la Ingeniería de Software, con el objetivo de realizar actividades de I+D+i orientadas al reconocimiento y desarrollo de la Ingeniería de Software como profesión, y a trabajar por la profesionalización y el mejoramiento del desarrollo de software. En la Figura 3 se aprecia su estructura administrativa.

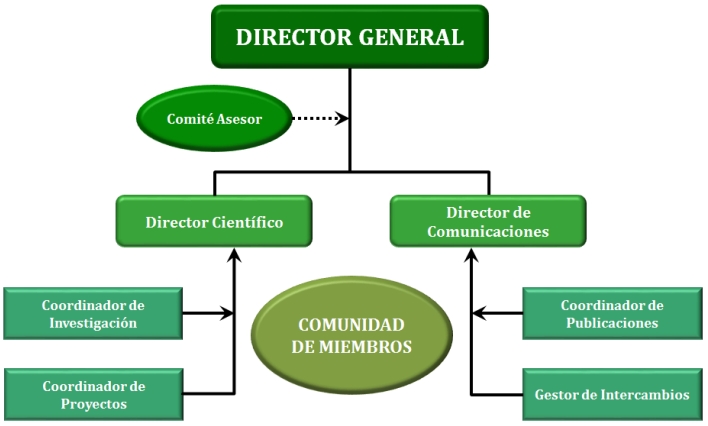


Figura 3. Estructura administrativa de la RedLatinaIS

La Red es una asociación de hecho, entendida como una comunidad, conformada por miembros que voluntariamente se vinculan y colaboran en busca de lograr los objetivos propuestos. Es una entidad sin ánimo de lucro que reúne como *Miembros Plenos* a investigadores, grupos y centros de investigación, y a representantes de la Industria y del Estado de América Latina, y como *Miembros Asociados* a investigadores, instituciones de investigación, desarrollo y difusión de fuera de la región, interesados en trabajar en las áreas y principios que originaron la Red. Se conformó con el propósito de tener duración indefinida, y su ámbito de actuación es América Latina, aunque sus iniciativas y resultados pueden ser de carácter mundial. Tiene domicilio en la Ciudad de Medellín, Antioquia, República de Colombia (http://fundacioniai.org/red.html). Su objeto es contribuir al desarrollo de la Ingeniería de Software en América Latina, a través de las siguientes actividades:

1. Fortalecer la comunidad latinoamericana alrededor de la Ingeniería de Software.

2. Promover y ejecutar investigación en las disciplinas relacionadas y sobre cuestiones prioritarias para el desarrollo de la región.

3. Fomentar, incentivar y orientar la capacitación y actualización permanente de los investigadores y científicos latinoamericanos.

4. Contribuir a la consolidación de principios comunes para la formación e internacionalización de los profesionales en el área de la Ingeniería de Software, aportando con ello a la integración académica entre la región, y de ella con el mundo.

5. Participar y proponer iniciativas de desarrollo e innovación industrial para los productos software, con resultados que impacten el progreso latinoamericano.

La filosofía de la Red es *Ser abierta*, porque se ofrece de forma universal sin ningún tipo de exclusión o discriminación, y porque se informa cotidianamente acerca de su funcionamiento, de sus iniciativas y de sus componentes, ofreciendo la oportunidad para que cualquier persona pueda hacer sus aportes. *Ser libre*, porque la comunidad puede tener sus propias iniciativas sin ningún tipo de restricciones o regulaciones, y disfrutar de las libertades propias de los procesos investigativos, independientemente de su nivel de participación en la Red, y sólo teniendo en cuenta que al accionar a nombre de ella no contradice los términos y condiciones de los principios que contienen sus estatutos. *Ser neutral*, porque es independiente de los contenidos y no los condiciona, por lo que circulan libremente y la comunidad los puede acceder y reproducir independientemente de su nivel de participación. Cuando esos contenidos se incorporan a la Red se hace con el fin de apoyar su difusión, gestionar resultados o simplemente como ejercicio colaborativo, pero en ningún caso con el objetivo de sustituir o bloquear otros contenidos.

2.1 Objetivos

- Generar y promover proyectos de investigación en Ingeniería de Software, que tengan especial importancia para la región, estimulando el trabajo cooperativo entre la comunidad y asegurando la independencia de la investigación científica y de los recursos necesarios.

- Promover la realización de reuniones científicas sobre temáticas de la Ingeniería de Software, y de su desarrollo en la América Latina, y colaborar en la celebración de reuniones profesionales de las diferentes disciplinas relacionadas, y reuniones especializadas de investigación y formación, y otras en las áreas de competencia de esta Ingeniería. En especial, colaborar y promover la realización anual del congreso LACREST.

- Mantener informada a la comunidad acerca de convenios, becas, premios, subsidios y otros incentivos para la investigación individual y grupal en la región.

- Promover y gestionar el intercambio de información entre la comunidad, sobre programas y convocatorias de investigación y de capacitación en cada uno de los países de la región. Para lo que actuará como nodo de intercambio de la información que se genere a este respecto, y que será compartida por y para la comunidad. Esto permitirá el reconocimiento recíproco entre investigadores y científicos de la región con sus pares en el resto del mundo.

- Actuar como mediador para el intercambio de investigadores y científicos entre las instituciones de la región, y sugerir, promover y concretar la cooperación de los mismos en la ejecución de proyectos de interés común.

- Asesorar a la comunidad en la formulación y el desarrollo de programas y proyectos de investigación y de formación, en lo que tiene que ver con los aspectos organizacionales, comunicacionales e institucionales.

- Trabajar por la vinculación de la Ingeniería de Software de América Latina con otros países del mundo, y mantener relaciones internacionales con organizaciones y centros de investigación básica relacionada. Asimismo, mantener una estrecha relación con los organismos latinoamericanos e internacionales, gubernamentales y no-gubernamentales, cuyos objetivos de trabajo sean afines a los de la Red.

- Estimular el análisis a la integración latinoamericana y de desarrollo comparado, en lo que tiene que ver con programas de investigación, formación y capacitación en el ámbito de la Ingeniería de Software, y facilitar las vinculaciones necesarias con los organismos de conducción de integración de América Latina.

2.2 Logo RedLatinaIS

En la Figura 4 se aprecia el logo de la red, en el que, en fondo blanco, dos brazos cubren una espiral que asciende y desciende en una representación de continuidad. Los brazos representan el trabajo en equipo de la comunidad, y la espiral el devenir constante de la producción científica, y su divulgación y apropiación social. En conjunto es un esquema de trabajo armónico, de un grupo de personas que conforman una comunidad unida, comprometida y responsable, con objetivos y metas claras, y que mira hacia adelante con prospectiva y espíritu emprendedor.

Figura 4. Logo de la RedLatinaIS

3. DEFINICIÓN DE TÉRMINOS

Debido a la amplia variedad de conceptos que se involucran en el contexto en el que se suscribe este libro, a continuación se presentan las definiciones de los términos que se utilizan de forma reiterativa en el texto, con el objetivo de dar claridad a su intencionalidad en el resto del documento. Estas definiciones se toman del "Manifiesto por la Profesionalización del Desarrollo de Software" [SER13], otra iniciativa de la RedLatinaIS y el Instituto Antioqueño de Investigación (IAI), en el que se exponen las razones por las que se debería reconocer como profesión a la Ingeniería de Software, y como profesionales a los ingenieros de software, a la vez que se convierte en un aporte para la profesionalización de su práctica representativa: *el desarrollo de software*.

3.1 Ingeniería

Esta área del conocimiento consiste en la aplicación práctica de la ciencia y las matemáticas, para resolver problemas mediante el planteamiento de soluciones innovadoras. Para hacerlo es necesario imaginar, inventar, crear y construir el mundo, que debido a su dinamismo se encuentra en permanente cambio. No basta sólo con presentar soluciones, también es necesario mantenerlas y mejorarlas, por lo que se debe aplicar matemáticas y ciencia en las actividades de diseño y creación. Esta disciplina utiliza el ingenio, combinado con las leyes conocidas de la ciencia, para diseñar y crear dispositivos y sistemas que mejoren la calidad de vida de la sociedad. En este proceso de conversión de los recursos naturales en dispositivos para el uso humano, se aplican diferentes principios científicos, con lo que será posible fabricarlos con pleno conocimiento del diseño, y para pronosticar su comportamiento futuros bajo condiciones específicas de operación.

3.2 Ingeniero

Es el profesional encargado de aplicar la ingeniería. Aunque en muchos contextos el término se ha convertido en algo genérico, que se aplica informalmente a la realización de cualquier actividad relacionada con tecnología, en el sentido tradicional se utiliza para designar a la persona que aplica su formación y capacidades para construir los componentes, físicos o no-físicos, que hacen parte de los diferentes sistemas. Para lograrlo requiere un amplio conocimiento científico y matemático, que aplica y utiliza de forma práctica, ingeniosa, creativa e innovadora, por lo que es un especialista en tecnología que diseña y produce soluciones a las demandas sociales. Su trabajo en investigación y desarrollo tiene un enfoque diferente al que se aplica en ciencia, porque crea un enlace entre los descubrimientos científicos y su aplicación a las necesidades humanas. Los ingenieros poseen algunas características diferenciadoras, porque son constructores, aventureros y solucionadores de problemas; porque buscan formas más rápidas, mejores y menos costosas de ejecutar proyectos; porque utilizan las fuerzas y materiales de la naturaleza para el beneficio de la humanidad; porque ayudan a mejorar la calidad de vida con dispositivos que ahorran trabajo, como motores y computadores, y con tecnologías para cuidar la salud, como riñones artificiales y máquinas corazón-pulmón. Por otro lado, ayudan a satisfacer necesidades sociales más amplias, como vivienda, alimentación, vestuario, transporte, agua, aire y energía. En términos generales, el ingeniero es una combinación de científico, matemático, inventor y planificador de proyectos, que toma el conocimiento de varios campos y lo aplica para resolver problemas, y por esto es creador.

3.3 Sistema

En su definición más amplia, es un conjunto integrado de elementos que cumplen un objetivo determinado, pero desde diferentes disciplinas de la ingeniería se tienen diferentes perspectivas de lo que es un sistema, por ejemplo, para la Ingeniería de Software es un conjunto integrado de programas informáticos, y para la Eléctrica se refiere a los circuitos integrados, o a un conjunto de unidades eléctricas; es decir, su significado depende de la perspectiva y el contexto en el que se aplique. En su definición más interna, es una conjunción de recursos y procesos que operan en conjunto para lograr un propósito común, y por lo tanto satisfacer alguna necesidad. En otras palabras, es un conjunto de componentes interactivos o interdependientes que conforman un todo integrado.

3.4 Software

En términos generales se refiere a la información que se almacena en algún tipo de medio digital. El software es la interfaz y la lógica abstracta que vincula al ser humano con una tecnología tangible y/o otro software, permitiendo una interacción y retroalimentación mutuas. En una visión más técnica, el término se utiliza para describir una colección de programas informáticos, algoritmos, procedimientos y la documentación necesaria que realiza algunas tareas en un sistema, como operar los equipos y dispositivos tecnológicos relacionados, y la gestión de la información para la toma de decisiones.

3.5 Informática

Es un término que presenta diversas interpretaciones y por tanto diferentes significados; por ejemplo, en Europa es común que se utilice para referenciar a las Ciencias Computacionales, y otros escenarios reúnen en un sólo programa a la Ingeniería de Software, a la interacción humano-

computador y al estudio de las tecnologías de la información en las organizaciones. Para este libro se asume como el conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de computadores, por lo que se puede entender como la disciplina encargada del estudio de métodos, procesos, técnicas y desarrollos, y de su utilización en computadores con el objetivo de almacenar, procesar y transmitir información y datos en formato digital. Es decir, se refiere al procesamiento automático de información mediante dispositivos electrónicos y sistemas computacionales.

3.6 Computación

De forma general se define como cualquier tipo de cálculo o uso de tecnología computacional en el procesamiento de información. Para lograrlo aplica un proceso que sigue un modelo bien definido, comprendido y expresado en un algoritmo, un protocolo, una topología de red, u otros componentes necesarios. En la computación se investiga lo que se puede o no hacer computacionalmente, aunque el término no está vinculado a los números, acrónimos, puntuación o sintaxis, por lo que una de las cosas que hace a la computación interesante es que no está del todo claro qué es realmente. Usualmente se acepta que cuando alguien, por ejemplo, organiza el saldo de su cuenta corriente está haciendo cómputo. Se sabe que los entornos *virtuales* en los video-juegos son en realidad simulaciones computacionales, pero también que el universo real efectivamente es un computador en el que la física cuántica se relaciona con la información, y que la materia y la energía son abstracciones construidas a partir de ella. Pero si el universo *sólo* es computación, entonces ¿qué no es computación? Por lo tanto, la computación es una idea en constante cambio, y esta época su significado está en proceso de renegociación. Hace cien años se pensaba que era una operación mental que implicaba números, y como tal sólo la podían realizar las personas; hoy en día, la inmensa mayoría considera que la computación la realizan las máquinas, pero al mismo tiempo, de alguna manera, todavía se asocia con el pensamiento.

3.7 Ciencias Computacionales

Las Ciencias Computacionales no se refieren a la construcción de computadores o a la escritura de programas, ni a las herramientas utilizadas en la computación, se refieren a cómo utilizar estas herramientas y a interpretar lo que se encuentra durante su uso. La solución a muchos de los problemas de estas ciencias ni siquiera requiere el uso de computadores, sólo lápiz y papel, e incluso la mayoría de ellos se ha abordado y solucionado desde décadas antes que se construyeran las máquinas modernas. Las Ciencias Computacionales se refieren al enfoque científico y matemático de las tecnologías de la información y sus aplicaciones, y al software y al hardware subyacente. Un científico computacional se puede especializar en la teoría computacional o en la experimentación y diseño de componentes y nuevas tecnologías. Abarcan un amplio rango de conocimientos, desde los fundamentos teóricos y algorítmicos hasta los desarrollos de vanguardia en robótica, visión por computador, sistemas inteligentes, bioinformática, y otras áreas relacionadas. Los científicos computacionales pueden diseñar e implementar software, crear nuevas formas de utilizar los computadores y desarrollar métodos eficaces para resolver problemas computacionales, y en ocasiones los planes de estudios en estos campos son criticados por no preparar a los estudiantes para un trabajo específico, porque mientras otras disciplinas lo hacen en habilidades para el trabajo inmediato, estas ciencias ofrecen una formación integral de adaptación a las nuevas tecnologías.

En términos generales, estas ciencias se ocupan del estudio sistemático de la viabilidad, la estructura, la expresión y la mecanización de los procesos, metódicos o algoritmos, que subyacen a

la adquisición, representación, procesamiento, almacenamiento, comunicación y acceso a la información, cuando está codificada en bits y bytes en un computador, o transcrita en los genes y en las estructuras proteínicas en una célula humana. La cuestión fundamental subyacente es ¿qué procesos computacionales se pueden automatizar e implementar eficientemente? Para responder esta pregunta, aparentemente simple, los científicos computacionales trabajan en diversas áreas complementarias, como la naturaleza misma de la computación, para establecer cuáles problemas son o no computables; comparan diversos algoritmos, para determinar si ofrecen una solución correcta y eficiente a un problema; diseñan lenguajes de programación, para especificar y expresar algoritmos; diseñan, evalúan y construyen sistemas computacionales, que puedan ejecutar eficientemente esas especificaciones, y aplican los algoritmos al dominio de aplicaciones importantes.

3.8 Sistemas de Información

Esta disciplina hace hincapié en las tecnologías como instrumento para generar, procesar y distribuir información. Los especialistas integran las soluciones tecnológicas y los procesos de negocio para satisfacer las necesidades de información de las empresas, lo que les permite alcanzar sus objetivos de manera eficaz y eficiente. Los profesionales en esta área se ocupan principalmente de la información que los sistemas computacionales proporcionan, para ayudarles a las empresas en la definición y el logro de sus objetivos, y los procesos que pueden implementar o mejorar con el uso de las tecnologías. Deben comprender tanto las técnicas como los factores organizacionales y ser capaces de orientar a la organización a determinar cómo la información y la tecnología pueden habilitar procesos de negocio que le proporcionen ventajas competitivas.

3.9 Tecnologías de la Información

El término se utiliza comúnmente para referir toda una industria, pero actualmente se define como el uso de computadores y software para gestionar información. En algunos entornos se conoce como servicios de gestión de la información, o simplemente como servicios de información, encargados de almacenar, proteger, procesar y transmitir la información, y recuperarla cuando sea necesario. Otros puntos de vista lo etiquetan dentro de dos significados: 1) en un sentido amplio, se utiliza para referir toda la informática y 2) en el ámbito académico, se refiere a los programas de pregrado que preparan a los estudiantes para satisfacer las necesidades tecnológicas computacionales de las organizaciones.

En un sentido más técnico, se define como la rama de la ingeniería que se ocupa del uso de los computadores y las telecomunicaciones para almacenar, recuperar y transmitir información. Sus principales campos son la adquisición, procesamiento, almacenamiento y difusión de la información verbal, de imágenes, de texto y numérica, mediante una combinación micro-electrónica basada en la informática y las telecomunicaciones. Algunos de sus campos nuevos y emergentes son: la siguiente generación de las tecnologías web, la bioinformática, *cloud computing*, los sistemas globales de información y las bases de conocimiento a gran escala. Sus avances se deben principalmente al desarrollo de las Ciencias Computacionales.

3.10 Ingeniería de Sistemas

Es un área del conocimiento que en la mayoría de países latinoamericanos no tiene una definición de consenso, aunque tradicionalmente se define como un campo multidisciplinar para construir grandes cosas complejas, y que para lo que aplica métodos ingenieriles. Es un campo inter y multi-

disciplinar de la ingeniería que se centra en cómo diseñar y gestionar los ciclos de vida de los proyectos ingenieriles, y se ocupa de los procesos de trabajo y de las herramientas para gestionar los riesgos en este tipo de proyectos; muchas veces se confunde con disciplinas técnicas y centradas en lo humano, como la ingeniería de control, la ingeniería industrial, los estudios organizacionales y la gestión de proyectos. Cuestiones como la logística, la coordinación de los diferentes equipos y el control automático de las máquinas se hacen más difíciles cuando se trata de proyectos grandes y complejos, por lo que esta ingeniería integra varias disciplinas y grupos de especialistas en un esfuerzo de equipo para conformar un proceso de desarrollo estructurado, el cual va desde el concepto de producción hasta el de operación. Por eso debe considerar al negocio y a las necesidades técnicas de todos los clientes, con el objetivo de ofrecer un producto de calidad que satisfaga las necesidades de los usuarios.

La ingeniería de sistemas es un área que se ha desarrollado desde hace relativamente poco tiempo. Su principal objetivo es asumir los sistemas de ingeniería como conjunto y no sólo a un componente en particular. Su objetivo surge del hecho de que casi todo lo que nos rodea y que utilizamos a diario es un sistema relativamente complejo, conformado por diferentes componentes mecánicos, electrónicos, hardware, software, y posiblemente otros tantos. Abarca el diseño de modelos matemáticos y el análisis de sistemas, centrándose especialmente en cómo encajan entre sí los distintos componentes, y en garantizar un diseño en el que todos interactúen de manera eficiente y eficaz. Muchas universidades de la región orientan este objeto de formación al desarrollo de proyectos software, aunque sus planes curriculares no logren ese cubrimiento.

3.11 Ingeniería Informática

Es un campo del conocimiento que tiene que ver con el diseño y la construcción de equipos y sistemas basados en computadores. Se trata del estudio de hardware, software, comunicaciones, y la interacción entre ellos. Sus planes de estudio se centran en las teorías, los principios y las prácticas de la ingeniería eléctrica tradicional y de las matemáticas, que luego aplica para resolver los problemas de diseño de computadores, y de los dispositivos basados en ellos. En esta ingeniería se estudia el diseño de sistemas hardware digitales, incluyendo los de comunicaciones, los computadores y los dispositivos que contienen procesadores, centrándose en los dispositivos digitales y sus interfaces con los usuarios y otros dispositivos.

Los ingenieros informáticos analizan y diseñan el hardware, el software y los sistemas operativos para los sistemas informáticos, y para esto deben combinar los campos de las Ciencias Computacionales y la Ingeniería Eléctrica. A menudo se confunde con las Ciencias Computacionales, pero los ingenieros informáticos también se forman en el diseño de software y en su integración con el hardware, por lo que deben aplicar algoritmos y principios de diseño digital para diseñar, construir y probar los componentes, que se utilizan para procesar, comunicar y almacenar la información, normalmente integrada en grandes sistemas de ingeniería y en ambientes de redes distribuidas. Sus áreas de aplicación incluyen a las comunicaciones, la automatización, la robótica, la potencia, la energía, la salud, los negocios, la seguridad, el entretenimiento, entre otros.

3.12 Ingeniería de Software

Es la disciplina ingenieril que proporciona y aplica los métodos y herramientas necesarios para construir software de calidad, ajustado al presupuesto, en un plazo determinado y en un contexto de constante cambio de requisitos. Es la aplicación de un enfoque sistemático, disciplinado y

cuantificable al desarrollo, operación y mantenimiento de software, es decir, es la *aplicación de ingeniería al software*. El software en la mayoría de sistemas es, en términos de costo y complejidad, el componente predominante, por tanto, las buenas prácticas de esta ingeniería y las herramientas pueden hacer una diferencia sustancial, incluso en la medida en que son la fuerza impulsora del éxito del proyecto. Se diferencia de las Ciencias Computacionales en que éstas tienen que ver con el desarrollo de aplicaciones científicas a gran escala, mientras que la Ingeniería de Software abarca no sólo los aspectos técnicos del desarrollo, sino también las cuestiones de gestión, como la dirección de equipos de diseño, de desarrollo, de pruebas, de presupuestos y de mantenimiento.

Actualmente, esta ingeniería ha evolucionado como respuesta a los factores del creciente impacto y costo de los grandes sistemas software, y a su importancia cada vez mayor en aplicaciones de seguridad crítica. Tiene un carácter diferente al de otras disciplinas ingenieriles, debido a la naturaleza intangible de sus productos y a la discontinuidad operativa. Es un área del conocimiento que integra en las prácticas ingenieriles principios matemáticos y de las Ciencias Computacionales, para desarrollar el software que requieren los artefactos tangibles y físicos. En términos filosóficos, la Ingeniería de Software se encarga de la aplicación de diferentes estrategias y disciplinas para mejorar la capacidad de los seres humanos para enfrentar los retos de la actual Sociedad de la Información y el Conocimiento.

3.13 Desarrollador

Los desarrolladores escriben *código de calidad*. Hacerlo limpio, claro, bien factorizado y libre de errores son cuestiones importantes que tienen en cuenta; además, conocen el significado de *buen código* dentro de un dominio. Tienen adecuados conocimientos en matemáticas, conocen de buenas soluciones para los problemas, poseen amplios conocimientos en algoritmia, buenas habilidades en su área de experticia y las relacionadas, y debido a que su trabajo se desenvuelve al interior de equipos multi-disciplinares poseen buena capacidad de comunicación verbal y escrita, y de interacción con otras personas. Son profesionales que contribuyen de diversas maneras para que el producto software tenga éxito. Su trabajo consiste en aplicar principios científicos e ingenieriles para comprender, abstraer y modelar un problema, que se puede resolver mediante un programa informático, para luego aplicar una metodología con el objetivo de llevar a la práctica la solución creada, tradicionalmente soportada en código de lenguaje de programación. En términos generales son responsables de aplicar procesos de calidad en todo el ciclo de vida del software, desde el problema hasta la implementación y el mantenimiento de la solución.

3.14 Programador

Los programadores escriben *buen código*. Hacerlo bien y limpio es un factor importante, pero a menudo tienen prioridad otras cuestiones. Las habilidades matemáticas son opcionales, aunque poseerlas les ayuda a ser conscientes de los problemas comunes y de las soluciones relacionadas con el dominio, pero las habilidades de comunicación e interacción social son primordiales. Son generalistas sin un tipo de especialización verdaderamente profundo. Son capaces de encontrar caminos en torno a los problemas, y de conectar diversos componentes para cumplir con una serie de requisitos. Su trabajo consiste en aplicar el conocimiento que poseen de un lenguaje de programación para escribir código de forma *eficiente* y *eficaz*, a la vez que respetan los conceptos de calidad y de seguridad, y responden al diseño que se ha construido desde la Ingeniería de Software.

REFERENCIAS

[SER13] Serna, M.E. (Ed.) (2013). Manifiesto por la Profesionalización de la Ingeniería de Software. Instituto Antioqueño de Investigación (IAI). Red Latinoamericana en Ingeniería de Software (RedLatinaIS).

CAPÍTULO II

LA INGENIERÍA DE SOFTWARE EN LA HISTORIA

Brigitte ORTIZ L. [1]; Jorge BEDOYA R. [2]; July CORREA S. [3]; Guillermo BONILLA M. [4];
Walter ARBOLEDA M. [5]; Eugenia MEJÍA de R. [6]; Paula TAMAYO O. [7];
María MORALES M. [8]; Raquel MARTÍNEZ M. [9]
[1, 3, 4, 6, 7, 8, 9] *Institución Universitaria de Envigado (IUE)*
Envigado, Antioquia, Colombia
[2] *Instituto Tecnológico Metropolitano (ITM)*
[5] *Universidad EAFIT*
Medellín, Antioquia, Colombia

INTRODUCCIÓN

De acuerdo con el Diccionario de Computación de IEEE [IEE90], el término software se define de tres maneras diferentes: 1) instrucciones de computador que, cuando se ejecutan, proporcionan la función y el comportamiento deseado, 2) estructuras de datos que les facilitan a los programas manipular adecuadamente la información y 3) documentos que describen la operación y el uso de los programas. En este mismo diccionario se define Ingeniería de Software como *la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software*; además, es una área de las Ciencias Computacionales que se encarga de la creación del software que las personas utilizan en escenarios como el comercial, el militar, la salud, el bancario, los negocios, el académico, el científico, entre otros.

En los años 50, la Ingeniería de Software era un concepto que no tenía una amplia aceptación, y no tenía el apoyo ni los seguidores suficientes de parte de la academia y los gobiernos, lo que generaba problemas permanentes por fallas y retrasos en los diferentes proyectos software. Debido a estos inconvenientes, la comunidad, cuyo trabajo tenía alguna relación con el software, decide reunirse en Alemania e Italia con los objetivos de marcar un rumbo al concepto de Ingeniería de Software, definir grupos de trabajo y formalizar el quehacer de los ingenieros de software. Antes de estas reuniones esta ingeniería no era aceptada como profesión, porque quienes la practicaban no tenían la formación suficiente, y la práctica generalizada era la de programar; además, la academia no ofrecía estudios formales, lo que en conjunto incrementaba el número de fallas, detectadas o no. Esta situación proporcionó el contexto para que aparecieran organizaciones y estándares orientados a mitigar en cierto grado lo que se llamó la *crisis del software*, y por otro lado a hacer de esta ingeniería una disciplina seria y orientada a la creación de productos de alta calidad. De este modo, y luego de la segunda guerra mundial, el desarrollo de software evolucionó, hasta convertirse en una actividad que se ocupa de construir software y de maximizar su calidad.

En las décadas de 1950 y 1960, los programas de computador eran simples, no eran portables, generalmente los desarrollaba una sola persona, la prioridad no era la documentación y la movilidad laboral era reducida. Para inicios de la década de 1970, la complejidad de los requisitos de usuario, personales o empresariales, se incrementó sustancialmente, por lo que construir software ya no era tan simple, porque implicaba traducir los requisitos en software, mediante el diseño y el desarrollo de componentes elaborados.

Por otro lado, la creciente importancia que adquiría el software en la sociedad y la dependencia que la industria comenzaba a tener de él, dieron origen a la disciplina que se debería encargar de construir los principios para mejorar su calidad, es decir, la *Ingeniería de Software*. El surgimiento del término se le atribuye a la conferencia de la OTAN en los años 1967 y 1968, que se organizó precisamente para abordar esa *crisis del software*, los problemas asociados con las fallas en los programas, y su deficiente calidad. Luego de estas conferencias aparecieron algunas definiciones para esta ingeniería:

- La Ingeniería de Software trata del establecimiento de los principios y métodos de la ingeniería, a fin de obtener software de modo rentable, que sea fiable y que trabaje en máquinas reales [BAU69].

- La Ingeniería de Software es la aplicación práctica del conocimiento científico al diseño y construcción de programas de computador, y a la documentación asociada requerida para desarrollarlos, operarlos y mantenerlos. Se conoce también como desarrollo de software o producción de software [BOE76].

- Ingeniería de Software es el estudio de los principios y metodologías para el desarrollo y mantenimiento de sistemas software [ZEL79].

En cuanto a las categorías de software, Pressman [PRE03] describe que en la actualidad existen siete que presentan retos continuos a los ingenieros de software: de sistemas, de aplicación, científico y de ingeniería, empotrado, de línea de productos, aplicaciones web y de Inteligencia Artificial. Según Mahoney [MAH90], para hablar de la historia de la Ingeniería de Software es necesario remontarse a los años 50, cuando se inició la comercialización de los computadores y se empezaron a obtener importantes avances en microprocesadores, memorias y periféricos, y cuando empresas, como IBM, no reconocían a la programación de computadores como un tipo de trabajo específico. De hecho, a los programas se les consideraba *objetos* que producían personas aficionadas, que no eran ni científicos, ni matemáticos, ni ingenieros, que no se entregaban a tiempo y que casi siempre contenían errores graves. Por esto es que hablar de Ingeniería de Software equivale a decir que es la disciplina de la ingeniería cuya meta es el desarrollo costeable de sistemas software de calidad.

Estos aportes permiten definir a esta ingeniería como un área de las Ciencias Computacionales en la que se utilizan diferentes metodologías y técnicas, como la gestión de proyectos, el diseño, el desarrollo, la documentación, las pruebas, el control y la gestión de calidad, y el mantenimiento de sus productos. Pero a pesar de que cada año se realizan conferencias, congresos y encuentros, y de que continuamente se incrementa la publicación de adelantos y progresos de las investigaciones en las diferentes áreas relacionadas con esta ingeniería, sus productos todavía presentan deficiencias [SOU11]. Por otro lado, también es cierto que cada año se incrementa la demanda por estos productos, al mismo tiempo que su complejidad, tamaño y cantidad de funcionalidades, lo que ha hecho que el software tenga mayor injerencia social que técnica, y que la mayoría de personas que lo producen se limiten a cumplir, de cualquier forma, las necesidades de los clientes. Si el objetivo es lograr superar estas dificultades y ofrecerle a la sociedad productos con mayor grado de fiabilidad y calidad, todas esas características y prácticas las deben abordar cada vez más los investigadores e innovadores de la Ingeniería de Software, en un trabajo mancomunado con la industria y el Estado.

1. HISTORIA DE LA INGENIERÍA DE SOFTWARE

La historia propiamente dicha de la Ingeniería de Software comienza alrededor de 1890, cuando la Oficina del Censo de los Estados Unidos utilizó tarjetas perforadas y máquinas tabuladoras, diseñadas por Herman Hollerith, para realizar el censo nacional. La compañía de Hollerith sería la base para el origen de IBM. Pero el inicio formal de esta ingeniería se sitúa en la década de 1950, cuando se crearon los primeros compiladores y lenguajes de programación, como RAL, Autocode, IPL, Flow-Matic, Fortran, Comtran, Lips, Algol 58, Fact, Cobol y RPG. En esa época todos los desarrollos de la Ingeniería de Software se orientaban sólo a la programación, utilizando únicamente la programación estructurada, lo que hacía que los programas fueran extensos en líneas de código, y que el control del software se convirtiera en un verdadero problema.

Los lenguajes de programación y los compiladores fueron las principales herramientas tecnológicas usadas en los años 50, y al no tener una metodología para el ciclo de vida del producto los programadores pasaban inmediatamente a escribir código, sin analizar ni diseñar el sistema en construcción. Estos desarrollos se caracterizaban por el tiempo invertido y por la cantidad de fallas y cambios que se debían realizar a los productos. Esos hechos marcaron el contexto que originó las conferencias de la NATO, en 1968 y 1969.

En el otoño de 1967, el Comité de Ciencia de la OTAN, con sede en Bruselas, estableció un Grupo de Estudio en Ciencias Computacionales para que abordara todo lo relacionado con estas ciencias, y para que al año siguiente lo difundiera en la primera conferencia del organismo en esta área [NAU78]. Esa primera conferencia se realizó en Garmisch-Partenkirchen, Alemania, entre el 7 el 11 de Octubre, y asistieron más de cincuenta personas de once países diferentes, todos profesionales involucrados con el software, y con roles como fabricantes, usuarios, docentes y estudiantes. En los cinco días se discutieron aspectos como la relación entre el software y el hardware, el diseño del software, la producción e implementación del software, la distribución del software, el software como servicio, los problemas en el cumplimiento de cronogramas y especificaciones en grandes proyectos y la formación de ingenieros de software.

Los participantes se reunieron en grupos de trabajo para discutir estos aspectos. Entre los asistentes desde Estados Unidos, como líderes de grupo, se encontraban: A.J. Perlis, del Department of Computer Science de Carnegie-Mellon University; B. Randell, de IBM Corporation; T.J. Watson de Research Center de New York; B. Galler, The University of Michigan Computing Center, North University Building; A. Arbor y D. Gries del Department of Computer Science de Stanford University; D. Babcock de Allen-Babcock Computing Inc., New York; R.S. Barton de Consultant in System Design, Avalon, California; R. Bemer de GE Information Systems Group, Phoenix, Arizona; E.E. David de Bell Telephone Laboratories Inc.; H.R. Gillette de Control Data Corporation, Palo Alto, California; R.M. Graham del Project MAC en el MIT; R.C. Hastings de IBM Corporation; J.A. Harr de Bell Telephone Laboratories Inc.; H.A. Kinslow de Computer Systems Consultant, Ridgefield, Connecticut; K. Kolence de Boole and Babbage Inc., Palo Alto, California; R.M. McClure de Computer Science Center, Institute of Technology de la Southern Methodist University de Dallas; M.D. McIlroy de Bell Telephone Laboratories Inc.; A. Opler de IBM Corporation; D.T. Ross de Electronic Systems Laboratory del MIT; F. Selig de Mobil Research and Development Corporation, Dallas, Texas, y J.W. Smith de Scientific Data Systems, California. El Secretario Científico fue L.K. Flanigan, del Department of Computer and Communication Sciences de The University of Michigan Computing Center, USA, y el Observador fue E.G. Kovach, Director Office of General Scientific Affairs, International Scientific and Technological Affairs, Department of State, Washington DC. Esta muestra de los asistentes al evento es una

demostración de que desde esa época se consideraba a la Ingeniería de Software como un área de trascendencia para el desarrollo de los países, y que la construcción de sus productos se debía considerar con mayor cuidado, dado el impacto que comenzaban a tener en la sociedad. Ejemplo de esto es la preocupación de la academia y del gobierno de los Estados Unidos, que envió como observador a un representante del Departamento de Estado.

La segunda conferencia se realizó en Roma, entre el 27 y el 31 de octubre de 1969, y las temáticas abordadas se centraron en los problemas técnicos de los proyectos de esta ingeniería [MAC01]. En esta versión hubo mayor participación, tanto de personas como de países, y Estados Unidos fue uno de los más activos e interesados. Por Canadá participaron P. Cress de Computing Center, University of Waterloo, y J.N.P. Hume, Department of Computer Science, University of Toronto. Esta participación académica canadiense permite inferir la importancia que entonces tenía la industria del software en el país, lo que se manifestaba en el desinterés de la comunidad científica en contraste con el nivel que se presentaba en Estados Unidos.

1.1 La Ingeniería de Software posterior a las conferencias

A principios de la década de 1970 surgieron empresas dedicadas al desarrollo de software, y éste se empieza a ver como un producto comercial, que se puede vender a usuarios con necesidades específicas. Debido a que no todos los productos no eran fiables, el mantenimiento del software se convierte también en un negocio, generando lo que se conoció como el efecto *iceberg:* además del esfuerzo para desarrollar el software, existe un mayor esfuerzo "oculto" correspondiente al futuro mantenimiento del sistema. Para finales de la década el costo del hardware estaba disminuyendo, mientras que con el software sucedía lo contrario. Los científicos e investigadores le prestan mayor atención y se presentan métodos y metodología para fabricar software, dando origen al surgimiento de la Programación Estructurada, como una tendencia metodológica de la Ingeniería de Software; aunque en términos generales no se puede hablar de una metodología ampliamente aceptada para esta ingeniería, porque sólo se presentan tendencias y paradigmas. Como ejemplos se puede mencionar a METRICA en España, MERISE en Francia y SSADM en Gran Bretaña. Algunos organismos y personas naturales también han difundido sus propuestas, como la de Yourdon, y la llamada Unified Process, de Rumbaugh, Booch y Jacobson.

Los primeros pasos orientados al diseño de software los realizó IBM, con Harlan Mills y Niklaus Wirth, quienes propusieron en los años 70 el Top Down Design, un diseño que permite descomponer la aplicación en una serie de módulos y funcionalidades, partiendo de las características globales y descendiendo a lo detallado y funcional mediante la creación de módulos, sub-programas, funciones y sub-rutinas, pero siempre cobijados bajo la Programación Estructurada. Para 1980 IBM continuó con sus investigaciones en el área de diseño, y con Grady Booch crearon una metodología que le daba otro contexto a esta cuestión involucrando en el diseño los conceptos de objetos y clases. Esta propuesta minimizaba el entendimiento del código y de los programas asociados, lo mismo que la cantidad de líneas en un programa, porque se aplicaba el principio de la reutilización del código.

En 1990, James Rumbaugh, Ivar Jacobson y Grady Booch, crearon en los laboratorios de IBM el Lenguaje de Modelado Unificado (UML por sus siglas en inglés). Este lenguaje surgió de la conjunción entre las propuestas que cada uno había hecho por separado: Jacobson con OOSE, Rumbaugh con OMT y la de Booch. De cada una de estas propuestas se extrajeron las características más sobresalientes, que luego se complementaron con los aportes de los tres investigadores. El UML unificó muchos de los criterios que los investigadores trabajaban individualmente, y logró un

consenso amplio en las diferentes comunidades relacionadas con el desarrollo y la Ingeniería de Software. Para finales de 2005, Jacobson anunció el Essential Unified Process (EssUP), una práctica totalmente orientada al desarrollo de software que integra las buenas prácticas de RUP, de las metodologías agiles y de CMMI, contribuyendo al mejoramiento de la madurez, la calidad y la agilidad en los procesos en el ciclo de vida del software. Varias herramientas actuales integran esta propuesta en sus productos, como Microsoft Visual Studio Team System y Eclipse.

1.1.1 El desarrollo del concepto *Ciclo de Vida*

Según el diccionario de computación de IEEE, el ciclo de vida del software es *el marco de referencia que contiene los procesos, actividades y tareas involucradas en el desarrollo, operación y mantenimiento de un producto software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso* [IEE90]. A continuación se describen algunos modelos de ciclo de vida.

- Para la década de 1970, el investigador de las Ciencias Computacionales Winston Royce planteó lo que se puede considerar como el primer ciclo de vida del software: *ciclo de vida en cascada* [ROY70]. Dicho ciclo sugería un desarrollo secuencial y lineal de actividades, desde el pre-análisis del software hasta la fase de mantenimiento, y se llamó en *cascada* por su analogía con una caída de agua, en la cual la fuerza de la misma es proporcional a qué tanto ha avanzado dentro de la cascada, sugiriendo que cualquier cambio que se deba hacer en las etapas tardías del ciclo, implicaba mayor esfuerzo al tenerse que devolver.

- Frederick Brooks, un ingeniero de software y científico computacional, presentó en 1975 el *ciclo de vida basado en prototipos*. Este ciclo de vida tenía como principal característica la generación de un prototipo en etapas tempranas del desarrollo, con el fin de que el usuario final conozca en todo momento la apariencia y funcionalidad del producto. Es uno de los ciclos de vida denominados *evolutivos*. Este investigador también promulgó una de las frases célebres de la Ingeniería de Software: *añadir personal a un proyecto retrasado, lo retrasará aún más* [BRO75], conocida como la ley de Brooks.

- Posteriormente, en 1984, el investigador y científico computacional Meir Manny Lehman propuso el *ciclo de vida incremental* [LEH84]. Un ciclo evolutivo en el que el software se desarrolla mediante incrementos o versiones intermedias. Es decir, cada versión generada del software es igual a la versión anterior sumada a posibles nuevos requisitos del usuario, que van surgiendo durante el desarrollo del mismo. Era el ciclo de vida ideal para entornos de alta incertidumbre, en los cuales los requisitos no son estables, pero tenía el inconveniente de que los errores capturados en los requisitos en etapas tardías del proceso son graves y difíciles de corregir.

- Barry Boehm, un ingeniero informático estadounidense, desarrolló en 1988 el *ciclo de vida en espiral*. La principal novedad de este ciclo, con relación a los anteriores, es que crea una nueva fase denominada identificación y evaluación de riesgos, en la cual se pueden identificar tres tipos: riesgos de proyecto, riesgos técnicos y riesgos del negocio. Entre sus ventajas se puede mencionar el hecho de que es realista, en el sentido de tener en cuenta el factor de riesgo, que siempre estará presente en cualquier proyecto, que a la vez se convierte en desventaja porque requiere de personal especializado en su manejo.

- El consultor y escritor francés Bertrand Meyer propuso en 1990 el *ciclo de vida de agrupamiento*, o de clúster [MEY90]. Se considera el primer ciclo de vida orientado a objetos, y se basa en la idea

de desarrollar componentes o agregaciones, en el que cada uno tiene un ciclo de vida que progresa a través de las fases de especificación, diseño, realización, validación y generalización del componente.

- Brian Henderson-Sellers y Julian M. Edwards, dos profesores australianos, presentaron en 1990 el *ciclo de vida fuente*, también orientado a objetos. Este modelo maneja el concepto de *piscina,* que es un sitio donde reposan las clases del sistema. Las fases que tiene en cuenta son: planificación del negocio, construcción y entrega. Sus características son: alto grado de iteración y solapamiento de la OO, reutilización, y que es aplicable a nivel de clase individual o de agrupamientos.

- El consultor británico James Martin, presentó en 1991 el *ciclo de vida de desarrollo rápido de aplicaciones* (RAD, por sus siglas en inglés) [MAR91]. Dicho ciclo se considera en cascada de alta velocidad, y promueve el uso de la programación visual y los generadores automáticos de aplicaciones, como Oracle Developer 2000. Dentro de sus ventajas se cuenta el aumento en la productividad de los programadores y la disminución del tiempo de desarrollo. Sus principales inconvenientes son la escasa documentación que genera, lo que complica el posterior mantenimiento, además, debido al solapamiento de las fases de desarrollo se tiene un menor control de las actividades.

- El *ciclo de vida en remolino* fue creado por James Rumbaugh en 1992. Es una versión ampliada del *ciclo de vida en cascada,* en el que en cada fase se lleva a cabo un proceso en forma iterativa, hasta que se alcanza el nivel de detalle previsto. A diferencia del de cascada, este proceso es fractal, es decir, más que lineal, y consiste en un desarrollo multi-cíclico con la forma de un remolino [RUM92].

- En 1995 se desarrolló el estándar ISO 12207, cuyo nombre es "Software Life-Cycle Processes". Este estándar establece que el ciclo de vida del software se compone de 17 procesos: cinco principales, cuatro organizacionales y ocho de apoyo [ISO95]:

  Procesos Principales
  - Adquisición: Se refiere a las actividades que debe realizar el cliente que desea adquirir el software.
  - Suministro: Actividades que debe desarrollar la empresa proveedora del software.
  - Desarrollo: Se compone de análisis, diseño, programación, pruebas e instalación.
  - Operación: Se refiere a la explotación o uso del software.
  - Mantenimiento: Se compone de los cambios que por diversas razones puede sufrir el software en el tiempo.

  Procesos Organizacionales
  - Gestión: Se compone de planificación, control, seguimiento y cierre de proyectos.
  - Infraestructura: Diseño de la arquitectura de cada proceso.
  - Mejora: Contiene las medidas o métricas para controlar y mejorar los procesos.
  - Formación: Trata de la elaboración de material de formación y capacitación del producto.

  Procesos de Apoyo
  - Documentación: Es la elaboración y mantenimiento de la documentación generada en los otros procesos.

- Gestión de la Configuración: Tiene que ver con el denominado control de versiones.
- Aseguramiento de la Calidad: Se refiere al control de la calidad de los productos software y de los procesos involucrados en su desarrollo.
- Verificación: Permite determinar si los requisitos y el software son completos y válidos. Se refiere a revisar si internamente cada módulo del software funciona bien.
- Validación: Es la actividad para determinar si el software cumple con los requisitos previstos y expresados por el usuario.
- Revisión Conjunta: Actividad que permite hacer la evaluación de los productos y su evolución dentro de cada proceso.
- Auditoría: Permite verificar el grado de cumplimiento del contrato de desarrollo de software. Generalmente esta actividad la llevan a cabo empresas externas con el fin de actuar de manera objetiva.
- Resolución de Problemas: Es un proceso "comodín", es decir, incluye todo lo que no cabe en los demás procesos.

1.1.2 Evolución del software

En términos generales se acepta que el primer computador de la era moderna fue el ENIAC, pero el concepto de software fue desarrollado alrededor de 100 años antes de este acontecimiento. Charles Babbage (1791-1871) fue un matemático brillante y uno de los más importantes pensadores de su tiempo y, como hijo de un banquero, tenía los recursos suficientes para dedicarse a su obsesión por los dispositivos mecánicos para el cálculo matemático. Como resultado de su trabajo creó el primer concepto de programación: la Máquina Analítica. En los tiempos de Babbage, las calculadoras mecánicas eran de uso común, pero eran calculadoras, no computadores, es decir, no podían ser programadas. Tampoco lo era su primera concepción, la Máquina Diferencial, que fue diseñada para producir tablas matemáticas, y que se basaba en el principio de que cualquier ecuación diferencial se puede reducir a un conjunto de diferencias entre ciertos números, lo que a su vez podría ser reproducido por medios mecánicos.

La Máquina Analítica era un dispositivo complejo, que contenía decenas de barras y cientos de ruedas, y que tenía un molino, un barril, y un eje de entrada y un eje de salida. Cada uno de estos componentes tiene cierta relación con las partes de un equipo moderno, pero lo más importante es que podía ser programada mediante el uso de tarjetas perforadas, una idea que concibió Jacquard a partir de los postulados de Babbage. La primera mujer programadora en la historia fue Ada, condesa de Lovelace e hija del famoso libertino y poeta inglés Lord Byron. Ada conoció a Babbage en 1833 y quedó fascinada con su obra. En 1843 tradujo del francés un resumen de las ideas de Babbage, que habían sido escritas por Luigi Federico Manabrea, un matemático italiano. A petición de Babbage amplio estas notas, y al final generó un documento tres veces más largo que el original. Sus notas incluyeron una forma para que la Máquina Analítica calculará los números de Bernoulli, lo que se considera como el primer programa del mundo para computador.

El término *software* se utilizó por primera vez en 1958, y es probable que fuera acuñado por el profesor John W. Tukey, de la Universidad de Princeton, en un artículo en American Mathematical Monthly [PET00]. La palabra *computar* se aplicó originalmente a los seres humanos que solucionan problemas matemáticos, y el ENIAC se diseñó para hacerse cargo del trabajo de cientos de *computadores* humanos que trabajaban en las tablas de balística. La mayoría de ellos eran mujeres, reclutadas de entre las mejores y más brillantes graduadas universitarios, mientras que los hombres iban a la guerra. La más famosa e influyente de ellas fue Grace Murray Hopper, una matemática que se unió a la reserva naval de EE.UU. durante la guerra, y que llegó a ser almirante. En 1951 desarrolló

un código de instrucciones para la UNIVAC, e ideó el término *programación automática* para describir su trabajo [CAM96]. Además, utilizó la palabra *compilador* para describir una rutina del programa de decisiones, lo que produce un programa específico para un problema particular [CER99].

Hopper se convirtió en una ferviente defensora del concepto de programación automática, y su trabajo condujo directamente al surgimiento de FORTRAN, el primer lenguaje de programación real del mundo, desarrollado a mediados de los años 50 por el equipo de John Backus en IBM. El primer sistema operativo fue el Michigan Algorithmic Decoder (MAD), desarrollado en la Universidad de Michigan en 1959, y que se basó en el ALGOL 3GL. Fue diseñado para manejar los diversos detalles de funcionamiento de un equipo, que era tedioso para codificar por separado. Pero el concepto de *sistema operativo* todavía era en gran parte desconocido, hasta el desarrollo trascendental del S/360 de IBM, en 1964 [CER99]. Para Pressman [PRE2003], el software ha evolucionado en cuatro bloques de tiempo: 1) los primeros años (1950-1965), 2) la segunda era (1965-1975), 3) la tercera era (1975-1988), 4) la cuarta era (1988-2000), que estuvieron marcados por la aparición de nuevas tecnologías y mejoras, tanto el software como en el hardware, así como cambios positivos en la infraestructura de red. Las características de esta evolución se puede observar en la Tabla 1.

Tabla 1. Etapas de la evolución del software

Etapas	Características
1950-1965	Procesamiento por lotes, sistemas no distribuidos y creación de software a la medida
1965-1975	Programación multiusuario y en tiempo real, y aparición de las bases de datos y de la creación y venta de productos de software
1975-1988	Sistemas distribuidos, incorporación de inteligencia al software, hardware de bajo costo y aumento en el consumo de software
1988-2000	Incremento en la potencia de los equipos, aplicación del paradigma orientado a objetos, aplicación al software de las Redes Neuronales y la Inteligencia Artificial, computación en paralelo y crecimiento de las redes de computadores

En estos procesos evolutivos también aparecieron, en varios países, algunas organizaciones de profesionales dedicadas a trabajar en el área del software. Algunas de ellas se propusieron el objetivo de emitir estándares, reglamentaciones y prácticas, con el objetivo de mejorar los procesos relacionados con el área y el ejercicio de la Ingeniería de Software. Algunas de esas organizaciones son:

- *Institute of Electrical and Electronics Engineers* (IEEE). Creado en 1884 en Nueva York, es la asociación profesional más grande del mundo con orientación al avance de la innovación tecnológica y excelencia en beneficio de la humanidad; sus miembros inspiran una comunidad global a través de publicaciones de amplia citación, conferencias, estándares tecnológicos, y de actividades profesionales y educativas. IEEE Technical Council on Software Engineering (TCSE) es un comité para la investigación, el desarrollo y la transferencia de adelantos en Ingeniería de Software, que además ofrece certificaciones en esta ingeniería como Certified Software Development Associate y Certified Software Development Professional. IEEE, conjuntamente con ACM, diseñaron el estándar internacional ISO/IEC TR 19759:2005, conocido también como Software Engineering Body of Knowledge (SWEBOK), cuyo objetivo es establecer el cuerpo de conocimiento de la Ingeniería de Software, como un paso en búsqueda de su profesionalización. En este trabajo se especifican aspectos relacionados con la ingeniería y sus buenas prácticas, como la ingeniería de requisitos, el diseño, la construcción, las pruebas, el mantenimiento, la gestión de la configuración, la calidad, los métodos, las herramientas y los procesos.

- *Association for Computing Machinery* (ACM). Creada 1947 en Nueva York, es una organización por membresía ampliamente reconocida por los profesionales de la informática, y que se dedica a trabajar por que esta área del conocimiento avance como ciencia y como profesión, permitiendo el desarrollo profesional de sus practicantes, y promoviendo políticas de investigación cuyos productos sean de beneficio para la sociedad. Posee capítulos de profesionales en 56 países y de estudiantes en 38. En la promulgación científica patrocina cerca de 12 conferencias mundiales, y publica y promueve innumerables revistas relacionadas.

- *Software Engineering Institute* (SEI). Creado en el año 1984 y financiado por el Departamento de Defensa de los Estados Unidos y administrado por la Universidad Carnegie Mellon; su objetivo es trabajar en el desarrollo de modelos de evaluación y de mejoramiento de los procesos del desarrollo de software. En 1991 desarrolló el modelo SW-CMM, que evolucionó al modelo CMMI.

- *The Computer Emergency Response Team* (CERT). Fue creado en 1988 en la Universidad Carnegie Mellon, por Defence Advanced Research Projects Agency (DARPA), para garantizar la seguridad de la comunidad de internet, encargándose proactivamente de aspectos que puedan afectar las características de seguridad de la red y de los computadores interconectados.

- *American Institute of Aeronautics and Astronautics* (AIAA). Aunque su propósito inicial fue trabajar por el aseguramiento de la industria aeronáutica, con el tiempo se dieron cuenta que el software permeaba todos sus procesos, y entonces lo consideraron como un producto crítico para sus operaciones. Desde entonces ha estructurado y publicado diferentes estándares en software relacionados para la industria aeronáutica, como:

  - AIAA G-010-1993. Guide for reusable software: Assessment criteria for aerospace applications
  - AIAA G-043-1992. Guide for the preparation of operations concept documents
  - AIAA G-013-1992. Recommended practice for software reliability

- *International Organization for Standadization* (ISO). Es el mayor desarrollador mundial de Normas Internacionales voluntarias, que se orientan a especificar productos, servicios y buenas prácticas, ayudando a que la industria sea más eficiente y eficaz. Todos sus desarrollos se realizan a través de un consenso global, lo que ayuda a eliminar las barreras del comercio internacional. Se fundó en 1947, y desde entonces ha publicado más de 19.500 normas, que abarcan aspectos de la tecnología, de los negocios, la seguridad alimentaria, las ingenierías y la agricultura. Esta organización posee un conjunto de estándares para Ingeniería de Software, en las áreas de gestión de la calidad y de la estandarización de tecnologías de la información, todas relacionados con el desarrollo de software.

- *The Society of Software Engineers* (SSE). Es una organización estudiantil compuesta por ingenieros de software, informáticos y otros estudiantes. Tiene más de cincuenta miembros activos que participan y aportan para que la profesión sea reconocida como tal. La ESS tiene fuertes relaciones con empresas como Microsoft, Google, Apple, Northrup Grumman, Goodrich, IBM y Oracle, y trabaja en estrecha colaboración con las empresas de la industria del software para proporcionar oportunidades profesionales potenciales.

- *The Association of Information Systems* (AIS). Es una sociedad para el avance del conocimiento y la promoción de la excelencia en la práctica y el estudio de los Sistemas de Información. Es una asociación profesional con alto reconocimiento para quienes lideran la investigación, la

formación, la práctica y el estudio de los sistemas de información en todo el mundo. Los objetivos de esta asociación son:

- Promover la AIS como líder mundial por su excelencia en la investigación de sistemas de información, la práctica y la formación
- Posicionar los sistemas de información como una profesión de liderazgo al servicio de la sociedad
- Liderar y promover la excelencia en la formación en sistemas de información
- Liderar y promover la calidad en el software como producto crítico de los sistemas de información
- Generar y mantener una comunidad que proporcione servicios y productos para satisfacer las diversas necesidades de los miembros y las comunidades relacionadas

- *IAENG Society of Software Engineering* (ISSE). Esta sociedad es organizada para los ingenieros y estudiantes de la disciplina "Ingeniería de Software". A través de conferencias y talleres regulares sirve de foro para la creación de redes, el intercambio de información, el intercambio de ideas y la solución a los problemas de la comunidad. Los miembros pueden organizar voluntariamente diferentes actividades, y participar en la presentación de las conferencias y talleres IAENG. Hace parte de International MultiConference of Engineers and Computer Scientists (IMECS), que sirve como plataforma para que los miembros de la comunidad de Ingeniería de Software intercambien ideas y resultados de investigación. ICSE es considerada la más importante conferencia a nivel mundial relacionada con la Ingeniería de Software. En ella se discuten los principales aspectos de la disciplina, y se presentan nuevas propuestas e investigaciones en metodologías, estándares, lenguajes, métricas, calidad y gestión de proyectos software.

1.1.3 El desarrollo de la industria del software

En la década de 1950, Estados Unidos y Canadá realizaron el Semi-Automated Ground Environment (SAGE), un ambicioso proyecto de procesamiento de información para asegurar la defensa aérea de ambos países [BOE06]. En el proyecto trabajaron conjuntamente ingenieros de radares, ingenieros de comunicaciones, ingenieros de computadores y los nacientes ingenieros de software, para desarrollar un sistema que pudiera detectar ataques y prevenir bombardeos de aeronaves enemigas.

Entre 1965 y 1970 la cantidad de palabras usadas en los programas para bombarderos, misiles nucleares, sistemas de radares y satélites era entre 20.000 y 600.000, y uno de los aspectos más delicados eran las fallas en la efectividad del armamento causadas por un software de baja calidad, lo cual impidió el desarrollo constante de estas herramientas [MDO10]. En esa época, la cuestión del software era tan crítica que un estudio realizado por Electronics Industry Association (EIA), para el Departamento de Defensa de los Estados Unidos, acerca del costo de la producción del software y de los problemas asociados a la calidad de los productos, se estimó que de continuar en esa tónica, para el año 2015 se consumiría todo el presupuesto del Departamento de Defensa. Esta situación generó preocupación por la producción de software con altos estándares de calidad.

En Estados Unidos, la industria del software recibió poca atención entre 1950 y 1970, por los pocos ingresos económicos que representaba [CAM95]. Fue sólo después de 1970 que esta industria, gracias a la atención de IBM y a los adelantos en los computadores personales, y los ingresos anuales de esta industria se fueron incrementando progresivamente: 1970 (US$ 1.2 billones), 1979 (US$ 2 billones), 1982 (US$ 10 billones), 1985 (US$ 25 billones), y para 1990 (US$ 100 billones). Esto llamó la atención de los economistas y del gobierno y comenzaron a prestarle mayor atención al

crecimiento de esta rama de la economía. Para el 2010, entre las empresas que obtuvieron mayores ingresos en Estados Unidos, se encuentran las que tienen alguna relación con el software, como se observa en la Tabla 2 [SOF11].

Tabla 2. Empresas con mayores ventas en Estados Unidos en 2010

Ranking 2010	Compañía	Ciudad
1	Microsoft	Redmond
2	IBM	Armonk
3	Oracle	Redwood Shores
4	HP	Palo Alto
5	Symantec	Sunnyvale
6	Activision Blizzard	Santa Monica
7	Lockheed Martin	Bethesda
8	Electronic Arts	Redwood
9	CA technologies	Islandia
10	Adobe	San José
11	EMC	Hopkinton
12	SunGard	Wayne
13	Cisco	San José
14	Autodesk	San Rafael
15	BMC	Houston
16	Take-Two Interactive	Nueva York
17	NCR	Duluth
18	Intuit	Palo Alto
19	Synopsys	Mountain View
20	Citrix	Fort Lauderdale
21	VMWare	Palo Alto
22	Apple	Cupertino
23	SAS Institute	Cary
24	Infor	New York
25	Salesforce.com	San Francisco

A finales del siglo XX, Pressman afirmaba que la industria del software ya era la cuna de la economía del mundo [PRE97]. Paralelamente a este proceso, también se preocupó por aportar a la Ingeniería de Software, sobre todo en la aplicación de estándares y metodologías que ayudaran a minimizar las pérdidas ocasionadas por fallas y retrasos, los cuales desde entonces aún se continúan presentando. Pfleeger establece que los factores que han cambiado el desarrollo del software han sido: la tecnología de objetos, las limitaciones del modelo en cascada, el tiempo de respuesta en el mercado, el cambio de la computación de escritorio a la computación en red, los cambios en la economía, los cambios en la interfaz gráfica y la mejora en la conectividad de las redes [PFL01]. Estas características han llevado a que se incremente la investigación, de desarrollo y de cambio constante alrededor de las metodologías y los estándares que se aplican en la Ingeniería de Software y la industria asociada. Por otro lado, la economía del software ha dependido siempre de la economía del hardware, así en cada nueva generación se construyen nuevos computadores que llegan a un nuevo tipo de usuarios y programadores, lo que cambia directamente la economía y define nuevas necesidades en términos de la percepción de los usuarios y sus ingresos [RAC97].

Por otro lado, el desarrollo de la Ingeniería de Software ha estado acompañado del amplio y rápido crecimiento que han mostrado la tecnología y la industria del software en las últimas cuatro décadas. Todo esto muestra que esta ingeniería no es una disciplina independiente, y que su desarrollo depende de los esfuerzos cooperativos entre el gobierno, las fábricas de software, la

academia y las empresas [YAN06]. Además, en la teoría acerca de proyectos en Ingeniería de Software se recomienda a la industria no asumir que los equipos de trabajo o los colegas tienen las tecnologías y las habilidades que se requieren para llevarlos a cabo, porque en su mayoría quizá trabajen con principios no recientes, y se les debe capacitar, obligándolas a salir de su zona de confort por el bien de todos, de lo contrario se podría generar una nueva crisis del software al interior de las empresas [EBE08].

Por su parte, Shepperd sostiene que la década de 1980 a 1990 estuvo caracterizada por el aprendizaje autónomo de la Ingeniería de Software de forma empírica, lo que marcó un hito en el crecimiento de la industria. Es decir, se tenía interés en esta área y se aprendía de diseño, especificaciones, planes de prueba, manuales de usuarios, pero en su mayoría por auto-aprendizaje. Aunque los tiempos cambian y ahora existen industrias de software que certifican sus ingenieros, también es importante que la academia se actualice en lo que tiene que ver con sus procesos formativos en Ingeniería de Software [SHE03]. Campbell-Kelly esboza que en estos momentos es posible que se esté viviendo una convergencia entre los negocios de información y las empresas de software, en el que se migra a la prestación de servicios para los usuarios, lo que dificulta la distinción entre fábrica de software o empresa prestadora de servicios. Además, en 20 o 30 años esta unión generará un cambio en la sociedad, como el que generó el paso de la máquina de vapor a los autos y los aviones [CAM07].

A este respecto, en la India los jóvenes encontraron afinidad de intereses en el desarrollo de software, lo que hizo que la avalancha por esta área de producción se incrementara tanto que el gobierno se vio obligado a crear entidades especializadas para la capacitación en Ingeniería de Software, a los que llamaron Indian Institutes Tecnology, y que hoy se cuenten por miles y forman una base fuerte para su crecimiento económico y para la disminución de sus niveles de pobreza. Desde principios de los años 90, el mundo ha presenciado el vertiginoso ascenso de la participación de la India en la industria global del software, y las tasas de crecimiento de sus exportaciones han dejado sorprendidos a las industrias encargadas de la producción de tecnologías de la información.

Recientemente, y debido a la falta de ingenieros de software, Estados Unidos quiso tomar como modelo al que la India implementó, y optaron por mandar a muchos de sus profesores al país asiático para que aprendieran de las metodologías de Ingeniería de Software que utilizan. Pero lo que encontraron fue excelentes oportunidades de negocio, por lo que optaron por importar mano de obra especializada, y empresas como Hewlett Packard, IBM y Microsoft tomaron la decisión de montar cedes en este país. Una muestra de la fortaleza que tiene la India en ingeniería de Software es que el 36% de los empleados de NASA, el 34% de Microsoft y el 28% de IBM, son de India.

Gemma Cairó Céspedes, profesora de Economía Mundial de la Universidad de Barcelona, y autora de varios estudios sobre la economía de India, asegura que semejante despegue tecnológico está soportado en razones salariales, porque el sueldo promedio del país es de 225 dólares, en contraste con el salario de los técnicos del resto del mundo, el cual está muy por encima de ese valor. Sin embargo, parece ser que el bajo salario no es lo que propicia la demanda por servicios técnicos en India, sino que está relacionado con un buen sistema educativo. India tiene más de 200 universidades y 1200 institutos dedicados a formar en ciencia y tecnología, un sistema de formación liberal que ha permitido que se impartan cursos de informática y programación en los que los computadores son herramientas integradas en el paisaje escolar.

Es por esto que los programadores y el software indio no han surgido de la nada, ni son cosa de ahora, porque la India cuenta con cerca de 300.000 profesionales calificados trabajando en la industria TIC y afines. Además, se gradúan al año casi 100.000 ingenieros de software, aportando una fuente inagotable de conocimientos para la industria local. Los jóvenes se inclinan por la especialización informática debido a un fuerte nivel de conocimiento previo en matemáticas, idiomas y lógica. En los últimos años, el gobierno ha incrementado notablemente los cupos de admisión en los principales Institutos de Tecnologías de la Información, de Ciencia y de Ingeniería. Además, están tratando de iniciar la formación en informática desde los niveles primarios. Todo esto ha permitido que la India sea un gran proveedor de programadores y de aplicaciones informáticas de calidad y a bajo costo.

A través de los años se han desarrollado recursos, herramientas, principios, metodologías y aplicaciones que han entrado a engrosar el cuerpo de conocimientos de la ingeniería de software. Algunos se han arraigado en la historia de esta disciplina, como la programación estructurada, la programación orientada por objetos, las herramientas CASE, la documentación, los estándares, los servicios web, el lenguaje UML, entre otros. Además, también se han hecho esfuerzos por incorporar la tecnología de los métodos formales al desarrollo de productos software, bajo el argumento de que sí es posible probar formalmente que el producto hace lo que se espera y no hace lo que no se espera, se podría predecir y planificar el software de la misma forma que se hace con los productos de las otras ramas ingenieriles. A continuación se describen algunas herramientas que aportan principios y metodologías a la Ingeniería de Software.

- *MERISE*. La primera versión de esta herramienta fue creada por un equipo universitario de ingenieros en Aix-en-Provence, Francia. El proyecto comenzó en el Centre Technique Informatique del Ministerio de Industria, y tenía como misión cubrir las necesidades de la industria y el Estado. Merise es un método de concepción, desarrollo e implementación de productos informáticos, más específicamente, de sistemas de información. Su premisa se basa en la separación de los datos y los procedimientos, lo que asegura una vida más larga del modelo. La metodología consta de cuatro fases: 1) estudio preliminar, 2) estudio detallado, 3) implementación y 4) realización y puesta en marcha. En la primera fase se realiza la planificación del sistema a desarrollar. La primera tarea es proporcionar un marco de trabajo que permita organizar los recursos, los costos y el tiempo, y la segunda es estimar los recursos requeridos para optimizar el desarrollo y el desempeño del sistema [PER06]. En la fase dos se realizan el análisis y el diseño del sistema. El análisis se lleva a cabo teniendo en cuenta los siguientes objetivos:

  - Identificar las necesidades del cliente
  - Evaluar qué conceptos tiene el cliente sobre el sistema para establecer su viabilidad
  - Realizar un estudio técnico y económico
  - Asignar funciones a los recursos (hardware, software, personas, bases de datos, etc.)
  - Establecer restricciones de presupuesto y planificación temporal

  En el diseño se define el proceso para aplicar ciertas técnicas y principios, con el propósito de definir un dispositivo, un proceso o un sistema, con los suficientes detalles como para permitir su interpretación y realización física. El producto se compone del diseño de la salida, el diseño de archivos y el diseño de interacciones con la base de datos. La tercera fase tiene como objetivo producir una solución eficiente en un lenguaje ejecutable que implemente lo que se definió en la etapa de diseño. En esta fase se selecciona un lenguaje de programación determinado para llevar a cabo la solución. La última fase se compone de dos actividades: transferencia del producto y

evolución. La primera tiene como objetivo instalar el producto en la máquina del cliente, con el fin de hacer las pruebas necesarias y lograr un equilibrio en el funcionamiento del producto. La segunda tiene que ver con el mantenimiento que se le puede hacer al producto, y que puede ser correctivo, perfectivo o evolutivo.

- *METRICA*. Fue creada por un grupo de trabajo constituido por personal de distintos Ministerios y Organismos del gobierno español. METRICA es una guía formal, aunque flexible en su utilización, para planificar, analizar, diseñar, construir e implantar sistemas de información, empleando conceptos y técnicas de Ingeniería de Sistemas de Información y Tecnología de la Información [PER06], y define un marco de trabajo que incluye los siguientes aspectos:

  - Una estructura de proyecto que sirva como guía al equipo humano e involucre a los usuarios en las decisiones críticas que se presenten
  - Un conjunto de productos finales para llevar a cabo
  - Un conjunto de técnicas que permiten llegar a obtener los productos finales
  - Funciones y responsabilidades de cada uno de los miembros del equipo de desarrollo

- *SSADM*. Structured Systems Analysis and Design Methodology, es una metodología británica creada en 1981 por Central Computing and Telecommunications Agency (CCTA) y Learmonth and Burchett Management Systems (LBMS). SSADM proporciona una variedad de procesos que permiten llevar a cabo el análisis y el diseño del software a implementar, pero no cubre las fases que tienen que ver con la planeación estratégica ni con la construcción del código. Sus características principales son:

  - Mayor énfasis en los usuarios, sus requisitos y su participación dentro del proceso de desarrollo
  - Definición previa del proceso de producción
  - Posee tres puntos de vistas diferentes: datos, procesos y eventos
  - Mayor flexibilidad en técnicas y herramientas de implementación

2. EL DESARROLLO DE SOFTWARE EN AMÉRICA LATINA

La necesidad de encontrar niveles de desarrollo adecuados lleva a los países a buscar diferentes fuentes para lograrlo. Entre las opciones que recientemente han identificado las naciones, el software se ubica como unas de las principales, en parte por la dependencia cada vez mayor que la sociedad tiene de estos productos. En dicha tónica, los países de la región centran su atención en esta fuente de desarrollo, porque sus economías están buscando alternativas a las establecidas desde hace tiempo en otras regiones del planeta. Cabe resaltar entonces que la Ingeniería de Software se establece como una actividad que debe ser bien remunerada, a la vez que se debe incrementar la calidad profesional de quienes la desempeñan, lo que redundaría en una industria competitiva. La calidad se obtiene en la medida en que se logra una mejora sustancial en los procesos de producción, mantenimiento y gestión del software. Ahora bien, al centrarse en una región como América Latina, que a lo largo de los años ha logrado incrementar su desarrollo económico gracias a la atención que otras regiones del mundo manifiestan, la Ingeniería de Software ofrece una alternativa sólida de consolidación económica y de desarrollo. Porque esta región comprendió que competir con tecnología es una cuestión ardua y desgastante, entonces mira al desarrollo de software como la opción que le permita posicionarse.

Desde hace varios años, el desarrollo de software en Latinoamérica se observa como una industria creciente con miras al fortalecimiento de la economía de las naciones, y está ligada al proceso evolutivo de la cultura y de las necesidades que se deben tener en cuenta a la hora de usar y aplicar la tecnología. Por esto es importante estandarizar y regular el proceso de desarrollo de software, porque esto permite generar productos más eficientes, seguros y confiables, de acuerdo con las necesidades del cliente. Además, los países de la región podrán ser más competitivos en el mercado global. En la región, el desarrollo de la industria del software se ha logrado mediante la cooperación de diferentes países, por lo que el mercado se incrementa y el beneficio es comunitario. Actualmente, Brasil posee la industria de software más amplia de América Latina, y en Argentina la inversión gubernamental para desarrollar el sector productivo de software va en aumento. Por este mismo camino se encuentra México, que apunta a incrementar la calidad en el desarrollo de software y de los procesos relacionados, con miras a mejorar su competitividad internacional.

Por otro lado, el creciente interés por este sector de la economía ha hecho que varios países latinoamericanos se interesen en las certificaciones internacionales, es el caso de Costa Rica, que ya cuenta con varias empresas certificadas en SW-CMM e ISO 9000-2000, y varios grupos consultores expertos en Ingeniería de Software. Otro país que va en esta tónica es Argentina, que se consolida como uno de los grandes productores y distribuidores a nivel internacional, y que cuenta también con varias certificaciones CMMI, es uno de los países con mayor inversión para el desarrollo del sector, y ha implementado un *Plan estratégico de software* que le permita incrementar su participación en el mercado.

Colombia, a pesar de contar con una creciente industria de software, todavía se encuentra desarticulada, y en cierta medida desorganizada, con respecto a otros países que incursionan en este mercado. Esto se debe a la falta de unión real y efectiva entre los diferentes actores, lo que no ha permitido que el sector se posicione para lograr el beneficio esperado. Además, el país no realiza la inversión suficiente en el desarrollo tecnológico, especialmente en la industria del software, lo que se refleja en que sea el país latinoamericano con la menor inversión en este campo. A esta nación le hace falta mayor inversión y mejores procesos formativos, para lograr el nivel de competencia que necesita, aunque cuenta con varias empresas certificadas ISO y CMMI. Primordialmente, la industria del software en América Latina está ligada sólo a desarrollos a la medida para la mediana y pequeña empresa, y en Colombia, comienzan a cobrar importancia los procesos de control y de mejora de la calidad de los productos de la Ingeniería de Software. El país comienza mirar los mercados internacionales, a vincular empresas nacionales con la industria foránea y a crear *clusters*, como Intersoftware, que asocia a las empresas del sector en varias ciudades del país, con el objetivo de crear músculo que les permita competir como mayor relevancia, pero sobre todo con productos de calidad. Entre los servicios que exporta Colombia se encuentran la consultoría, el desarrollo de software y la Ingeniería de Software, los cuales impulsan el desarrollo de la industria nacional, pero que tiene un problema al momento de competir internacionalmente, el factor bilingüismo, que se convierte en una barrera para que la industria llegue a un mejor posicionamiento.

América Latina se enfrenta al reto de posicionar al desarrollo de software como uno de sus programas bandera con miras a incursionar en nuevos mercados, porque encontró en la Ingeniería de Software una importante herramienta para mejorar sus procesos y productos. Pero se necesitan políticas conjuntas que reúnan a la academia, la industria y el Estado, lo mismo que un sistema de regulación a la profesión. Por otro lado, es necesario incrementar la inversión de los países en TI, con lo que podrán implementar mejores estrategias que impulsen al desarrollo de software como principio del desarrollo económico. Ya se han comenzado procesos en este sentido, como la Red de

Ingeniería de Software para Sistemas No Convencionales de Latinoamérica (RedISLA), que tiene el objetivo de articular la gestión y la difusión del conocimiento en Ingeniería del Software, con énfasis en los Sistemas No Convencionales. Entre estos sistemas se encuentran los sistemas móviles, los sistemas multi-mediales vinculados a la televisión digital, las plataformas virtuales de trabajo colaborativo, los sistemas de explotación de información, los sistemas basados en conocimiento, entre otros. A través del intercambio de información científica y tecnológica, conocimientos, experiencias y soluciones, la red contribuye a mejorar la cadena de valor del desarrollo en la industria del software y fomenta la cooperación institucional entre universidades y empresas de la industria relacionada.

A diferencia de RedISLA, cuya adhesión debe ser institucional, la Red Latinoamericana en Ingeniería de Software (RedLatinaIS) basa la participación en lo personal. La RedLatinaIS es una iniciativa del Instituto Antioqueño de Investigación (IAI), en Medellín, Colombia, que se conformó en 2012 y que se dedica a trabajar en el área de la Ingeniería de Software, con el objetivo de realizar actividades I+D+i orientadas al reconocimiento y al desarrollo de la Ingeniería de Software como profesión, y a trabajar por la profesionalización del desarrollo de software. La Red es una asociación de hecho, entendida como una comunidad conformada por investigadores que voluntariamente se vinculan y colaboran en busca de lograr los objetivos propuestos. Es una entidad sin ánimo de lucro que reúne como Miembros Plenos a investigadores, grupos y centros de investigación, y a representantes de la Industria y del Estado de América Latina, y como Miembros Asociados a investigadores e instituciones de investigación, desarrollo y difusión de fuera de la región. Las actividades planteadas para lograr sus objetivos son:

- Fortalecer la comunidad latinoamericana alrededor de la Ingeniería de Software
- Promover y ejecutar la investigación regional en las disciplinas relacionadas y sobre cuestiones prioritarias para el desarrollo de la región
- Fomentar, incentivar y orientar la capacitación y actualización permanente de los investigadores y científicos latinoamericanos
- Contribuir a la consolidación de principios comunes para la formación e internacionalización de los profesionales en el área de la Ingeniería de Software, aportando con ello a la integración académica entre la región y con el mundo
- Participar y proponer iniciativas de desarrollo e innovación industrial para los productos software, con resultados que impacten el progreso latinoamericano

Actualmente, esta red cuenta con cerca de un centenar de miembros de países Latinoamericanos, Estados Unidos y España, y tiene participación activa en el Latinamerican Congress on Requierements Engineering & Software Testing (LACREST), un evento que promueve y fortalece los principios y objetivos de la Red. Además, cuenta con la Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS) (http://fundacioniai.org/raccis/index.htm) como órgano de difusión para los trabajos científicos de sus miembros y de todos los interesados.

En Argentina, la principal asociación es la Cámara de Empresas de Software y Servicios Informáticos (CESSI), que inició en 1990 como un producto de la fusión de la Cámara de Empresas de Software (CES) y la Cámara Empresaria de Servicios de Computación (CAESCO). CESSI es una organización sin fines de lucro que reúne a las empresas y entidades regionales dedicadas al desarrollo, producción, comercialización e implementación de software, y todas las variantes de servicios en el ámbito de la república. Por otro lado, en el 2007 se creó el Consejo Federal de Entidades Empresariales de la Industria del Software y los Servicios Informáticos, cuya misión es

establecer un ámbito de debate amplio y plural, donde se puedan elaborar definiciones estratégicas, y considerar y consensuar diversas acciones, de manera coordinada y concluyente, además de compartir e intercambiar experiencias y logros regionales que puedan ser adoptados por otros miembros, sumando al desarrollo del conjunto. Argentina cuenta desde el 2004 con la Ley de Promoción de la Industria del Software, cuyo objetivo es mejorar la competitividad de las empresas a través del otorgamiento de beneficios fiscales, estimulando el desarrollo la industria del software y servicios informáticos en el país, incentivando la inversión, fomentando la I+D, mejorando los estándares de calidad de productos y procesos, promoviendo las exportaciones, y contribuyendo al incremento del empleo. A partir de esta ley se creó el Fondo Fiduciario de Promoción de la Industria del Software (FONSOFT), un ente que promueve el fortalecimiento de las actividades de producción de software a nivel nacional.

La Asociación de Empresas Brasileñas para la Tecnología de la Información (Assespro), fundada en 1976, tiene el propósito de defender los intereses de las empresas de informática, fortalecer sus asociados y contribuir a la creación de empleo mediante la ampliación del mercado interno y las exportaciones. Actualmente, es miembro activo del Comité de Área de Tecnología de la Información y del Comité Gestor de Internet del Ministerio de Ciencia y Tecnología, y agrupa más de 1400 empresas. En 1998 Brasil promulgó la Ley sobre la Protección de la Propiedad Intelectual del Software, su comercialización en el país, y otras disposiciones, que ha sido reglamentada en diferentes ocasiones buscando ampliar su aplicación a todos los sectores relacionados con el desarrollo de software.

En Bolivia, la organización más representativa del sector informático y del desarrollo de software es la Cámara Boliviana de Tecnologías de la Información (CBTI), que busca promover el desarrollo de la sociedad y el crecimiento de sus empresas, mediante las Tecnologías de la Información como factor de competitividad regional y global en la Sociedad del Conocimiento. En 1997 se aprueba el Reglamento del Soporte Lógico o Software, y en 2011 se establece el régimen general de telecomunicaciones y Tecnologías de Información y Comunicación, y se promueve y prioriza la utilización del software libre y los estándares abiertos, en el marco de la soberanía y seguridad nacional.

La Asociación Chilena de Empresas de Tecnologías de la Información nació en 1984, con el objetivo de potenciar y contribuir a hacer de la industria nacional TIC una industria de clase mundial, estimulando la absorción de tecnología de punta, tanto en Pymes como en clusters claves para la economía del país y la modernización del Estado. La Asociación Chilena de Distribuidores de Software es una asociación gremial sin fines de lucro, fundada en 1988, y cuyo objetivo principal es promover la protección y el desarrollo de la propiedad intelectual del software en el país, apoyando la modernización de las leyes que protegen la propiedad intelectual mediante la Integración de grupos de trabajos con el gobierno. En 2002 se funda la Asociación Gremial de las empresas chilenas desarrolladoras de software (GECHS), con el objetivo de promover el desarrollo de la industria y los servicios relacionados, buscando que las empresas asociadas logren posicionar sus productos y servicios en el ámbito nacional e internacional.

La Asociación Ecuatoriana de Software (AESOFT) es una organización gremial privada y sin fines de lucro que se creó en 1995 con el objetivo de impulsar a las empresas de la industria de las Tecnologías de Información y Comunicaciones. El software se regula de acuerdo con la Ley sobre la Propiedad Intelectual, y por el Decreto Ejecutivo de 2008 se estableció la política gubernamental de uso de Software Libre en la Administración Publica Central.

En 1982 se creó la Cámara Paraguaya de la Informática y las Comunicaciones (APUDI), que tiene como finalidad primordial lograr un adecuado manejo del material informático y de las telecomunicaciones; obteniendo incentivos fiscales, promoviendo intercambios tecnológicos, impulsando a la capacitación del personal, propulsando a la creación de software nacional, y promoviendo y auspiciando Exposiciones, Congresos, Simposios, Paneles nacionales e internacionales. En 1998 se establece que los programas de computador (software) se protegen en los mismos términos que las obras literarias.

La Cámara Uruguaya de Tecnologías de la Información (CUTI) es la organización que representa a la industria nacional de las Tecnologías de la Información y las Comunicaciones. Fue fundada en 1989 y está integrada por más de 300 empresas. El objetivo principal de la Cámara es impulsar el desarrollo sostenible del sector TIC, dinamizando los mercados, facilitando el crecimiento y la globalización de sus miembros, y poniendo énfasis en el desarrollo de las personas y la responsabilidad social. Desde 2003, el software en Uruguay es protegido jurídicamente bajo las normas de derechos de autor.

En el 2000 se fundó la Asociación Peruana de Productores de Software (APESOFT), con el objetivo principal de brindar soporte integral al desarrollo de la industria del Software en el país, aglutinando esfuerzos empresariales alrededor de la creación de la base Tecnológica y Económica, necesaria y suficiente para obtener la competitividad internacional. Mediante la Ley de Derecho de Autor se protege jurídicamente el software en Perú y se reglamenta desde 1996; en 2003 de dictan medidas para garantizar la legalidad de la adquisición de programas de software en entidades y dependencias del Sector Público.

La Cámara Venezolana de Empresas de Tecnologías de la Información (CAVEDATOS) nació en 1983 como el representante nacional del sector privado de industria y comercio, relacionado con la fabricación e integración de software, hardware y redes, incluyendo consultoría en tecnologías de información, internet y otras áreas complementarias de las comunicaciones y la informática. El objetivo principal es representar, fomentar, desarrollar, defender y proteger al sector privado de las tecnologías de la información. El Centro de Excelencia en Ingeniería del Software (CEISoft) se creó en 2002, con el apoyo de la Corporación Andina de Fomento (CAF), el Instituto Europeo de Software (ESI), la Cámara Venezolana de Tecnologías de la Información (CAVEDATOS), la Universidad de Los Andes (ULA), y la participación activa de un conjunto de empresas e instituciones. Es un programa de apoyo a la Industria Venezolana de Software que desarrolla una infraestructura de servicios orientada a elevar la competitividad del sector mediante el mejoramiento continuo de las prácticas de calidad en la industria. En 2005 se promulgó la Ley de Ciencia, Tecnología e Innovación, que tiene como objeto desarrollar los principios y estrategias para la actividad científica, tecnológicas de innovación y sus aplicaciones, a fin de fomentar la capacidad para la generación, uso y circulación del conocimiento, y de impulsar el desarrollo nacional mediante el apoyo a los programas de desarrollo, fortalecimiento y expansión de la industria venezolana de las Tecnologías de Información, especialmente en lo relativo a hardware y software.

La Federación Colombiana de la Industria del Software (FEDESOFT), surgió en 1999 con la misión de velar por el fortalecimiento del sector a través del desarrollo de políticas que normalicen, defiendan y promuevan los intereses de los industriales del software en Colombia. El origen de la Federación hace parte del proceso evolutivo de dos asociaciones gremiales del sector, INDUSOFT y FEDECOLSOFT. La primera se consolidó en 1987 como vocera de la industria del software ante el gobierno, logrando posicionarse como una de las asociaciones más prometedoras de fin de siglo. En

Colombia, la legislación asimila el software a la escritura de una obra literaria, permitiendo que el código fuente de un programa esté cubierto por la ley de Derechos de Autor. Desde 1989 se reglamenta la Inscripción del Soporte Lógico (software) en el Registro Nacional del Derecho de Autor.

3. LÍNEA DEL TIEMPO DE LA INGENIERÍA DE SOFTWARE

La Ingeniería se define como la aplicación de la ciencia en la conversión óptima de los recursos naturales para uso de la humanidad, y fue definida por Engineers Council for Professional Development (ECPD), en Estados Unidos, como la aplicación creativa de los principios científicos para diseñar o desarrollar estructuras, máquinas, aparatos o procesos de fabricación, que se utilizan por separado o en combinación para la realización de diversas obras; para construir o explotar los recursos con plena conciencia de su diseño; para pronosticar su comportamiento en determinadas condiciones de funcionamiento, y demás aspectos como economía de funcionamiento y seguridad para las personas y la propiedad [EDI09].

Por otro lado, el ingeniero debe ser capaz de identificar y comprender las limitaciones (disponibilidad de recursos materiales, humanos, técnicos y económicos), así como los requisitos (utilidad, seguridad, costo, estética) aplicables al objeto o sistema que pretende diseñar y construir [EDI09]. A partir de ese conjunto de exigencias, y utilizando sus conocimientos de áreas como la física, la química, las matemáticas y la economía, entre otras, además de su experiencia, el ingeniero propone soluciones adecuadas a los problemas planteados. En la mayoría de casos la solución no será única, por lo que es necesario evaluar las diferentes opciones para escoger la óptima [EDI09].

El Renacimiento estuvo fuertemente influenciado por el trabajo de Leonardo Da Vinci, considerado ingeniero, inventor y artista. Se construyeron catedrales y grandes edificios, castillos y fortificaciones, aunque la Ingeniería civil surgió propiamente como disciplina en el siglo XVIII, cuando se fundaron las primeras escuelas profesionales de ingeniería, y sus egresados, los primeros ingenieros civiles, realizaron construcciones como abastecimientos de agua y sistemas de saneamiento, redes ferroviarias y carreteras, y planificaron ciudades [EDI09]. Inglaterra y Escocia fueron la cuna de la ingeniería mecánica, como derivación de las invenciones del ingeniero James Watt y los maquinistas textiles de la Revolución Industrial. El desarrollo de los británicos en la industria de máquinas impulsó el estudio de la ingeniería mecánica, tanto en el Reino Unido como en el extranjero. La Revolución Industrial en Europa occidental dominó la evolución de la ingeniería en el siglo XVIII, y fue significativa la influencia de Thomas Savery, Thomas Newcomen, James Watt, Richard Trevithick, Joseph Whitworth, George Stephenson e Isambard Kingdom Brunel, entre otros. El impacto y el potencial de las actividades realizadas por los ingenieros, y la necesidad de contar con escuelas e institutos específicamente dedicados a esta área del conocimiento, fueron determinantes para que Napoleón accediera a fundar, en 1795, L'École Polytechnique, que se convirtió en la primera escuela de ingeniería en el mundo. Tiempo después, en 1824, se fundó la primera escuela de ingeniería en Estados Unidos, The Rensselaer Polytechnic Institute.

Hasta finales del siglo XIX la ingeniería era sólo civil o militar, sin embargo, en 1880 se fundó la Sociedad Estadounidense de Ingenieros Mecánicos, cuatro años más tarde la Sociedad Estadounidense de Ingenieros Eléctricos, en 1908 se creó el Instituto Estadounidense de Ingenieros Químicos, y en 1945 se fundó el Instituto Estadounidense de Ingenieros Industriales. La formalización de las carreras de ingeniería, así como la creación de nuevas escuelas, centros de investigación, empresas y sociedades, sirvieron de motor para continuar descubriendo aplicaciones

de la ciencia y para lograr mejoras para la humanidad. Los conocimientos en electricidad, logrados a partir de la célula original de Alessandro Volta, de los experimentos de Michael Faraday y otros, culminaron con el desarrollo de la dínamo Gramme y del motor eléctrico, que fueron las bases conceptuales para el surgimiento de las ingenierías eléctrica y electrónica. En electrónica se destacaron los aportes de los científicos James Clerk Maxwell y Heinrich Hertz, a fines del siglo XIX, y de los científicos Lee De Forest, con la invención del tubo de vacío, y William Shockeley con el transistor, en el siglo XX.

Posteriormente, los desarrollos a mediados del siglo XX dieron lugar a nuevas y más sofisticadas disciplinas ingenieriles, y la invención y comercialización de los computadores dio origen a las Ciencias Computacionales, y posteriormente a la Ingeniería de Software. La mayoría de los historiadores coincide en que el siglo XX fue el más productivo en toda la historia de la humanidad, en cuanto a la cantidad y el impacto de los descubrimientos. En este siglo, la *American Telephone and Telegraph* (AT&T) jugó un papel importante en el nacimiento de la Ingeniería de Sistemas: 1) por la apremiante complejidad que planteaba el desarrollo de redes telefónicas, 2) por su tradición de investigación relativamente liberal y 3) por su solides financiera. En 1943 AT&T fusiona los departamentos de Ingeniería de Conmutación e Ingeniería de Transmisión bajo la denominación de Ingeniería de Sistemas.

Estas nuevas disciplinas de la ingeniería tienen diferentes denominaciones en los países, por ejemplo, en España, a la Ingeniería de Sistemas se le conoce como Ingeniería Informática, la cual data desde 1969 cuando en la Universidad Politécnica de Madrid se creó el Instituto de Informática, una dependencia del Ministerio de Educación y Ciencia; en 1976 se crea la facultad, pionera en los estudios de Licenciatura en Informática, titulación que se sustituye en 1996 por Ingeniero en Informática [MAR00]. El primer computador que se instaló en este país fue un IBM 650 en la compañía Telefónica en 1958. En 1962 se empezaron a introducir los primeros computadores en empresas privadas. En esa época, la enseñanza del manejo y fundamentos de estas máquinas corría a cargo de las propias empresas. En 1969 comienza de manera oficial la enseñanza de la Informática con la creación del Instituto de Informática en Madrid donde se imparten estudios de cinco años. A finales de los años 70 se empiezan a impartir Estudios Superiores de Informática en la Universidad, concebidos como diplomaturas y licenciaturas. Hace algunos años, según las directrices del entonces Ministerio de Educación y Ciencia sobre nuevas titulaciones, se han reconvertido las titulaciones a ingenierías técnicas y superiores. De forma paralela a la creación de estudios específicos de informática, en los planes de estudios de otras carreras de corte científico y tecnológico han ido apareciendo asignaturas relacionadas, en mayor o menor grado, con las Ciencias Computacionales y la Inteligencia Artificial [MAR00].

La primera referencia a la ingeniería de sistemas fue publicada en 1950, por Melvin J. Kelly, entonces director de los laboratorios de la Bell Telephone, subsidiaria de investigación y desarrollo de AT&T. Previamente, esta ingeniería se había practicado durante años, pero su reconocimiento como entidad organizativa generó mayor interés y recursos en las organizaciones. En 1950 se crea en el MIT el primer curso de postgrado relacionado, y sería el propio Hall el autor de un tratado completo sobre el tema. De acuerdo con él, la Ingeniería de Sistemas es una tecnología, por la que el conocimiento de investigación se traslada a aplicaciones que satisfacen necesidades humanas mediante una secuencia de planes, proyectos y programas de proyecto.

En Colombia, esta ingeniería inicia a finales de los años 60, cuando las aplicaciones se limitaban a uso administrativo, los sistemas operativos eran elementales y los lenguajes de programación

totalmente limitados. Poco a poco surgió la necesidad de realizar una capacitación profesional en las instituciones universitarias del país, y en 1968 las universidades Nacional, los Andes e Industrial de Santander, establecen el programa de Ingeniería de Sistemas y Computación. Los primeros profesionales, egresados entre 1972 y 1973, eran contratados mayoritariamente por las grandes empresas públicas y privadas que contaban con computadores, y por los proveedores de estos equipos.

La Ingeniería de Software se estableció desde mediados del siglo XX, cuando los científicos computacionales se dieron cuenta que el desarrollo de las máquinas de cómputo automático necesitaban algo más que hardware. Luego de las conferencias de la NATO, a finales de los 60, la academia, la industria y los Estados, concluyeron que era necesario profesionalizar la labor cuyos productos eran software, por lo que se acuñó la frase *crisis del software*. El objetivo último de esta afirmación era concientizar a la sociedad de la necesidad de prestar mayor atención al componente complementario del hardware, lo que inició movimientos en todo el mundo para atender la solicitud. La mayoría de países latinoamericanos crearon los programas en Ingeniería de Sistemas, desde los cuales orientaron la formación en desarrollo de software, mientras que el resto del mundo pensaba en una ingeniería independiente. En la primera década del siglo XXI, cuando el software se ha convertido en una de las herramientas tecnológicas de mayor penetración en el mercado mundial, se han iniciado proyectos orientados a formar profesionalmente, y específicamente, en el desarrollo de software. Las próximas décadas serán neurálgicas para concretar las iniciativas y estructurar programas en Ingeniería de Software en toda la región, que aprovechen la voluntad política de los gobiernos de apoyarlas y promocionarlas.

REFERENCIAS

[BAU69] Bauer, F.L. (1969). "Software Engineering Report." Conference Sponsored by the NATO Science Committee. Garmish, Germany, October 7-11.

[BOE06] Boehm, B. (2006). "A View of 20th and 21st Century Software Engineering". Proceedings of the 28th International Conference on Software Engineering, ICSE'06. Shanghai, China, May 20-28.

[BOE76] Boehm, B. (1976). "Software Engineering." IEEE Transaction on Computers, 25(12): 1226-1241.

[BRO75] Brooks, F.P. (1975). "The Mythical Man-Month: Essays on Software Engineering". Addison-Wesley.

[CAM07] Campbell-Kelly, M. (2007). "The History of the History of Software". Annals of the History of Computing, 29(4), 40-51.

[CAM95] Campbell, M. (1995). "Development and structure of the international software industry: 1950-1990". Business and Economic History, 24(2), 73-110.

[CAM96] Cambell-Kelly, M. & Aspray, W. (1996). "Computer: A History of the Information Machine". HarperCollins.

[CER99] Ceruzzi, P.E. (1999). "A History of Modern Computing". MIT Press.

[EBE08] Ebert, C. (2008). "A Brief history of Software technology". IEEE Software, 25(6), 22-25.

[EDI09] Editores. (2009). "La Ingeniería". Revista Digital Lámpsakos, 1, 13-21.

[HEN90] Henderson-Sellers, B. & Edwards, J.M. (1990). "The Object-Oriented Systems Life Cycle". Communications of ACM, 33(9), 142-159.

[IEE90] IEEE (1990). "ST-610-1990 - IEEE Standard Computer Dictionary." Compilation of IEEE Standard Computer Glossaries.

[ISO95] ISO/IEC. (1995). "Information Technology / Software Life Cycle Processes". ISO/IEC 12207.

[KEI96] Keil-Slawik, R. & Brennecke, A. (1996). "History of Software Engineering". En Seminar 9635. Dagstuhl.

[LEH84] Lehman, M.M. (1984). "A further model of coherent programming processes". Proceedings of the IEEE Software Process Workshop. Eghan, UK.

[MAC01] McClure, R.M. (2001). "Software Engineering Report." The NATO Science Committee. Roma, Italia, October 27-31, 1969.

[MAH90] Mahoney, M. S. (1990). "The Roots of Software Engineering." Princeton University Press.

[MAR00] Martínez, R. & García-Beltrán, A. (2000). "Breve Historia de la Informática." División de Informática Industrial. Universidad Politécnica de Madrid. Online: http://ocw.upm.es/ciencia-de-la-computacion-e-inteligencia-artificial/fundamentos-programacion/otrosrecursos/brevehistoriainformatica.pdf. [May. 2013].

[MAR91] Martin, J. (1991). "Rapid Application Development". Macmillan USA.

[MDO10] McDonald, C. (2010). "From Art Form to Engineering Discipline? A History of US Military Software Development Standards, 1974-1998". IEEE Annals, 32(4), 32-47.

[MEY90] Meyer, B. (1990). "The New Culture of Software Development." In Advances in Object-Oriented Software Engineering, Mandrioli, D. & Meyer, B. (Eds.). Prentice Hall, 1991, pp. 51-64.

[NAU68] Naur, P. & Randell, B. (1968). "Software Engineering." Report on a Conference Sonsored by the NATO SCIENCE COMMITTEE. Garmisch, Germany, October 7-11.

[PER06] Pérez, G.A. (2006). Tesis de Maestría. Universidad Autónoma del Estado de Hidalgo, México. Instituto de Ciencias Básicas e Ingeniería. Licenciatura en Sistemas Computacionales.

[PET00] Peterson, I. (2000). "Software's Origin". Online: www.maa.org/mathland/mathtrek_7_31_00.html. [Jun. 2013].

[PFL01] Pfleeger, S. L. (2001). "Software Engineering - Theory and Practice". Prentice Hall Inc.

[PRE03] Pressman, R. (2003). "Software Engineering". McGraw-Hill.

[PRE97] Pressman, R. (1997). "Software Engineering". McGraw-Hill.

[RAC97] Raccoon, L.B. (1997). "Fifty Years of Progress in Software Engineering". Software Engineering Notes, 22(1), 88-104.

[ROY70] Royce, W.W. (1970). "Managing the development of large software systems". Technical Papers of Western Electronic Show and Convention (WesCon). August 25-28, Los Angeles, USA.

[RUM92] Rumbaugh, J. (1992). “Over the waterfall and into the Whirlpool”. Journal of Object-Oriented Programming, 5(2), 23-26.
[SHE03] Shepperd, M. (2003). “Empirically-based Software Engineering”. European Journal for the Informatics Professional, 4, 37-41.
[SOF11] Softwaretop100. (2013). “The top 100 software companies in the United States”. Online: http://www.softwaretop100.org/the-top-100-software-companies-in-the-us. [Feb. 2013]
[SOU11] Sousa, G. & Da Mota, S.N. (2011). “25 Years of Software Engineering in Brazil: An Analysis of SBES History.” 25th Brazilian Symposium on Software Engineering. Sao Paulo, Brazil, September 28-30.
[YAN06] Yang, F. & Mei, H. (2006). “Development of Software Engineering: Co-operative efforts from academia, government and industry”. Proceedings of the 28th International Conference on Software Engineering (ICSE’06). Shanghai, China.
[ZEL79] Zelkowitz, M.V., Shaw, A.C. & Gannon, J.D. (1979). “Principles of Software Engineering and Design.” Prentice-Hall Software Series.

CAPÍTULO III

LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA

Federico S. BOBBIO[1]; Alexei SERNA A.[2]; Humberto RICO M.[3]; Lina CASTAÑEDA S.[4]
[1] *Instituto Nacional de Tecnología Industrial (INTI)*
Córdoba, Argentina
[2] *Instituto Antioqueño de Investigación (IAI)*
[3, 4] *Instituto Tecnológico Metropolitano (ITM)*
Medellín, Antioquia, Colombia

INTRODUCCIÓN

La Ingeniería de Software llegó tarde a América Latina, una cuestión un poco natural dada la dependencia tecnológica que esta parte del mundo ha tenido de los países desarrollados. Pero el asunto no es en qué momento se inició el trabajo formal en esta ingeniería en los países latinoamericanos, sino qué están haciendo y cómo lo están haciendo. En este capítulo se presenta una descripción del estado actual de la Ingeniería de Software en el continente, que surge la idea de presentar en detalle los procesos que se generan desde la academia, la industria y el Estado, y que impactan directamente el que hacer de los profesionales de esta ingeniería. Desde ese punto de vista se hace un recorrido por los países de la región, visitando: 1) las universidades, para encontrar los programas que ofrecen y la investigación que realizan alrededor de esta ingeniería, 2) los organismos de control, para mostrar las normas y políticas que promulgan para regular el ejercicio profesional del desarrollo de software y 3) la industria, para describir el estado de este renglón de la economía.

Para lograr el primer punto se consultó el sitio *Ranking Web de Universidades* (http://www.webometrics.info/es), en sus ediciones del último semestre de 2013. Este *Ranking* es una iniciativa del Laboratorio de Cibermetría, perteneciente al Consejo Superior de Investigaciones Científicas (CSIC), el mayor centro nacional de investigación de España. El objetivo de la búsqueda fue consultar los programas académicos, relacionados con la Ingeniería de Software, que ofrecen las diez primeras universidades del Ranking en cada país. La información que estas universidades presentan en sus páginas fue el criterio principal para incluirlas en la muestra. Es decir, no importa el nombre del programa sino la descripción que se hacía del mismo. Por esto es que algunas universidades ubicadas en estos puestos no quedaron incluidas en este reporte. Para el segundo punto se utilizaron varias estrategias: 1) se ingresó a las páginas de los organismos de control relacionado para buscar las normas específicas, 2) se utilizó un buscador empleando palabras clave y 3) se consultó a las oficinas de control y de certificación solicitando la información respectiva.

1. ARGENTINA, BRASIL, CHILE, PARAGUAY, URUGUAY

1.1 Políticas

En el ámbito de las políticas de estado que los países del Cono Sur han direccionado hacia la industria del Software y las TIC se identifican fuertes componentes que ejercen una influencia determinante en la Ingeniería de Software y en su identidad Latinoamericana. En la generalidad regional, las políticas de promoción de estas industrias han tenido como principal objetivo crear una integración entre gobiernos, comunidades científicas y el sector privado, a fin de potencializar su capacidad de formación de recurso humano, de desarrollo de nuevas tecnologías, y del aprovechamiento del flujo de capital internacional, para transformar el ciclo de emprenderismo latente en innovación aplicada, y ampliar las competencias con vistas a la superación de los desafíos económicos y sociales [MCT11]. En el caso argentino, el Plan Industrial 2020 afirma que la cadena de valor del software reviste un carácter estratégico, porque mediante la tecnología informática se posibilitan mejoras trasversales en productos y procesos de todas las ramas productivas [MIN11]. Esto plantea claramente una apuesta concreta y estratégica en pro de un sector que debe ser desarrollado y profesionalizado para obtener beneficios, y adoptar una nueva posición frente a las problemáticas actuales, teniendo injerencia directa en la Ingeniería de Software.

El bloque de países latinoamericanos han interpretado, y lo han enunciado varios autores y programas públicos, que la industria TIC representa una *respuesta de salida y de apoyo* a la crisis financiera mundial y a los desafíos sociales de inclusión. La esperanza se deposita o en la calidad y en la innovación, como lo expresa el estudio de la Secretaría Permanente del Sistema Económico Latinoamericano y del Caribe (SELA): *"En otras palabras, la tendencia histórica que se ha venido imponiendo supone que el desarrollo de software vale cada vez menos (es útil, pero no crea diferenciación) y, por tanto, lo que en término de políticas públicas debe promoverse y lo que en términos empresariales debe buscarse, es la innovación (que puede, desde luego, implementarse contratando desarrollos de software)* [SEL09]. De esta forma se busca, a través de políticas públicas variadas y en diversos formatos, crear un apoyo que genere capacidades y resultados en estos puntos fundamentales.

Ahora bien, la Ingeniería de Software sería la encargada de vincular estos componentes de innovación y calidad con el del desarrollo de Software, y con el mismo proceso de creación, por lo que se verá influenciada por las diversas aristas que estos lineamientos plantean y por la realidad práctica con que se implementen. Pero, si bien las apuestas dejan en evidencia lo que se interpreta como una industria TIC y/o de Software en establecimiento en la región, existen cuestiones culturales y humanas que han erosionado negativa y fuertemente estos avances y procesos:

- La falta de coordinación y capacidad de gestión técnica, que generan un gasto excesivo de recursos de todo tipo, por lo que, para trasladar realmente a la práctica una política definida en un alto nivel de abstracción y especialidad se debe pasar por la creación de organismos, empresas, licitaciones, contratación de instituciones académicas y académicos/científicos, que en muchas ocasiones no responden adecuadamente a las políticas expresadas, dejando por fuera el análisis real de estas equivocaciones.

- La desigualdad social y la corrupción en las estructuras públicas, a veces confundidas con idiosincrasia, son características que se pueden atribuir a la *juventud* del sistema político y social que domina estos países. Esto sin hacer análisis histórico de los años de *independencia* vividos.

Estos factores, sumados a períodos de inestabilidad económica e ideológica, han generado quiebres en procesos de avanzada que habían tomado rumbo para dar vida a una identidad industrial, profesional y académica. En este aspecto se identifica una necesidad real, concreta y clave de trabajo para potenciar la generación de nuevos profesionales, hombres de ciencia, sociedad y práctica, que ejerzan su ejercicio profesional como personas de bien y con alto compromiso.

1.1.1 Software público

Como parte de las políticas de los estados Latinos de principios de siglo XXI se identificó una fuerte tendencia hacia la búsqueda de independencia, en lo que tiene que ver con aplicaciones de gestión y de servicios y a la reutilización de desarrollos a través de programas comúnmente denominados de *Software Público*. La premisa más fuerte en la región se relacionó con la adecuación del software libre a las necesidades de las entidades públicas, un asunto que también requiere inversión en infraestructura tecnológica de soporte. Paralelamente se masificó la publicación de casos exitosos en formato abierto, y la oferta de un ecosistema de desarrollo de software integrado y con servidores propios, desde los cuales se podía consumir a la vez que evolucionar.

Este eje plantea quizás uno de los desafíos más osados a las estructuras, generalmente poco flexibles de desarrollo de software, para las dependencias estatales. Implica un cambio sustancial a la hora de pensar un desarrollo y llevarlo adelante, y aquí la tarea de la Ingeniería de Software pasa principal y claramente por un trabajo social de cambio cultural y de aprehensión de nuevos paradigmas, pensando un modelo y una forma de aplicarla que se adecúen a una variedad de entornos amplios y psicologías grupales e individuales. Por otro lado, la juventud y la pujante capacidad potencial que poseen las nuevas generaciones de profesionales, técnicos, políticos y usuarios tienen condiciones históricas y de entorno, y pueden servir como transformadoras definitivas desde la raíz de los procesos y servicios públicos. Esto se puede lograr desde el interior de los mismos Estados, como extensión a la sociedad y vinculaciones con otros países, siempre que se atienda a una transparencia franca, a un criterio trabajado y consensuado, y a una labor social sobre la voluntad y los valores. En este sentido, los países del Cono Sur utilizan, implementan o promulgan iniciativas como estas:

- Argentina: http://www.agendadigital.gob.ar/software-publico
- Brasil: http://www.softwarepublico.gov.br/
- Chile: http://www.softwarepublico.cl/
- Paraguay: http://www.setics.gov.py/plataforma-estandar-de-desarrollo-de-sistemas
- Uruguay: http://softwarepublico.gub.uy/

1.2 Formación y capacitación

En cuanto a los medios de incorporación y transferencia de la Ingeniería de Software en América Latina, particularmente en el Cono Sur, se debe hacer una salvedad antes de comenzar el desarrollo, y tiene que ver con la extensión que se podría dar al presente capítulo; pero debido a cuestiones de alcance y practicidad se realizará un enfoque básico, aunque apropiado, y de inicio de discusión/acción, con el objetivo de describir el actual panorama educativo de la Ingeniería de Software, y la valoración que se percibe en amplios sectores de la comunidad involucrada.

1.2.1 Medios de acceso

Este término se refiere a las diversas ofertas en formación, capacitación o instrucción en Ingeniería de Software, y que en la región se resumen en:

- *Educación pre-universitaria*. Se identifican casos con una superficialidad marcada y sin referencia directa a la rama profesional. Generalmente reducidos a actividades de programación o utilización de herramientas de desarrollo de software y en el marco de materias como *Informática* o *Computación*. En algunos casos con introducción a paradigmas específicos, como la programación Estructurada u Orientada por Objetos, pero manteniendo la superficialidad y, nuevamente, sin introducir conceptos firmes de la especialidad de Ingeniería de Software. La mayoría de estos casos se enmarcan en un diseño curricular amplio, que no define específicamente la necesidad de ofrecer una formación en desarrollo de software, y menos en Ingeniería de Software.

- *Educación de grado*. Dentro de las carreras de grado orientadas al software, la informática, la computación o a los sistemas, en las principales universidades del Cono Sur existen en los planes de estudio materias específicas dedicadas a la ingeniería de Software, como Análisis de Sistemas, Diseño de Sistemas, Ingeniería de Software, Calidad del Software y otras. En algunas también existen acreditaciones nacionales a nivel de carrera, que brindan una estructura homogénea. Por otro lado, en los estadios de formación universitaria la Ingeniería de Software se suele trabajar la teoría y la práctica de una determinada metodología, y actualmente la más extendida es la propuesta por Rumbaugh, Jacobson y Booch, conocida como Proceso Unificado de Desarrollo, acompañada del Unified Modeling Language (UML), que propagó rápidamente sus conceptos y sentó las bases para una nueva forma de comprender un paradigma particular para desarrollar software: la Programación Orientada por Objetos (POO). Este enfoque se complementó posteriormente con los trabajos de Sommerville y Pressman, en los que ampliaron el concepto de Ingeniería de Software, y que han tenido amplia difusión y alcance en Latinoamérica, especialmente como fuentes de consulta para profesores y estudiantes. Otros nombres recurrentes en la bibliografía relacionada son Kenneth y Julie Kendall, Ghezzi, Jazayeri, Mandrioli, Eric Gamma, Glenford Myers, Steve McConnell, Martin Fowler, y otros tantos.

  En algunos programas universitarios se identifican grupos de docentes, profesionales de otras áreas del conocimiento, con un interés particular en la Ingeniería de Software, que orientan su formación permanente hacia la misma, y que trabajan arduamente por realizar aportes significativos, como el mundo de las metodologías ágiles. El trabajo de estos investigadores ha servido para colocar a América Latina en el mapa del desarrollo de principios para concebir el software, pero los principales aportes continúan siendo extranjeros. Esto evidencia un claro retraso en los procesos de la academia, en contraste con los de la industria, las empresas y la producción científica en la materia, complementado por la falta de usar e implementar nuevas tecnologías y herramientas para impartir conocimientos y prácticas evolucionados, y con el poco incentivo a una creación o aporte regional. En la Prospectiva TIC 2020, elaborada desde Argentina, se pueden identificar algunas propuestas de mejoramiento, y se plantea el rol central de la formación en Ingeniería de Software en lo que respecta a *medidas y acciones recomendadas* para la industria y la sociedad: *"Es fundamental tener en cuenta que la implementación de cambios en el sector educativo necesita, para la generación de contenidos y material de soporte, una cantidad de tiempo sustancial y de personal altamente calificado, no sólo en los temas técnicos sino en las formas de transmitirlos."* [BAU08].

  Existen casos en los que se reflejan intentos de *rejuvenecer* contenidos, y grupos de docentes interesados en las materias relacionadas con la Ingeniería de Software, o en crear nuevas cátedras que apunten a metodologías y prácticas de punta, lo que puede significar una manera de actualizarse con los nuevos principios y desarrollos, y de que la industria acompañe de forma más cercana a esta aparente nueva revolución industrial, o al menos a la clara convergencia en las TIC,

que plantea una diferenciación de contextos extremadamente amplia hacia donde la Ingeniería de Software debe operar y solucionar, lo que anteriormente no se contemplaba.

- *Educación de pos-grado*. Estos programas poseen una mayor tendencia hacia la calidad, y las falencias de enfoque académico en los programas de maestría, doctorado y especialización, que ofrecen las principales universidades de la región, son particulares, diversas y focalizadas. En algunos casos, los programas atienden al contexto de aplicación de la Ingeniería de Software, como en la televisión digital, el software embebido, los sistemas distribuidos, el software libre, el desarrollo remoto, el acceso a internet móvil de alta velocidad y la convergencia digital en general. Pero si bien son una opción de profesionalización válida, pocos estudiantes cursan estos estudios, y menos los que vuelcan activa y productivamente los resultados a la práctica o la investigación de impacto, quedando muchas de las producciones en generación de material teórico que recorre únicamente circuitos académicos. Pero se tiene la confianza generalizada de que este es un camino que puede llevar a un ajuste en la academia de grado, y a mejorar sus prácticas con la realidad industrial y las necesidades de actualización de los futuros profesionales, porque el fomento y la vinculación de estas carreras ha tenido, en los últimos años, una promoción mayor a las realidades prácticas.

- *Educación no oficial*. Debido a que en el actual estado tecnológico el acceso a la información se puede realizar de manera autodidacta y con accesibilidad prácticamente sin barreras, se descubren diversas formas de acceder a los conocimientos en Ingeniería de Software que configuran el quehacer actual en la región:

  - Cursos de actualización de duración breve enfocados a cuestiones prácticas, que generalmente ofrecen instituciones académicas sin aval o certificado oficial

  - Bibliografía y material de producción científica, accesible a través de los canales tradicionales e internet

  - Congresos y encuentros de las comunidades, que ofrecen un espacio común para compartir experiencias y congregar los avances de diversas fuentes, a fin de potenciar nuevas direcciones y generaciones

  - Prácticas en entidades, empresas o industrias, es otra fuente identificada para acceder a conocimientos en Ingeniería de Software, y permite generar experiencias y métricas en pos de una evolución real

1.2.2 Valoración general

En cuanto a la valoración general de los medios de acceso a la formación en Ingeniería de Software en el Cono Sur, los entrevistados y las percepciones recolectadas apuntan casi en su mayoría a que existe un atraso evidente de la academia formal, debido a que se centra principalmente en las carreras de grado, y a que no se oficializan propuestas genuinas ni originales. Es decir, no se detecta un *producto* latino, amplio y tangible, de Ingeniería de Software, ni un aporte sustancioso a la globalidad de la profesión.

Por otro lado, se denota una falta de iniciativa en la industria hacia la inversión en procesos de calidad, que desarrollen avances específicos en Ingeniería de Software, una situación que se subsana por medio de incentivos y aportes estatales. Aunque hasta el momento, en términos generales, la

Ingeniería de Software no es un área favorecida con subsidios, tampoco se vislumbra una línea clara que le genere aportes desde áreas como la cultura, las experiencias o la idiosincrasia, ni propuestas de mejoramiento alterno. Como contrapartida al atraso de la academia de grado se evidencia un avance en la capacitación auto-gestionada, desde nuevos medios de comunicación que permiten una formación en línea cada vez más eficaz y avalada, y que abre nuevos frentes hacia el futuro. Cabe destacar la importancia de los encuentros en congresos y jornadas en todos los niveles, en los que se generan redes de profesionales que trabajan conjuntamente a lo largo del tiempo, en temas específicos de interés, y que cada vez se presentan resultados más concretos que pueden potencializar avances y nuevas generaciones.

1.3 Programas relacionados con la Ingeniería de Software

A continuación se relacionan algunas instituciones de la región que ofrecen programas relacionados con la Ingeniería de Software, o con los roles que un ingeniero de software desempeña en las organizaciones.

ARGENTINA

Institución	Título	Ciudad	Duración	Sector
Universidad Argentina de la Empresa	Técnico Universitario en Desarrollo de Software	Capital Federal	3 Años	Privado
Universidad CAECE	Técnico Universitario en Administración de Proyectos de Software	Capital Federal	3 Años	Privado
Universidad Nacional del Sur	Ingeniero en Sistemas de Software	Bahía Blanca	5 Años	Publica
Universidad del Aconcagua	Licenciado en Informática y Desarrollo de Software	Mendoza	4 Años	Privado
Universidad Empresarial Siglo XXI	Ingeniero en Software	Córdoba	5 años	Privado
Pontificia Universidad Católica Argentina Santa María de los Buenos Aires	Especialista en Ingeniería de Software	Capital Federal	1 Año	Privado
Universidad CAECE	Magister en Ingeniería de Software	Capital Federal	2 Años	Privado
Universidad Nacional de La Plata	Especialista en Ingeniería de Software Magister en Ingeniería de Software	La Plata	1 Año 2 Años	Publica/Estatal
Universidad Nacional de Jujuy	Magister en Ingeniería de Software	Salvador de Jujuy	2 Años	Publica/Estatal
Universidad Nacional de La Rioja	Magister en Ingeniería de Software	La Rioja	3 Años	Publica/Estatal
Universidad Nacional de San Luis	Especialista en Ingeniería de Software Magister en Ingeniería de Software	San Luis	1 Año 2 Años	Publica/Estatal

CHILE

Institución	Título	Duración	Ubicación
CFT CENCO	Técnico De Nivel Superior En Análisis De Sistemas	7 Semestres 5 Semestres	San Felipe, Quillota
CFT CENCO	Técnico De Nivel Superior En Programación Y Diseño Informático	5 Semestres	San Felipe, Quillota
CFT CRECIC	Análisis Computacional De Sistemas	6 Semestres	Curanilahue

CFT CRECIC	Técnico Analista Programador Computacional	6 Semestres	Concepción
CFT De ENAC	Técnico Analista Programador	6 Semestres	Santiago
CFT DUOC UC	Analista Programador Computacional	5 Semestres	Padre Alonso Ovalle, Melipilla
CFT ICEL	Análisis De Sistemas	7 Semestres	Santiago
CFT INACAP	Analista Programador	5 Semestres	Parinacota, Iquique, Antofagasta, Copiapo, La Serena, Renca, Maipú, Apoquindo, Pérez, Rosales, Rancagua, Talca, Curicó, Chillan, Los Ángeles, Talcahuano, Temuco, Valdivia, Coyhaique, Puerto Montt, Osorno
CFT Instituto Tecnológico De Chile I.T.C.	Análisis De Sistemas	5 Semestres	Santiago
CFT Laplace	Análisis De Sistemas	7 Semestres	Santiago
CFT Laplace	Programación En Computación E Informática	5 Semestres	Santiago
CFT Magnos	Informática Mención Desarrollo De Aplicaciones	6 Semestres	Santiago
CFT Magnos	Informática Mención Desarrollo De Aplicaciones Y Diseño Web	6 Semestres	Santiago
CFT Massachusetts	Programación Computacional Mención Análisis De Sistemas	5 Semestres	Linares
CFT Prodata	Técnico En Informática Con Especialización En Desarrollo De Software Y Multimedios	5 Semestres	Los Lagos
CFT San Agustín De Talca	Analista Programador	8 Semestres	Talca, Cauquenes, Linares
CFT San Agustín De Talca	Técnico En Desarrollo De Aplicaciones Web.	7 Semestres	Talca
CFT Simón Bolívar	Programación Y Diseño Informático	5 Semestres	Santiago
CFT Simón Bolívar	Técnico En Análisis Y Programación	5 Semestres	Santiago
IP AIEP	Ingeniería De Ejecución En Informática Mención Desarrollo De Sistemas	9 Semestres	La Serena, Providencia, Santiago, San Fernando, Rancagua, Curicó, Concepción
IP AIEP	Programación Y Análisis De Sistemas	7 Semestres	La Serena, Viña Del Mar, San Felipe, Providencia, Santiago, San Fernando, Rancagua, Curicó, Talca, Los Ángeles, Concepción, Temuco, Puerto Montt, Osorno
IP AIEP	Programación Computacional	5 Semestres	Valparaíso Providencia, Santiago, San Fernando, Rancagua Concepción
IP CIISA	Técnico En Programación Computacional	5 Semestres	Santiago
IP De Chile	Analista Programador Computacional	5 Semestres	La Serena, Santiago, San Joaquín Rancagua

IP De Ciencias De La Computación Acuario Data	Análisis De Sistemas	9 Semestres	Santiago
IP DR. Virginio Gómez G.	Técnico Analista Programador	4 Semestres	Los Ángeles, Chillan, Concepción
IP DUOC UC	Analista Programador Computacional	5 Semestres	Viña Del Mar, Plaza Oeste, Puente Alto, Maipú, San Joaquín, San Bernardo, Estación Central, San Joaquín, Concepción
IP Santo Tomas	Analista Programador	5 Semestres	Temuco
Universidad Central De Chile	Técnico De Nivel Superior En Programación	4 Semestres	Santiago
Universidad De Las Américas	Técnico De Nivel Superior En Redes Informáticas	4 Semestres	Viña Del Mar
Universidad De Playa Ancha De Ciencias De La Educación	Programador En Aplicaciones Computacionales	6 Semestres	Valparaíso
Universidad Técnica Federico Santa María	Técnico Universitario En Programación De Computadores	8 Semestres	Viña Del Mar
Universidad Técnica Federico Santa María	Ingeniería Ejecución En Software	6 Semestres 4 Semestres	Santiago
Universidad Técnica Federico Santa María	Ingeniería Ejecución En Software	6 Semestres	Rancagua
Universidad UCINF	Análisis De Sistemas	7 Semestres 6 Semestres	Santiago

URUGUAY

Institución	Título	Ubicación	Nivel Académico
Universidad De La Empresa	Analista En Ingeniería Informática	Montevideo	Licenciatura/Pregrado
Universidad ORT Uruguay	Analista En Tecnologías De La Información	Montevideo	Técnica/Pregrado
Universidad ORT Uruguay	Analista Programador	Montevideo	Técnica/Pregrado

PARAGUAY

Institución	Título	Ciudad	Nivel Académico
Universidad Americana	Ingeniería En Informática Con Énfasis En Ingeniería Del Software	Asunción	Pregrado
Universidad Autónoma De Asunción	Ingeniería En Ciencias De La Computación	Asunción	Pregrado
Universidad Autónoma De Asunción	Licenciatura En Ciencias Informáticas	Asunción	Licenciatura/Pregrado
Universidad Autónoma De Asunción	Programación De Computadoras	Asunción	Técnico/Pregrado
Universidad Católica Nuestra Señora De La Asunción	Análisis De Sistemas Informáticos	Asunción	Técnico/Pregrado
Universidad Columbia Del Paraguay	Licenciatura En Análisis De Sistemas	Asunción	Licenciatura/Pregrado
Universidad Ibero-Americana	Licenciado En Analista De Sistemas Informáticos	Asunción	Licenciatura/Pregrado
Universidad Nacional De Pilar	Análisis De Sistemas	Pilar	Técnica/Pregrado

BRASIL

Institución	Nivel	Título	Ciudad
Fundacao Universidade Federal Do Pampa-Unipampa-Unipampa	Pregrado	Engenharia De Software	Alegrete
Centro Universitario Univates-Univates	Pregrado	Engenharia De Software	Lajeado
Universidade Da Regiao De Joinville-Univalle	Pregrado	Engenharia De Software	Joinville
Universidade Do Oeste De Santa Catarina-UNOESC	Pregrado	Engenharia De Software	Videira
Centro Universitario De Maringá-UNICESUMAR-UNICESUMAR	Pregrado	Engenharia De Software	Maringá
Pontificia Universidade Católica De Minas Gerais-PUC MINAS	Pregrado	Engenharia De Software	Belo Horizonte
Universidade Federal De Goiás-UFG	Pregrado	Engenharia De Software	Goiania
Universidade De Rio Verde-FESURV	Pregrado	Engenharia De Software.	Rio Verde
Universidade Federal Do Amazonas-UFAM	Pregrado	Engenharia De Software.	Itacoatiara
Universidade Federal Do Rio Grande Do Norte-UFRN	Pregrado	Engenharia De Software.	Natal
Universidade Federal Do Ceará-UFC	Pregrado	Engenharia De Software.	Quixadá

1.4 Reglamentaciones relacionadas con la Ingeniería de Software

Los organismos evaluadores y acreditadores en la región son:

- Argentina: Comisión Nacional de Evaluación y Acreditación Universitaria (CONEAU)
- Brasil: Comissão Nacional de Avaliação da Educação Superior (CONAES)
- Chile: Comisión Nacional de Acreditación (CNA)
- Paraguay: Agencia Nacional de Evaluación y Acreditación de la Educación Superior (ANEAES)
- Uruguay: No posee [DAV11].

Estas comisiones y los sistemas de aseguramiento de la calidad correspondientes fueron creados en el marco de los términos contenidos en el Memorándum de Entendimiento de 2002, firmado por los Ministros de Educación del MERCOSUR, Bolivia y Chile [MER02], que previamente se habían establecido en otro Memorándum de 1998, y que dieron origen a leyes que estructuraron los diseños curriculares y las orientaciones de las carreras de grado en todos los países. A nivel académico, el impacto en la Ingeniería de Software de esta iniciativa fue relevante, entre otras cosas, porque introdujeron materias en la mayoría de currículos, como las ya especificadas. En la mayoría de casos la formación en el área se trabajó con una mirada teórica y estructurada, basada principalmente en los autores extranjeros que publicaron sus obras a finales del siglo XX y principios del XXI.

Agradecimientos
El anterior contenido se desarrolló con la colaboración de los siguientes profesionales e instituciones:

- Ing. Consuelo López
- Ing. Emiliano Zilocchi
- Fabio Grigorjev
- Ing. Álvaro de Mendarozqueta
- Taller Technologies

- Ing. Tomás Aliaga
- Thiago Coelho Prado
- Lic. Germán Ceballos
- Santiago Rosa
- Lic. Renato Cherini
- Ing. Victoria Martínez Suárez
- Instituto Nacional de Tecnología Industrial
- Universidad Tecnológica Nacional
- Guilherme Lacerda
- Ing. Gonzalo Matheu
- Rodrigo Liberal
- Dr. Ricardo Medel
- Dr. Alejandro Hossian
- Lic. Daniel Bronstein
- Ing. Paula Lucila Strada
- Lic. Elisa Zabala

2. RESTO DE SURAMÉRICA

2.1 VENEZUELA

- *Universidad de los Andes*. Ofrece el programa de Ingeniería en Sistemas, en el que forma profesionales con conocimientos básicos, científicos, técnicos y éticos; capaces de planificar, proyectar, mantener, supervisar y administrar proyectos en el área de Control y Automatización, Investigación de Operaciones y Sistemas Computacionales. Ingenieros creativos, emprendedores, capaces de resolver problemas en su área de aplicación y de manera interdisciplinaria. El programa tiene una duración de diez semestres, nueve para cursar las asignaturas y uno para el trabajo de grado. El enfoque de formación se orienta a: 1) Ingeniería de Software (incluyendo BD), 2) Inteligencia Artificial (incluyendo compiladores) y 3) Sistemas Operativos (incluyendo Redes). Además, tiene investigadores trabajando en las áreas de Sistemas Distribuidos y Paralelos, Ingeniería del Conocimiento, Ingeniería del Dato, Robótica, entre otros. El grupo de investigación de Ingeniería de Datos y Conocimiento (GIDyC), tiene como principal actividad investigar y realizar prototipos que avalen los resultados teóricos obtenidos en las áreas de Ingeniería de Software, Bases de Datos, Ingeniería de Métodos, Estructuras de Acceso, Interfaz Humano/Computador, Sistemas Multi-agentes y Computación Gráfica.

- *Universidad Simón Bolívar.* Ofrece el programa de Ingeniería de Computación, con el objetivo de formar un profesional altamente calificado para desempeñarse en situaciones susceptibles de automatización mediante el uso de sistemas de computación digital. Sus egresados se pueden desempeñar tanto en el sector público como en el privado, en empresas manufactureras y distribuidores de equipos de computación, agencias de servicios de computación, compañías petroleras, ministerios, banca, organismos financieros y en toda empresa u organización que utilice un equipo de computación. El programa tiene una duración de diez semestres, otorgándole al egresado el título de Ingeniero de Computación, luego de cumplir un Plan de Estudios que comprende 205 unidades-crédito. El énfasis que tiene esta Ingeniería comprende una formación básica profesional en las áreas de programación, algoritmos, bases de datos, sistemas de

operación, Ingeniería de Software, interfaces persona-máquina, cálculo numérico, lenguajes de programación, investigación de operaciones, sistemas de información y matemáticas.

- *Universidad Dr. Rafael Belloso Chacín*. La universidad ofrece el programa de Ingeniería en Informática, y sus egresados están capacitados para el diagnóstico de las necesidades de información en las organizaciones y el diseño e implantación de los sistemas automatizados orientados a proveer, de forma confiable y oportuna, la información requerida para la toma de decisiones en los niveles tácticos, estratégicos y operativos. El programa tiene una duración de doce semestres estipulados en el plan de estudio, y su énfasis es el desarrollo de software.

- *Universidad Centroccidental Lisandro Alvarado*. Ofrece los siguientes programas:

  - *Análisis de Sistemas*. Es un profesional capacitado para planificar, elaborar y coordinar los procedimientos automáticos y/o manuales asociados a los sistemas; define, en combinación con la gerencia, las necesidades de información de una organización administrativa; estudia la factibilidad técnico-económica de las alternativas que satisfacen estas necesidades, y evalúa el costo-efectividad de los recursos humanos, máquinas y técnicas empleadas. Así mismo, maneja los sistemas de procesamiento electrónico de datos referidos a su área de competencia. El programa tiene una duración de seis semestres, y sus egresados se pueden desempeñar tanto en la administración pública como en la privada, y en aquellas instituciones que manejen grandes volúmenes de información y requieran automatización de la misma, para la comunicación en redes, la Intranet y Servidores. Así mismo, en Ministerios, Compañías Petroleras, de Servicios y también en Institutos de Educación Superior.

  - *Ingeniería en Informática*. Es un profesional capacitado para el manejo de los recursos informáticos; instrumenta, analiza y diseña sistemas de información que permitan el desarrollo integral de la organización; desarrolla distintos tipos de estructuras lógicas para solucionar problemas con el uso del computador; maneja diferentes lenguajes de programación; usa técnicas y disciplinas afines a los sistemas de información, como aspectos administrativos, organizativos, estadísticos y control de proyectos, y está capacitado para liderar proyectos que requieren el manejo de grandes volúmenes de información. El programa tiene una duración de diez semestres, y los egresados están capacitados para desempeñar sus funciones en empresas privadas y públicas, sobre todo en aquellos organismos que requieran el manejo de grandes volúmenes de información. Pueden participar en equipos multidisciplinarios manejando modelos computacionales y los lenguajes de programación que correspondan. Está en capacidad de elaborar nuevos modelos computacionales e informáticos, y está capacitado para ingresar al sistema de educación superior como docente o investigador.

- *Universidad Católica Andrés Bello (UCAB)*. Ofrece el programa de Ingeniería Informática, en el que forma un profesional integral con capacidad organizativa y de trabajo en equipo, con alto sentido crítico y compromiso social. Posee una formación científica y técnica que le permite analizar y dar soluciones a problemas relacionados con el diseño, producción e implantación de sistemas informáticos, con conocimientos teóricos prácticos en las áreas de Ciencias Básicas, Ciencias de la Computación, Telemática e Ingeniería de Software; con formación en el área Gerencial, que le permite trabajar en las diferentes áreas de conocimiento y con la capacidad de seleccionar cualquiera de estas áreas para continuar con su aprendizaje hacia una especialización. El programa tiene una duración de diez semestres, con un énfasis enfocado en las áreas de Desarrollo

de Software y Telemática. El Grupo de Investigación en Ingeniería del Software (INGESOF) realiza sus procesos en las áreas de:

- Ingeniería de Requisitos
- Gestión de Proyectos en Sistemas de Información
- Gestión de la Calidad del Software
- Diseño de Sistemas de Información
- Programación y Pruebas de Sistemas de Información
- Ingeniería del Software Libre
- Sistemas de Información Gerencial

- *Universidad Nueva Esparta*. Ofrece el programa Licenciatura en Computación, cuyos egresados están en capacidad de manejar hardware y software de actualidad para dar asistencia a las necesidades de empresas nacionales e internacionales, asiste a profesionales de otras disciplinas, crea paquetes especializados que contribuyen con el progreso y la sistematización de distintas áreas, está preparado para actuar en equipos expertos y en sistemas de información, para adaptarse a las tendencias contemporáneas. Además, demuestra su ética profesional para respetar la propiedad intelectual y el derecho a la reserva de la información. También maneja tecnologías de punta en el análisis, desarrollo e implantación de sistemas en redes locales, nacionales y mundiales. El programa tiene una duración de doce semestres.

2.1.1 Reglamentaciones relacionadas con la Ingeniería de Software

- *Ley de Tecnología de Información* (http://www.fau.ucv.ve/documentos/noticias/ley_tecnologia_informacion.pdf). Que tiene por objeto establecer las normas, principios, sistemas de información, planes, acciones, lineamientos y estándares, aplicables a las Tecnologías de Información; además de estipular los mecanismos que impulsarán su extensión, desarrollo, promoción y masificación en todo el ámbito del Estado. Se excluye del objeto de esta ley lo previsto en las leyes que regulan la materia de contenidos de información y de telecomunicaciones.

- *Ley Orgánica de Ciencia, Tecnología e Innovación (LOCTI)* (http://ociweb.mcti.gob.ve/@api/deki/files/6305/=mcti-Ley_de_Ciecia_y_Tecnologia.pdf). Esta Ley tiene por objeto dirigir la generación de ciencia, tecnología, innovación y sus aplicaciones, con base en el ejercicio pleno de la soberanía nacional, la democracia participativa y protagónica, la justicia y la igualdad social, el respeto al ambiente y la diversidad cultural, mediante la aplicación de conocimientos populares y académicos. A tales fines, el Estado Venezolano, formulará, a través de la autoridad nacional con competencia en materia de ciencia, tecnología, innovación y sus aplicaciones, enmarcado en el Plan Nacional de Desarrollo Económico y Social de la Nación, las políticas públicas dirigidas a la solución de problemas concretos de la sociedad, por medio de la articulación e integración de los sujetos que realizan actividades de ciencia, tecnología, innovación y sus aplicaciones como condición necesaria para el fortalecimiento del Poder Popular.

El organismo que regula la profesión de la Ingeniería de Software y la Información en Venezuela es el Ministerio de Ciencia Tecnología e Innovación (MCTI), quien emite las normas y regulaciones necesarias para el ejercicio de la profesión. Algunas empresas que se destacan en el sector son: DBAccess (certificada CMM), Softech (Producto Empresarial Profit Plus), Actimedia Digital, C. A., B&mt Business Management Technology y Computus Consulting It, C.A, las cuales desarrollan actividades alrededor de: 1) desarrollo, arquitectura e innovación, infraestructura, calidad, mercadeo

y usabilidad, gestión de talento en red, administración legal, 2) estrategia corporativa, y en la implantación, soporte, migración, programación y adiestramiento y 3) arquitectura, desarrollo y soporte.

2.2 ECUADOR

- *Escuela Superior Politécnica del Litoral*. Ofrece los siguientes programas:

  - *Ingeniería en Ciencias Computacionales*. La misión es formar ingenieros en Ciencias Computacionales que a lo largo de su vida profesional potencien sus capacidades de investigar, proponer, diseñar e implementar soluciones a problemas, del país y la región, dentro de equipos interdisciplinarios, aplicando competencias fundamentales en su campo, considerando aspectos sociales, económicos y ambientales, y enmarcados en lo ético y lo moral. Entre otras, este profesional estará capacitado para plantear y liderar exitosamente el desarrollo y la implantación de soluciones vinculadas a su disciplina, ya sea como gestor, empleado o asesor de una empresa, y de esta manera contribuir a los distintos segmentos de la sociedad. Tendrá la habilidad para diseñar, implementar y evaluar un sistema basado en computadores, procesos, componentes o programas que cumplan necesidades específicas. El programa tiene una duración de diez semestres.

  - *Licenciatura en Sistemas de Información*. Este programa forma profesionales emprendedores, analíticos, con sólidos conocimientos en tecnologías de información, capaces de liderar áreas informáticas y desarrollar aplicaciones, que permitan automatizar y optimizar procesos, aprovechando de forma adecuada los recursos de una organización, para apoyar en la toma decisiones y el logro de sus objetivos. Su misión es formar profesionales innovadores, con capacidad de análisis y habilidad para trabajar en equipos interdisciplinarios, para el desarrollo y gestión de sistemas de información. Los egresados tienen la habilidad para diseñar, desarrollar, implementar y evaluar sistemas de información basados en tecnología y procesos que cumplan necesidades específicas. El programa tiene una duración de diez semestres.

- *Universidad San Francisco de Quito.* Ofrece el programa de Ingeniería en Sistemas, a través del cual forma profesionales con conocimientos de computación, sistemas e informática empresarial, que utilizan los sistemas automatizados en la resolución de problemas y facilitando el trabajo de las empresas en las tareas administrativas, productivas y de investigación. Promueve en el graduado el desarrollo de habilidades de investigación, análisis y síntesis que le permiten entender y liderar la adopción y desarrollo de nuevas tecnologías, de una manera efectiva y eficiente. El programa tiene una duración de cinco años y su énfasis es el desarrollo de software.

- *Escuela Politécnica Nacional.* Ofrece el programa de Ingeniería de Sistemas, en el que los ingenieros adquieren las competencias necesarias para proveer soluciones de sistemas de información, gestionar proyectos informáticos, desarrollar software, y gestionar la infraestructura de Tecnologías de Información y Comunicación de las organizaciones. Sus egresados poseen los conocimientos fundamentados en las Ciencias Básicas, las Ciencias Sociales y Humanísticas, las Ciencias Administrativas, las Ciencias de la Computación, las Redes, las Comunicaciones, las Seguridades, la Ingeniería de Software, y la Gestión de las TIC. El programa tiene una duración de nueve semestres, ocho para terminar las asignaturas y uno para el trabajo de grado o proyecto de titulación. El énfasis del programa es el desarrollo de software.

- *Pontificia Universidad Católica del Ecuador*. Ofrece el programa de Ingeniería de Sistemas y Computación, en la que el egresado es un profesional capaz y comprometido con el desarrollo de la sociedad, por su ingenio en el desarrollo de sistemas de información. El cambio continuo de la tecnología obliga a que este profesional gestione y aproveche los recursos tecnológicos disponibles. Se puede desempeñar en cualquier tipo de organización o empresa en el departamento de sistemas, planificando, diseñando, desarrollando, auditando e implantando sistemas de información en intranets, extranets, redes LAN y WAN. En general, el ingeniero en sistemas se desempeña como el arquitecto y el administrador de las tecnologías de la información, además, su formación le permite ejercer exitosamente la función de gerente. El programa tiene una duración de nueve semestres.

- *Universidad Técnica Particular de Loja*. Ofrece el programa de Sistemas Informáticos y Computación con una duración de diez semestres. Estos profesionales son capaces de poner en práctica las habilidades y destrezas adquiridas a lo largo de la carrera, poniéndolas al servicio de la sociedad para apoyar el desarrollo tecnológico y del conocimiento.

- *Universidad de Cuenca*. Ofrece el programa de Ingeniería de Sistemas con una duración de diez semestres, cuya misión es formar ingenieros informáticos en las áreas de ingeniería de software y gestión de la información y el conocimiento; profesionales altamente competitivos a nivel nacional e internacional, con bases sólidas en los campos tecnológico, ético, humanístico y de investigación, conscientes de la necesidad de actualizar sus conocimientos constantemente.

- *Escuela Politécnica del Ejército Ecuador.* Ofrece el programa de Ingeniería de Sistemas e Informática, en el que los profesionales graduados desarrollan capacidades en el campo cognitivo y en el axiológico, que les permite adquirir competencias en fundamentos de Ingeniería de Software, en el uso de herramientas tecnológicas (sistemas operativos, lenguajes de programación, bases de datos, herramientas CASE). El programa tiene una duración de nueve semestres, un pre-politécnico y una tesis o proyecto de graduación, y su énfasis es la Ingeniería de Software, de acuerdo con lo estipulado en su campo ocupacional.

- *Escuela Superior Politécnica de Chimborazo*. Ofrece el programa de Ingeniería en Sistemas Informáticos, en el que el egresado será capaz de demostrar sus capacidades relacionadas con conocimientos (saber), habilidades, destrezas (saber hacer) y actitudes (saber ser). Entre otras, estará en capacidad de: 1) conceptualizar problemas de sistematización de información y evaluar la factibilidad de las alternativas de soluciones informáticas, 2) emprender y gestionar un proyecto de software y 3) aplicar habilidades e identificar técnicas y herramientas tecnológicas en el desarrollo de Sistemas Informáticos. El programa tiene una duración de diez semestres, y su énfasis es la Ingeniería de Software.

- *Universidad de Las Américas*. Oferta el programa de Ingeniería en Sistemas de Computación e Informática, cuyos egresados tendrán la capacidad de analizar, diseñar, desarrollar y operar soluciones en el campo de los sistemas de computación e informática, en beneficio de las organizaciones, a través de un enfoque sistémico, incorporando nuevas tecnologías y empleando prácticas certificadas. Podrá también aplicar metodologías para el desarrollo de software, implementar y administrar infraestructura de tecnologías de la información, gestionar y asegurar la integridad de los datos. El programa tiene una duración de diez semestres, y una de las líneas de énfasis es la Ingeniería de Software.

- *Universidad Técnica Estatal de Quevedo*. Ofrece el programa de Ingeniería en Sistemas, cuyo perfil es un profesional que aplica los conocimientos científicos y técnicos y de ingeniería, para implementar en las empresas e Instituciones soluciones que permitan optimizar sus procesos, utilizando como principal herramienta los recursos informáticos (software, hardware y redes de computadores) con los más altos estándares de calidad. Este profesional podrá desempeñarse con eficiencia en el desarrollo de software, considerando los mandamientos de la ética informática. El programa tiene una duración de diez semestres, y su énfasis es la Ingeniería de software.

2.2.1 Reglamentaciones relacionadas con la Ingeniería de Software

- *NTE INEN ISO/IEC 23026*. Ingeniería de software - Práctica recomendada para el Internet - Ingeniería, administración y ciclo de vida de sitios Web.

- *NTE INEN ISO/IEC TR 29138-1*. Tecnologías de la información - Consideraciones de accesibilidad para personas con discapacidades. Parte 1: Resumen de necesidades del usuario.

- *NTE INEN ISO/IEC TR 29138-2*. Tecnologías de la información - Consideraciones de accesibilidad para personas con discapacidad. Parte 2: Inventario de las Normas.

El organismo encargado de regular la Ingeniería de Software en Ecuador es la Secretaria de la Administración Pública. Según la Federación Ecuatoriana de Exportadores (Fedexpor), 480 firmas, en especial pequeñas y medianas, se dedican al desarrollo de software, y emplean por lo menos a 8000 trabajadores. Algunas de estas industrias son:

- Insoft Soluciones de Software: Diseño y desarrollo de aplicaciones web
- Fugu Software Factory: Consultoría, productos Open Source, desarrollo de portales web
- Alfa Digital: Diseño web, desarrollo de software a la medida
- Grupo Provedatos: Desarrollo de software
- La Era Digital: Diseño, desarrollo de sitios web y software a la medida

2.3 PERÚ

- *Pontificia Universidad Católica del Perú.* Ofrece el programa de Ingeniería Informática, cuyo egresado estará capacitado, entre otras, para diseñar sistemas, componentes o procesos que satisfagan las necesidades presentadas, para lo que aplica los conocimientos relacionados con los lenguajes de programación, modelado de sistemas de información, construcción de software de calidad y administración de recursos tecnológicos. El programa tiene una duración de diez años: cuatro en estudios generales ciencias y seis en Facultad. Los énfasis son en sistemas de información e Ingeniería de Software, a las cuales se suman los sólidos fundamentos que recibe en Ciencias Computacionales, tecnología de computadoras y redes. El Grupo de Investigación y Desarrollo en Ingeniería de Software (GIDIS) se dedica al estudio y desarrollo de métodos, técnicas y herramientas, entre otros, necesarios para el trabajo eficiente y eficaz del ingeniero de software.

- *Universidad Nacional Mayor de San Marcos.* Ofrece el programa de Ingeniería de Software, cuyo egresado estará en condiciones de asumir las siguientes responsabilidades:

- Desarrollar productos software, de manera eficiente y efectiva, considerando estándares internacionales de calidad
- Administrar proyectos de desarrollo de software de gran escala
- Liderar equipos de especialistas en tecnologías de información para el desarrollo de software
- Constituir y dirigir empresas dedicadas a la investigación, el desarrollo y la consultoría de software

Los profesionales egresados de la especialidad podrán desarrollar sus actividades en forma independiente o en empresas públicas o privadas, nacionales o extranjeras, dedicadas al desarrollo de software, a la prestación de servicios de consultoría y asesoría en ingeniería de software y tecnologías de información, a la investigación científica y tecnológica, entre otras. El programa tiene una duración de diez semestres, más uno electivo, y su énfasis es la Ingeniería de Software. Cuenta con la Revista de Investigación de Ingeniería de Sistemas e Informática (RISI), cuyas líneas de investigación son: 1) Tecnologías Open Source y Software Libre, 2) Proceso de Desarrollo de Software y 3) Ingeniería de Software dirigido por modelos.

- *Universidad Ricardo Palma*. Ofrece el programa de Ingeniería Informática, cuyo objetivo principal es preparar ingenieros para la práctica exitosa de la ingeniería informática, a través del análisis, diseño y gestión de sistemas de información, aplicados a la mejora y optimización de los procesos de una organización. Estos profesionales demuestran una sólida competencia técnica para el análisis, el diseño, la implementación y la gestión de sistemas de información basados en software, para mejorar y optimizar los procesos en las organizaciones. El programa tiene una duración de diez semestres, y los egresados estarán capacitados para hacer sistemas de información sobre la base de un plan estratégico de tecnología de la información.

- *Universidad de San Martín de Porres*. Ofrece el programa de Ingeniería de Computación y Sistemas, que se orienta a la formación de profesionales cuya labor es desarrollar sistemas de información que permitan resolver problemas. La misión es formar profesionales en la especialidad de Ingeniería de Sistemas de Información con sólidos valores, competentes y creativos para resolver problemas en el contexto laboral y social; desarrollar habilidades para responder a los cambios y desafíos del entorno; fomentar la investigación aplicada en el desarrollo de sistemas de información; promover actividades de extensión y proyección social que beneficien a la sociedad. Estos egresados planifican el ciclo de vida de los sistemas de información, y aplica en forma óptima la metodología de desarrollo de sistemas de información a fin de mejorar los procesos y resolver problemas en las organizaciones. El programa tiene una duración de diez semestres. El Laboratorio de Base de Datos e Ingeniería de Software investiga y desarrolla el conocimiento tecnológico, siguiendo una de las principales tendencias de las Ciencias Computacionales, apoyado en las más recientes metodologías, estándares y herramientas de la industria del software, como las tecnologías orientada a objetos, los cuales proyectan a la optimización de resultados en el proceso de desarrollo de sistemas, haciendo posible la construcción rápida de software basado en componentes.

- *Universidad de Lima*. Ofrece el programa de Ingeniería de Sistemas, en el que asume su responsabilidad preparando profesionales con talento para identificar, diseñar, implementar y validar soluciones a los problemas, utilizando las tecnologías de la información (TI). El egresado está preparado para afrontar los retos y las exigencias de las organizaciones, gracias a su capacidad para integrar los procesos de negocios, y de proponer, desarrollar, implementar y gestionar soluciones basadas en tecnologías de información que se aplican a las mismas,

permitiéndoles alcanzar sus objetivos estratégicos de una manera efectiva. El programa tiene una duración de diez semestres. El objetivo del Área de Ingeniería de Software es otorgar al estudiante conocimientos de análisis, diseño, desarrollo y despliegue de soluciones computacionales que respondan a las expectativas empresariales, los cuales deben estar basados en las mejores prácticas. En el Laboratorio de Ingeniería de Software los estudiantes mejoran sus habilidades en cada etapa del desarrollo de proyectos de software, utilizando computadores con gran capacidad de proceso y con herramientas informáticas de tipo software comercial y *open source*, como Rational, Eclipse, Netbeans, Android, entre otras, en diferentes entornos operativos que incluyen soluciones de éxito en el mercado. Estas herramientas son usadas en cada una de las etapas del ciclo de vida de los productos software.

- *Universidad Católica de Santa María*. Ofrece el programa de Ingeniería de Sistemas con énfasis en la Ingeniería de Software y los Sistemas de Información. El programa tiene una duración de diez semestres. El egresado es un profesional competente para el desarrollo de productos software de calidad, que responda a los exigentes y complejos requerimientos de la demanda actual con una fuerte tendencia innovadora, por tal razón posee las siguientes capacidades:

  - Demuestra dominio sobre las capacidades y conocimientos de Ingeniería de Software y aspectos profesionales necesarios para ejercerla

  - Posee capacidad de gestionar proyectos con objetivos contrapuestos, considerando la factibilidad de los mismos, y las limitaciones de tiempo, costo, recursos humanos, integración con los sistemas existentes y la propia organización

  - Diseña soluciones adecuadas en uno o más dominios de aplicación, utilizando métodos de Ingeniería de Software que integren aspectos éticos, sociales, legales, de seguridad y económicos

  - Conoce y aplica las teorías, modelos y técnicas actuales que provean la base para la identificación y análisis de problemas, el diseño de software de calidad, su desarrollo, implementación, verificación y documentación

- *Universidad Peruana de Ciencias e Informática*. Ofrece el programa de Ingeniería de Sistemas e Informática, cuya formación profesional le permite al egresado participar activamente en cada etapa del ciclo de software, y en la gestión de proyectos que involucran la aplicación de tecnologías de información a nivel de software e infraestructura aplicando las mejores prácticas y bajo los estándares de calidad que el mercado requiere. El programa tiene una duración de diez semestres, y tiene énfasis en la creación de herramientas tecnológicas en el campo de la informática, para implementarlas en cualquier lugar y teniendo como fuente principal a la Ingeniería de Software.

- *Universidad Tecnológica del Perú*. Ofrece el programa de Ingeniería del Software, cuyo egresado está en capacidad de analizar, diseñar y desarrollar productos de software en general, proponiendo la plataforma tecnológica más apropiada. Domina los lenguajes y técnicas de programación, así como las metodologías de desarrollo de software, incluyendo la especificación de requisitos, el análisis y el diseño, las pruebas, la configuración, el mantenimiento y la documentación. Participa en proyectos de diseño y desarrollo de software, aplicando metodologías y estándares internacionales que le permitan obtener productos de alta calidad. Realiza investigaciones aplicadas en el campo del diseño de software, orientadas a la innovación

y la generación de soluciones para satisfacer necesidades presentes y futuras. El programa tiene una duración de doce semestres.

2.3.1 Reglamentaciones relacionadas con la Ingeniería de Software

- *NTP-RT-ISO/IEC 29110:5-1-2:2012*. Ingeniería de Software. Perfiles del ciclo de vida para las pequeñas organizaciones (PO). Parte 5-1-2: Guía de gestión e ingeniería: Grupo de perfil genérico. Perfil básico.
- *NTP-RT-ISO/IEC 29110-1:2012*. Ingeniería de Software. Perfiles del ciclo de vida para las pequeñas organizaciones (PO). Parte 1: Visión general.
- *NTP-RT-ISO/IEC 29110-3:2012*. Ingeniería de Software. Perfiles del ciclo de vida para las pequeñas organizaciones (PO). Parte 3: Evaluación de procesos.
- *NTP-ISO/IEC 20000-1:2012*. Tecnología de la Información. Gestión del servicio. Parte 1: Requisitos del sistema de gestión del servicio. 2da Edición.
- *NTP-ISO/IEC 20003:2011*: Tecnología de la información. Gestión del servicio. Parte 3: Guía sobre la definición del alcance y la aplicabilidad de la ISO/IEC 20000-1
- *NTP-ISO/IEC 15504-1:2011*: Tecnología de Información: Evaluación de procesos. Parte 1. Conceptos y vocabulario.
- *NTP-ISO/IEC 15504-2:2011*: Tecnología de Información: Evaluación de procesos. Parte 2. Evaluación de Procesos.
- *NTP-ISO/IEC 38500:2010*: Tecnología de Información: Gobierno corporativo de la tecnología de la información.
- *NTP-291.100-1:2009*: Tecnología de la Información. Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 1: Definición de conceptos y productos.
- *NTP-291.100-2:2009*: Tecnología de la Información. Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 2: Requisitos de procesos (MoProSoft).
- *NTP-291.100-3:2009*: Tecnología de la Información. Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 3: Guía de implantación de procesos.
- *NTP-291.100-4:2009*: Tecnología de la Información. Modelos de procesos y evaluación para desarrollo y mantenimiento de software. Parte 4: Directrices para el evaluación de procesos (EvalProSoft).
- *NTP-ISO/IEC 20000-1:2008*: Tecnología de la Información. Gestión del servicio. Parte 1: Especificaciones.
- *NTP-ISO/IEC 20000-2:2008*: Tecnología de la información. Gestión del servicio. Parte 2: Código de buenas prácticas.
- *NTP-ISO/IEC 90003:2008*: Ingeniería de Software. Guía de la aplicación de la ISO 9001:2000 al software
- *NTP-ISO/IEC 14598-6:2008*: Ingeniería de Software. Evaluación del producto. Parte 6: Documentación de módulos de evaluación.
- *NTP-ISO/IEC 14598-5:2007*: Ingeniería de Software. Evaluación del producto. Parte 5: Procesos para evaluadores.
- *NTP-ISO/IEC 15271:2007*: Tecnología de Información. Guía para la NTP-ISO/IEC 12207 (Proceso del Ciclo de vida del Software).
- *NTP-ISO/IEC 14598-4:2006*: Ingeniería de Software. Evaluación del producto. Parte 4: Procesos para adquirientes.
- *NTP-ISO/IEC 12207:2006*: Tecnología de la información. Procesos del ciclo de vida del software (Reglamento Técnico).

- *NTP-ISO/IEC 16326:2006*: Ingeniería de Software. Guía para la aplicación de la NTP-ISO/IEC 12207 para la gestión de proyectos.
- *NTP-ISO/IEC 9126-3:2005*: Ingeniería de software. Calidad del producto. Parte 3: Métricas internas.
- *NTP-ISO/IEC 9126-4:2005*: Ingeniería de software. Calidad del producto. Parte 4: Métricas de calidad en uso.
- *NTP-ISO/IEC 14598-1:2005*: Tecnología de la información - Evaluación de producto software. Parte 1: Visión General.
- *NTP-ISO/IEC 14598-2:2005*: Ingeniería de software. Evaluación del producto. Parte 2: Planificación y gestión.
- *NTP-ISO/IEC 14598-3:2005*: Ingeniería de software. Evaluación del producto. Parte 3: Proceso para desarrolladores.
- *NTP-ISO/IEC 12119:2005*: Tecnología de la Información. Paquetes Software. Requerimientos de calidad y pruebas.
- *NTP-ISO/IEC 12207:2004*: Tecnología de la información. Procesos del ciclo de vida del software. (Reemplazada por NTP-ISO/IEC 12207:2006).
- *NTP-ISO/IEC 9126-1:2004*: Ingeniería de software. Calidad del producto. Parte 1: Modelo de calidad.
- *NTP-ISO/IEC 9126-2:2004*: Ingeniería de software. Calidad del producto. Parte 2: Métricas externas.

Algunas de las industrias que se dedican a la Ingeniería de Software en Perú son:

- Tauro Soluciones: Desarrollo de sistemas a medida
- ABI Sistemas: Desarrollo de software
- ApolloSystems: Desarrollo de software para empresas
- Business Quality Solutions S.R.L: Software para empresas
- Data Análisis: Desarrollo de Sistemas
- Premium Soft: Desarrollo de software

2.4 BOLIVIA

- *Universidad Mayor de San Simón.* Ofrece los siguientes programas:

  - *Ingeniería en Informática*. Cuyos egresados están capacitados para analizar problemas informáticos en el nivel de abstracción adecuado, para de este modo identificar las entidades que lo forman y qué papel juega cada una de ellas, para luego formalizar el problema y evaluar posibles alternativas para desarrollar e implantar la solución más adecuada. Además, son capaces de planificar y gestionar proyectos informáticos, con conocimiento actualizado del mercado, así como del código ético profesional y los aspectos legales en el entorno de las tecnologías de la información. El programa tiene una duración de nueve semestres, y ofrece las líneas de especialización: 1) Bases de Datos, 2) Tecnologías de Lenguajes de programación, 3) Ingeniería de Software y 4) Inteligencia Artificial.

  - *Licenciatura en Ingeniería de Sistemas*. Que prepara profesionales para hacer un mejor uso de los recursos de tecnología Informática y de administración de los mismos, y sus actividades están relacionadas con la organización de equipos de desarrollo de software, administración de centros de cómputo tanto en tecnología como en personal, buscando una optimización en la

administración de dichos recursos. Poseen la capacidad de introducir, adecuar y desarrollar tecnologías de Información, acorde a las necesidades de los diferentes sectores. El programa tiene una duración de diez semestres y se oferta en las mismas líneas de especialización de la Ingeniería en Informática.

- *Universidad Católica Boliviana San Pablo.* Ofrece el programa de Ingeniería de Sistemas, y sus egresados serán capaces de brindarles a las organizaciones soluciones TIC para sus Sistemas de Información, desde perspectivas como el modelado y desarrollo de soluciones empresariales, aplicando Tecnologías de Información y Redes de Computadores; desarrollo de software de alta calidad para una amplia variedad de ámbitos; desarrollo de soluciones sistémicas, en ámbitos como los Sistemas de Información Geográficos, la Computación móvil y otros; aplicación de Tecnologías de Seguridad en Sistemas de Información o empleo de Tecnologías de Inteligencia Artificial para el apoyo a la Toma de Decisiones. El programa tiene una duración de diez semestres y el énfasis es la Ingeniería de Software, para lo cual firmaron en 2011, con la Cámara Nacional de Comercio, un convenio para la promoción y mejoramiento de la industria del software en Bolivia.

- *Universidad Privada Boliviana.* Ofrece el programa de Ingeniería de Sistemas Computacionales. La constante evolución tecnológica implica la necesidad en las organizaciones de contar con profesionales de alto nivel capaces de incorporar y desarrollar las variables tecnológicas eficientemente, por lo que estos profesionales enfocan sus estudios en campos relacionados con la Administración de Sistemas y la Ingeniería del Software, recibiendo una sólida base multidisciplinaria para desarrollar actividades en proyectos múltiples y en la obtención de Certificaciones Profesionales. El programa tiene una duración de nueve semestres, y los estudiantes pueden elegir áreas de estudio como: 1) Programación en Sistemas Informáticos, 2) Inteligencia Artificial, 3) Administración y Desarrollo de Sistemas de Información, 4) Teleinformática o 5) Generación de Empresas.

- *Universidad NUR*. Ofrece el programa de Ingeniería de Sistemas, mediante el cual forma profesionales utilizando un enfoque sistémico, que les permita aprovechar potencialmente los recursos tecnológicos en todos los procesos organizacionales referidos a la gestión computacional de información. Los egresados serán capaces de construir programas computacionales de gestión empresarial utilizando la Ingeniería de Sistemas; de aplicar modelos, métodos y técnicas de Ingeniería de Software para administrar los procesos de desarrollo tecnológico, y de dirigir equipos de desarrollo de software. El programa tiene una duración de diez semestres, y su campo laboral cubre áreas como gerente de proyectos de desarrollo de software, arquitecto de software, consultor de proyectos de desarrollo de software, ingeniero de software, entre otros.

- *Universidad Privada de Santa Cruz de la Sierra*. Ofrece el programa de Ingeniería de Sistemas, cuyos profesionales se capacitan para: 1) Desarrollar y modelar sistemas informáticos, optimización de sistemas y diseño de redes de computadores, 2) Manejar proyectos de software y desarrollar software con normas y estándares de calidad y 3) Estar capacitados para asimilar los cambios de tecnología informática. El programa tiene una duración de diez semestres, y el Ingeniero de Sistemas se podrá desempeñar en las empresas de servicio y manufactura prestando servicios de ingeniería para planificación, desarrollo y mantenimiento de software. Su perfil se orienta a:

  - Modelar y desarrollar sistemas usando la tecnología adecuada
  - Diseñar sistemas basados en redes de computadores

- Diseñar y construir software para sistemas específicos
- Participar en equipos profesionales multidisciplinarios de diseño y construcción de sistemas de software específico
- Mejorar la calidad de los productos software
- Desarrollar y aplicar nuevas metodologías para la construcción de sistemas de software específicos.

2.4.1 Reglamentaciones relacionadas con la Ingeniería de Software

- *NB/ISO/IEC 90003:2005*: Ingeniería del Software - Guía de aplicación de la norma ISO 9001:2000 al software (Correspondiente a la norma ISO-IEC 90003:2004).
- *NB/ISO/IEC 9126-1:2007*: Ingeniería del software - Calidad del producto de software - Parte 1: Modelo de calidad (Correspondiente a la norma ISO/IEC 9126-1:2001).
- *NB/ISO/IEC/TR 9126-2:2010*: Ingeniería de software - Calidad del producto - Parte 2: Métricas externas (Correspondiente a la norma ISO 9126-2:2003).
- *NB/ISO/IEC 14598-1:2006*: Tecnología de la información - Evaluación del producto de software - Parte 1: Visión general (Correspondiente a la norma ISO-IEC 14598-1:1999).
- *NB/ISO/IEC 14598-2:2007*: Ingeniería del software - Evaluación del producto de software - Parte 2: Planificación y gestión (Correspondiente a la norma ISO/IEC 14598-2:2000)... hasta la parte 6.

El organismo que regula la profesión de Ingeniería en Software en Bolivia es el Instituto Boliviano de Normalización y Calidad (IBNORCA), una institución privada, sin fines de lucro, que desde 1993 promueve la cultura de la calidad en país, a través de la normalización técnica, y la capacitación y certificación de productos y/o sistemas de gestión en las organizaciones. Algunas de las industrias en esta área son:

- TrueSoft: Investigación y desarrollo en software
- Intersoft: Desarrollo de aplicaciones, control de calidad (en software) y consultoría
- SIAP Software: Diseño y desarrollo de software
- Artexacta SRL: Desarrollo de software a la medida y Outsourcing y Offshoring
- Sol Radiante: Desarrollo de software a la medida
- Koala-Soft: Diseño y desarrollo de aplicaciones y páginas web

2.5 COLOMBIA

- *Universidad de los Andes*. Ofrece el programa de Ingeniería de Sistemas y Computación, con una duración de ocho semestres. Esta es una profesión dedicada a crear y construir soluciones informáticas que beneficien a la sociedad, para lo cual forma excelentes profesionales comprometidos con el país y con sus problemas, preparados para la competencia en un mundo globalizado; recursivos, con iniciativa e ingenio, que son capaces de adaptar y hacer transferencia tecnológica. El ingeniero de sistemas se mueve, sobre todo, en tres grandes áreas de conocimiento: 1) Construcción de software, 2) Tecnologías de información y 3) Sistemas de información. El egresado del perfil construcción de software tiene oportunidades a nivel mundial, donde la demanda de desarrolladores es alta, y donde las organizaciones que desarrollan software encuentran dificultades cada vez mayores para contratar ingenieros capacitados para hacer desarrollo de software de buena calidad. El grupo de investigación en Tecnologías de Información y Construcción de Software (TICSw) tiene tres líneas de trabajo: 1) Construcción de Software, 2) Métodos Formales y 3) Tecnologías de Información y Comunicaciones.

- *Pontificia Universidad Javeriana*. Ofrece el programa de Ingeniería de Sistemas en diez semestres. Los ingenieros de sistemas pueden crear soluciones que mejoren la calidad de vida de las personas y las organizaciones. Para ello la Universidad ofrece una formación sólida en sistemas de información, Ingeniería de Software y Ciencias Computacionales, utilizando las Tecnologías de Información y Comunicaciones. El ingeniero de sistemas es un agente de cambio que debe constituirse en un verdadero integrador de la tecnología y las organizaciones, teniendo como eje central a las personas y su calidad de vida, acorde con los retos presentes y futuros de la disciplina. Uno de los énfasis de la carrera es el área de Arquitecturas y Construcción de Software, donde el egresado se capacita para dirigir, diseñar y construir aplicaciones informáticas complejas, compuestas por módulos de software que se comunican entre sí, para que de manera integrada puedan contribuir en el cumplimiento de los objetivos generales de una organización. Las áreas de actuación del grupo Sistemas de Información e Ingeniería de Software (ISTAR) son: 1) Arquitectura Empresarial, 2) Ingeniería de Software, 3) Gestión de Información y 4) Sistemas de Información Geográfica.

- *Universidad Distrital Francisco José de Caldas*. Ofrece el programa de Ingeniería de Sistemas en diez semestres, y sus egresados son profesionales con capacidad para trabajar exitosamente en equipos multi-disciplinarios, inter-disciplinarios y/o trans-disciplinarios de investigación y/o desarrollo de empresa en Ingeniería de Sistemas; su formación es cimentada en una sólida sinergia entre las disciplinas de Ciencias Computacionales, Ingeniería de Software y las áreas de énfasis elegidas de las ofrecidas por el diseño curricular

- *Universidad del Valle*. Ofrece el programa de Ingeniería de Sistemas, con una duración de diez semestres. El egresado es un ingeniero emprendedor con profundo conocimiento de las ciencias y las tecnologías de la computación, con capacidad para liderar proyectos de desarrollo de software a gran escala, con criterios de calidad internacional. El Centro de Desarrollo de Software (CEDESOFT) está dedicado a la investigación y al desarrollo en el área de Ingeniería de Software, Ingeniería Web, Software Educativo, y en general al uso y aplicación de las TIC.

- *Universidad de Nariño*. Ofrece el programa de Ingeniería de Sistemas en diez semestres. El egresado es un profesional integral, agente liderador de cambio, con una sólida estructuración científico técnica, capaz de analizar, diseñar, desarrollar, implantar y controlar sistemas telemáticos, de gestión empresarial, educativos computacionales y sistemas basados en el conocimiento. Entre otras cosas, está capacitado para gerencia o crear empresas de servicios informáticos, producción de software y comercialización de equipos y suministros computacionales. La Universidad, como generadora de tecnología y conocimiento, adquiere con la sociedad, los gobiernos, la industria y la academia, el compromiso de dotarlas de profesionales que garanticen la construcción de software de alta calidad, que satisfaga los requerimientos modernos de administración de la información. El Grupo de Investigación Aplicado en Sistemas (GRIAS) tiene como objetivo desarrollar software encaminado a solucionar problemas del entorno, a través de sus líneas de Desarrollo de Software y Descubrimiento de Conocimiento en Bases de Datos.

- *Universidad EAFIT*. Ofrece el programa de Ingeniería de Sistemas en nueve semestres. El egresado es una persona capacitada para promover y gerenciar proyectos informáticos que lleven a las organizaciones la modernización y competitividad, y es capaz de proponer, modelar, adaptar, diseñar, construir, evaluar, auditar y mantener soluciones informáticas. Tiene entre sus líneas de énfasis a la de Desarrollo de software. El Grupo de Investigación Desarrollo e Innovación en

Tecnologías de la Información y las Comunicaciones (I+D+i en TIC), tiene entre sus líneas de investigación a la Ingeniería de Software, y trabaja para generar metodologías para el fortalecimiento de la calidad en el desarrollo de software.

- *Universidad Pontificia Bolivariana.* Ofrece en diez semestres el programa de Ingeniería de Sistemas e Informática. El programa busca formar profesionales idóneos en el Área de Ingeniería de Sistemas e Informática, con criterio científico, técnico, empresarial y humanista, capaces de diseñar, implementar, soportar y gestionar soluciones en el campo de las Tecnologías de Información, Comunicación y Computación, para satisfacer las necesidades organizacionales y sociales en el ámbito nacional e internacional. El egresado es un profesional con sólidas bases científicas, técnicas, empresariales y humanistas, que le permitan desempeñarse competentemente en diferentes campos, como el análisis, diseño y desarrollo de software, la telemática y las telecomunicaciones y la gestión de proyectos de tecnología. Está capacitado para diseñar, desarrollar e implementar software que ayude a la gestión del conocimiento en la organización que le permita aumentar su ventaja competitiva. El énfasis del programa es el Desarrollo de Software, que soporta a través del trabajo del Grupo de Investigación en Desarrollo y Aplicación en Telecomunicaciones e Informática (GIDATI).

- *Universidad de San Buenaventura*. Ofrece el programa de Ingeniería de Sistemas, con una duración de diez semestres, y orientado a formar profesionales que solucionen de forma óptima problemas de procesamiento de información, enfocándose en objetos de estudio estratégicos como la Construcción de Software. El egresado está en capacidad de diseñar, construir, implementar y administrar, de forma segura y confiable, soluciones informáticas a través del uso de las TIC, y de diseñar e implementar sistemas de información para satisfacer necesidades específicas en un campo de aplicación. Identifica las necesidades de negocio para el diseño y desarrollo de soluciones informáticas. Una de las líneas de énfasis es la Ingeniería de Software. El Grupo Laboratorio de Investigaciones para el Desarrollo de la Ingeniería de Software (LIDIS), busca el afianzamiento de todos los procesos de investigación formativa realizados en el programa, y la conformación de una serie de nuevos procesos de investigación con base disciplinar en el área específica del Desarrollo de Software. El área de Ingeniería de Software fue considerada como tema central del Grupo, debido al carácter estratégico del mismo en los planes de desarrollo a nivel nacional, y en los procesos de docencia y proyección social que adelanta el programa.

2.5.1 Reglamentaciones relacionadas con la Ingeniería de Software

- Ley 23 de enero 28 de 1982: Sobre derechos de autor
- Ley 44 de febrero 5 de 1993: Por la cual se modifica y adiciona la Ley 23 de 1982
- Ley 842 de octubre 14 de 2003: Por la cual se modifica la reglamentación del ejercicio de la ingeniería, de sus profesiones afines y de sus profesiones auxiliares, se adopta el Código de Ética Profesional y se dictan otras disposiciones
- Norma Técnica Colombiana NTC 5420-1 de 2006: Ingeniería del software. Parte 1: Modelo de calidad
- Norma Técnica Colombiana NTC 5420-2 de 2007: Ingeniería de software. Calidad del producto de software. Parte 2: Métricas externas
- Norma Técnica NTC-ISO/IEC Colombiana 90003 de 2005: Ingeniería de Software. Directrices para la aplicación de la NTC-ISO 9001:2000 a software de computador

Algunos organismos encargados de regular la profesión en Colombia son: Consejo Profesional Nacional de Ingeniería (COPNIA), Asociación Colombiana de Ingenieros de Sistemas (ACIS), Federación Colombiana de la Industria TI (FEDESOFT), Asociación Colombiana de Facultades de Ingeniería (ACOFI), y Red de Decanos y Directores de Ingeniería de Sistemas y Afines (REDIS). A continuación se detallan algunas de las industrias relacionadas con el área:

- MVM Ingeniería de Software: Desarrollo de Software a la Medida, Fábrica de Software, Gestión de aplicaciones, Taller de Software, Inteligencia de Negocios.

- Intergrupo: Desarrollo de aplicaciones móviles, Desarrollo a la medida, Fábrica de software, Mantenimiento de software, Desarrollo de soluciones móviles.

- Personal SOFT: Ingeniería de procesos, Ingeniería de Requisitos, Ingeniería de Software, Diseño y Construcción de Software, Testing de Aplicaciones.

- Ceiba Software: Integración y colaboración, Arquitectura y aplicaciones, Soluciones Móviles, Inteligencia de Negocios.

- PSL: Análisis, diseño, desarrollo, implantación y soporte de software por encargo a la medida de las necesidades de nuestros clientes.

- Choucair Testing: Software Testing.

3. CENTROAMÉRICA

3.1 COSTA RICA

- *Universidad Nacional de Costa Rica*. Ofrece el programa de Ingeniería en Sistemas de Información, con una duración de ocho semestres. Esta ingeniería se refiere al proceso de examinar una situación con la intención de mejorarla mediante nuevos procedimientos y métodos, en los que normalmente se utilizan sistemas computacionales para alcanzar los objetivos propuestos. Entre sus ejes curriculares se encuentra el de Ingeniería de Software con altos niveles de calidad, en el que cada uno de los proyectos e investigaciones que realicen los estudiantes debe prevalecer como meta, además de la eficacia y eficiencia, la calidad de software. El egresado se puede desempeñar como Programador de aplicaciones informáticas o Programador de aplicaciones Web. Perfiles en los cuales podrá desarrollar software según los requerimientos de la organización, documentar e implementar el software producido y darle mantenimiento.

- *Universidad Latina de Costa Rica*. Ofrece los siguientes programas:

  - *Ingeniería de Sistemas Informáticos,* con una duración de ocho semestres. Tiene como propósito la formación de profesionales en computación e informática para el desarrollo y adaptación de tecnología de información en la industria y los servicios, tanto en el sector público como privado, nacional e internacional. Entre otras, los egresados están en capacidad de desarrollar aplicaciones para las funciones organizacionales, programar aplicaciones computacionales con lenguajes y metodologías de última generación, y de diseñar, desarrollar y evolucionar sistemas de información que les permitan a las organizaciones alcanzar objetivos estratégicos

y obtener ventajas competitivas haciendo uso de mejores prácticas que garanticen la calidad. El perfil del programa se orienta al desarrollo, implementación, evaluación y mantenimiento de aplicaciones software y sistemas informáticos, acordes a las necesidades del entorno.

- *Licenciatura en Desarrollo de Software* en cuatro años, y que tiene como objetivo general especializar a profesionales informáticos en el desarrollo de software mediante la aplicación de métodos, técnicas y tecnologías modernos, de manera que puedan desempeñarse eficazmente en empresas que buscan desarrollar económicamente software con estándares de calidad mundial. Los egresados están en capacidad de sistematizar los requisitos para el desarrollo de software, diseñar sistemas software (para ello dispondrá de los principios y técnicas de diseño basados en componentes y patrones para construir software de alta calidad en los niveles conceptual y detallado), desarrollar sistemas Web, aplicar los procesos de Ingeniería de Software en proyectos de desarrollo de software, con atención a la calidad, los riesgos y los recursos, construir sistemas modulares en arquitecturas basadas en componentes, diseñar un anteproyecto de investigación en el área de desarrollo de software, diseñar productos software específicos para dispositivos móviles, asegurar la calidad de un producto software.

- *Universidad Estatal a Distancia*. Ofrece el programa de Ingeniería Informática y Calidad del Software en ocho semestres, y viene a satisfacer las necesidades de la Industria del Desarrollo de Software, que enfrenta la expansión a nivel internacional, y donde la calidad de los productos debe ser evidenciada en modelos que permitan certificarla. Por tanto, esta licenciatura prepara profesionales que puedan diseñar sistemas de calidad en desarrollo de software, que garanticen o certifiquen el proceso de la industria de software, y les permita a las organizaciones ser más competitivas en el mercado nacional e internacional. El egresado es un profesional con habilidades para la implementación de modelos de calidad en organizaciones orientadas al desarrollo de software, y están en capacidad de laborar en empresas privadas, públicas o propias, que cuenten con un área dedicada al desarrollo de software. Podrá desempeñarse en puestos relacionados con el proceso de sistemas de calidad en software, como gestor de la calidad, líder de proyectos de certificación de la calidad, gestor de procedimientos, normas y métricas de calidad.

- *Universidad Latinoamericana de Ciencia y Tecnología*. Ofrece el programa de Licenciatura en Ingeniería Informática con Énfasis en Desarrollo de Software, con una duración de once semestres. Estos ingenieros aplican los principios y técnicas de la computación, la ingeniería y el análisis matemático al diseño, desarrollo, medición y evaluación del software y de los sistemas que les permiten a los computadores realizar sus múltiples aplicaciones; analizan los requisitos de los usuarios, y luego los construyen, evalúan y dan mantenimiento. Los egresados están preparados para diseñar software para diversos sistemas operativos, y desarrollan amplias competencias en las metodologías de programación, pero el énfasis yace más en el desarrollo de algoritmos y en el análisis y resolución de problemas de programación, que propiamente en la escritura de código. Entre otros, los perfiles de estos profesionales son:

  - Director de un departamento de desarrollo de software
  - Desarrollador de plataformas de comercio electrónico
  - Gerente de proyectos software
  - Director de un departamento de investigación y desarrollo

3.1.1 Reglamentaciones relacionadas con la Ingeniería de Software

- Ley General de Aduanas, Nº 8373: artículo 104
- Ley de Contratación Administrativa, artículo 40
- Código de Normas y Procedimientos Tributarios, Nº4755: artículo 122
- Ley de Certificados, Firmas Digitales y Documentos Electrónicos, Nº 8454
- Ley del Sistema Nacional de Archivos, Nº 7202
- Decreto Ejecutivo Nº 31344-MAG: que "Declara de Interés Público el Desarrollo de INFOAGRO"
- Decreto Ejecutivo Nº 30146-H: que reconoce el débito electrónico como medio de pago de impuestos
- Decreto Ejecutivo Nº 30151-J: que ordena a las instituciones del Gobierno a prevenir y combatir el uso ilegal de programas de cómputo
- Decreto Ejecutivo Nº 25116-MP-MICIT: que crea la "Red Gubernamental GOBNet"
- Resolución Nº 29-01: para enviar y pagar declaraciones tributarias a través de Internet
- Decreto Ejecutivo Nº 30628-MICIT: que estableció el estatuto de la Academia Nacional de Ciencias
- Decreto Ejecutivo Nº 32596: que crea una Subcomisión Técnica de Indicadores para las Tecnologías de la Información y la Comunicación
- Ley N° 6683: de Derechos de Autor y Derechos Conexos (Reformada mediante las leyes N° 6935 y N° 7397)
- Ley Nº 7951: de Protección a los Sistemas de Trazados de los Circuitos Integrados
- Ley Nº 8039: de Procedimientos de Observancia de los Derechos de Propiedad Intelectual Reglamento Ejecutivo Nº 24611-J, Decreto Ejecutivo N° 30151-J
- Expediente N° 15397, proyecto de Ley de Delitos Informáticos

La industria del software en Costa Rica no tiene más de 30 años, y ha tenido un desarrollo creciente en la última década, debido especialmente a factores históricos y culturales que han sido fundamentales para el crecimiento, posicionamiento y éxito del sector. En el país se han identificado alrededor de 450 empresas desarrolladoras de software, que por su alta calidad y funcionalidad han convertido al país en apto para el posicionamiento de la industria en beneficio del desarrollo. Las entidades relacionadas con la industria del software en país son:

- Ministerio de Ciencia y Tecnología (MICIT). Que promueve, incentiva y estimula la creación de condiciones apropiadas para la investigación, la innovación, el conocimiento y el desarrollo tecnológico del país.

- Consejo Nacional para Investigaciones Científicas y Tecnológicas (CONICIT). Institución autónoma con personalidad jurídica y patrimonio propios. Su función es promover el desarrollo de las ciencias y de la tecnología, para fines pacíficos, por medio de la investigación sistematizada o del acto creador.

- Cámara de Empresas de Tecnología de Información y Comunicaciones (CAMTIC). Está integrada por cerca de 200 empresas, que conforman un bloque estratégico que busca fortalecer y apoyar al sector de las tecnologías digitales en el país.

3.2 CUBA

- *Universidad de La Habana*. Ofrece el programa de Ciencias de la Computación en diez semestres. La relevancia que tenga o se le asigne a la dimensión computacional de un problema interdisciplinario y el grado en que se pretenda o sea factible computarizar su solución,

determinan la esencialidad o no del problema como problema computacional y, por ende, el grado de participación del profesional en Ciencias de la Computación. Por lo tanto, este profesional se forma para resolver los problemas propios de su profesión, y para participar en la solución de problemas interdisciplinarios, las cuales son las esferas de su actuación. El egresado está capacitado para desarrollar, aplicar y facilitar el uso de la capacidad potencial de los computadores para los procesos de información, mediante la creación de software que permita la realización y ejecución eficiente de procedimientos de solución de problemas, facilite la comunicación con los recursos computacionales y su manipulación, facilite las tareas de programación y la creación de interfaces adecuadas para programas y el almacenamiento y la recuperación, así como las actualizaciones pertinentes de información. La investigación del programa se manifiesta en la línea Nuevas Tecnologías de la Información y las Comunicaciones, a través de la que se desarrollan espacios virtuales de aprendizaje mediante plataformas de tele-formación, software de redes sociales, etc., y la elaboración de recursos educativos abiertos exportables a estándares, que permitan la introducción de procesos de innovación para el desarrollo de los programas educativos.

- *Instituto Superior Politécnico José Antonio Echeverría*. Ofrece el programa de Ingeniería Informática en diez semestres. Se forma a profesionales que tienen como objeto el desarrollo de sistemas informáticos, con sólida formación técnica y tecnológica, que se ocupan de los procesos de captación, transmisión, almacenamiento, tratamiento y presentación de la información, mediante el uso eficiente de los computadores y otros medios técnicos. Los modos de actuación del ingeniero informático están asociados con los procesos relacionados con el desarrollo y explotación de un sistema informático, así como la autogestión del aprendizaje, en correspondencia con el carácter sistemático de los avances en la tecnología informática. Su esfera de actuación comprende los procesos del ciclo de vida del sistema informático, la explotación de sistemas y las herramientas de desarrollo, desempeñando diferentes roles en el equipo de trabajo, así como la gestión del conocimiento y la capacitación. La investigación se orienta a través de la línea de Ingeniería de Software, en áreas como las metodologías de desarrollo de software, la calidad del software, las pruebas, la Ingeniería de Requisitos, el trabajo en equipos de desarrollo, los almacenes de datos, y la Inteligencia Artificial aplicada a la Ingeniería de Software.

- *Universidad de las Ciencias Informáticas*. Ofrece el programa de Ingeniería en Ciencias Informáticas en diez semestres, y tiene como objeto de trabajo el ciclo de vida del software, con una perspectiva industrial; aplicado a los procesos de tratamiento y gestión de la información y del conocimiento en organizaciones productivas y de servicios, con el objetivo de incrementar la eficacia, la eficiencia y la competitividad en su funcionamiento. Los egresados están capacitados para ejecutar los diferentes roles asociados a la proyección, construcción y mantenimiento de software, tanto en empresas de producción industrial como en otras organizaciones que desarrollen sus propios programas. Para lograrlo, durante su proceso de formación debe desarrollar competencias profesionales para proyectar, construir y mantener software, aplicando utilizando metodologías, métodos, técnicas y herramientas apropiadas de la Ingeniería de Software. La investigación relacionada se realiza desde el grupo de Ingeniería y Calidad del Software y de la línea homónima, y se difunde a través de la Revista Cubana de Ciencias Informáticas (RCCI).

- *Universidad de Pinar del Rio*. Ofrece el programa de Ingeniería Informática en diez semestres. Los egresados son profesionales que se insertan de manera multidisciplinaria con especialistas de

diversas ramas, para concebir y desarrollar soluciones informáticas, que brinden respuestas a las necesidades del problema en cuestión, siendo capaces de asimilar los modelos correspondientes, seleccionar y utilizar el equipamiento, técnicas y métodos más efectivos para el procesamiento de la información. Su campo de acción está asociado a la concepción, modelado, diseño, desarrollo, implantación, integración, mantenimiento y prueba de sistemas informáticos, y desarrollan habilidades en Ingeniería de Software, técnicas de programación, tecnología asociada al funcionamiento de los medios de cómputo y de comunicaciones, Inteligencia Artificial, métodos matemáticos y otros espacios de aplicación de la informática.

3.2.1 Reglamentaciones relacionadas con la Ingeniería de Software

El Comité Ejecutivo del Consejo de Ministros [COM97] hizo el llamado para trazar los "*Lineamientos estratégicos para la informatización de la sociedad cubana*", en los cuales se aborda integralmente la reglamentación para garantizar normas claras para el desarrollo informático del Estado. En el año 2000 se transfirió, al Ministerio de Informática y Comunicaciones (MIC), la capacidad de tomar decisiones y de ejecución de la política estatal relacionada. La industria cubana del software tiene una historia relativamente corta, pues no fue sino hasta el 2004, en el marco de la X Convención y Feria Internacional Informática, que hizo su presentación oficialmente. Algunas empresas representativas son:

- Industria Cubana del Software (INCUSOFT). Creada con el objetivo de aunar los esfuerzos individuales que realizan diversas instituciones para alcanzar una fortaleza que permita incursionar en los mercados extranjeros.

- Centro Nacional de Calidad de Software (CALISOFT). Es una unidad presupuestada subordinada al Ministerio de la Informática y las Comunicaciones (MIC).

- Empresa Cubana del Sector de las TIC (CITMATEL). Proporciona una amplia gama de soluciones informáticas integrales, entre los que se incluyen servicios de consultoría, proyectos, asistencia técnica, venta e instalación del equipamiento, software, desarrollo de software, soluciones de conectividad, diseño de aplicaciones para Internet y para móviles (WAP, SMS), foros virtuales, cursos en línea y comercio electrónico. Provee servicios de Internet en todo el territorio cubano (ISP) y el registro de nombres de dominio .cu.

- Agencia de Negocios para la promoción de exportaciones de software, Productos y Servicios (AVANTE). Pertenece al Ministerio de la Informática y las Comunicaciones, y es la representación de INCUSOFT.

- Empresa Nacional de Software (DESOFT). Proporciona soluciones integrales en Tecnología de la Información para la Informatización de la Sociedad Cubana. Actualmente desarrolla proyectos internacionales y explora las posibilidades para nuevos contratos en el exterior.

- Empresa de Software de Calidad (SoftCal). Ofrece soluciones informáticas asociadas a la calidad como patrón determinante del resultado final del producto.

3.3 GUATEMALA

- *Universidad de San Carlos de Guatemala*. Ofrece el programa de Ingeniería en Ciencias de la Computación y Sistemas de Información en diez semestres. La visión de la carrera implica

capacitar a los estudiantes para identificar las oportunidades de mejoramiento, y poder aplicar los conceptos teóricos de manera creativa en el diseño, construcción e implementación de aplicaciones, que sean acordes a la situación nacional. A través de estas soluciones, desarrolladas junto a grupos multidisciplinarios de trabajo, los egresados podrán elevar el nivel tecnológico y productivo de las empresas y organizaciones en donde se desempeñen. El egresado tiene un perfil en el que aplica conocimientos de índole específica, referentes a las ciencias de su especialidad, que cubren las Ciencias Computacionales, la metodología de sistemas y el desarrollo de Sistemas de Información.

- *Universidad del Valle de Guatemala*. Ofrece la Ingeniería en Ciencias de la Computación y Tecnologías de la Información, en diez semestres. El egresado de esta licenciatura crea y administra sistemas computarizados, analizando las necesidades de los usuarios, y su formación profesional, a partir de las áreas de excelencia, le permite desempeñarse a nivel nacional e internacional como diseñador e implementador de software. Se refiere al trabajo en desarrollo de software, que ha ido creciendo hasta incluir aspectos de desarrollo web, diseño de interfaces, aspectos de seguridad, computación móvil y otros.

- *Universidad Rafael Landívar*. Ofrece el programa de Ingeniería en Informática y Sistemas en diez semestres. Este ingeniero se ocupa de los problemas relacionados con la información, como su captura, manejo, procesamiento, distribución, acceso y presentación, y además de los conocimientos propios del área, que comprenden un fuerte componente de técnicas de programación, manejo de archivos y bases de datos, redes de computadores y procesamiento distribuido, sistemas operativos, análisis y diseño de sistemas, arquitectura de computadores, programación en Internet, entre otras; se caracteriza por poseer sólidos conocimientos gerenciales, administrativos y financieros, que lo preparan para desenvolverse profesionalmente en un entorno caracterizado por premiar la iniciativa y el emprendimiento creativo.

- *Universidad Mariano Gálvez*. Ofrece el programa de Ingeniería en Sistemas y Ciencias de la Computación en diez semestres. Entre otros, el egresado está capacitado para dominar los fundamentos de los lenguajes más recientes de programación, utilizándolos como herramientas para la solución de problemas relacionados con el manejo de datos y con el desarrollo de Sistemas de Información. Para esto integra conocimientos de las Ciencias de la Ingeniería y la aplicación de Modelos y Herramientas de las Tecnologías de Información para el desarrollo y gestión de los Sistemas de Información.

3.3.1 Reglamentaciones relacionadas con la Ingeniería de Software

El sector de software en Guatemala posee escaso marco jurídico, constituido básicamente por la Ley de Derecho de Autor y Derechos Conexos, decreto de ley No. 33-98, y por la Ley de Propiedad Industrial, decreto de ley No. 57-2000. Pero no se encontraron normativas que regulen la Ingeniería de Software como tal, ni la exigencia de la norma ISO para el efecto (ISO/IEC 24773). La profesión se ejerce mediante el registro en el Colegio de Ingenieros de Guatemala, pero con la misma regulación de una ingeniería. En cuanto a la industria, la Comisión de Software de Exportación (SOFEX) agrupa las empresas dedicadas al desarrollo de Software, las cuales ofrecen productos y servicios altamente calificados, para atender el mercado local e internacional de manera competitiva e innovadora, por medio del talento y profesionalismo de su recurso humano. Algunas de esas empresas son:

- Aldea Systems
- Asesores en Informática (ASEINFO)

- Business Development Group, S.A. (BDG)
- Blanco Silva, Consultoría Informática
- Byte
- Canella
- Coinsa
- Districalc Corporation
- Business Software Solutions (GYSSA)
- e-Business Solutions (ICON)
- MegaSoluciones, S.A.
- Micro Finance Solutions Inc. (MFSI)
- Open Consult
- Sistemas de Adminitración Virtual (SAVSA)
- Grupo SEGA
- Soluciones Internacionales de Tecnología y Procesos, S.A. (SITECPRO)
- Software y Servicios de Automatización S.A. (SSASA)
- Strategic de Análisis de Centroamérica, S.A.
- Soluciones Informáticas y Empresariales, S.A. (VIA Asesores)

3.4 MÉXICO

- *Universidad Nacional Autónoma de México*. Ofrece la Ingeniería en Computación en diez semestres. Este ingeniero es un profesional de alto nivel científico y tecnológico, con conocimientos sólidos y generales que le permiten identificar, analizar, planear, diseñar, organizar, producir, operar y dar soporte a los sistemas electrónicos (Ingeniería de Hardware), para el procesamiento digital de datos y control de procesos a los sistemas de programación, tanto de base como de aplicación (Ingeniería de Software). Realiza actividades fundamentales en donde existan computadores o dispositivos de control automático, y se desempeña en los sectores público y privado, en organismos estatales, descentralizados, secretarías de Estado, o bien en instituciones dedicadas a la docencia y a la investigación. Su campo de trabajo incluye áreas como la Ingeniería de Software y Hardware y el diseño y construcción de software de entretenimiento. A través del Grupo de Gestión de Objetos —*Object Management Group* (OMG)— se generan herramientas de interés para los creadores de software. El grupo está conformado por compañías de varios países, las cuales establecen estándares internacionales de calidad de software, como protocolos o normas que deben cumplir los que crean, distribuyen y venden sistemas informáticos.

- *Universidad Autónoma Metropolitana*. Ofrece el programa de Ingeniería en Computación en diez semestres. El egresado estará en capacidad de Resolver problemas que requieran de la integración de software, hardware y redes, con el fin de contribuir al bienestar de la sociedad, y de aplicar sus conocimientos y habilidades en el análisis, diseño, desarrollo y mantenimiento de proyectos de computación, buscando el mejor aprovechamiento de los recursos.

- *Universidad de Guadalajara*. Ofrece la licenciatura en Informática en diez semestres. El egresado estará en capacidad de desarrollar sistemas y encontrar soluciones creativas e innovadoras para las necesidades que existan en sus lugares de trabajo. Para cumplir con la demanda ocupacional, deberá estar actualizado en la utilización de computadores, en el diseño de bases de datos y sistemas de redes, así como en la capacidad de elaborar *software* que resuelva diversas aplicaciones complejas, para lo cual debe involucrarse en las diferentes ramas de la ingeniería.

- *Instituto Politécnico Nacional*. Ofrece el programa de Ingeniería en Informática en ocho semestres. El egresado es un profesionista interdisciplinario que implementa y administra sistemas software de calidad mundial; proporciona soluciones de transmisión de voz y datos; aplica las metodologías de normalización y calidad en el proceso de desarrollo y administración de software y hardware garantizando su seguridad, y propone procesos planificados de innovación en el campo de la Ingeniería Informática a través de la investigación y el desarrollo de soluciones software y hardware, acordes con las necesidades actuales.

- *Universidad de Sonora*. Ofrece el programa Ingeniería en Sistemas de Información en ocho semestres. El egresado estará en capacidad de desarrollar software aplicando los métodos, modelos y estándares de calidad de la industria. Además, definir alcances, costos, tiempos, recursos y factibilidad de proyectos software, así como proponer soluciones integrales que permitan el control de los procesos organizacionales y de apoyo a la toma de decisiones, aplicando tecnologías de la información.

- *Universidad Autónoma de Nuevo León*. Ofrece los siguientes programas:

  - *Ingeniería en Tecnología de Software,* en diez semestres. Los egresados son profesionales con una formación amplia y sólida que les prepara para dirigir y realizar las tareas de todas las fases del ciclo de vida del software, y aplicaciones y productos que resuelvan problemas de cualquier ámbito de las industria, aplicando conocimientos científico, métodos y técnicas propios de la Ingeniería de Software en dispositivos móviles y en sistemas inteligentes. Es capaz de desenvolverse satisfactoriamente en diferentes áreas y compañías donde se requiera el manejo de metodologías de desarrollo de software, diseño y modelado de plataformas, sistemas integrados y de cómputo móvil, crear soluciones innovadoras de cómputo integrado para satisfacer las necesidades de los numerosos campos de aplicación de las tecnologías de la información. Está capacitado para dirigir, coordinar y llevar a cabo proyectos de desarrollo y mantenimiento de aplicaciones integradas y sistemas inteligentes, así como dominar todas las etapas de vida de un proyecto.

  - *Ingeniería en Administración de Sistemas,* en diez semestres. El egresado posee competencias para el desarrollo, integración y gestión de software para el sector industrial y de servicios, aplicando modelos y soluciones bajo estándares de calidad y seguridad, en un ambiente multidisciplinario, y con compromiso ético, profesional y humano. Es un profesionista competente en el desarrollo de aplicaciones software, que aporta y administra soluciones integrales e innovadoras para la toma de decisiones, promoviendo la investigación y el desarrollo tecnológico, y con una formación integral orientada a satisfacer las necesidades de la sociedad en el área de las tecnologías de la información.

3.4.1 Reglamentaciones relacionadas con la Ingeniería de Software

México tiene una inadecua legislación vigente acerca del software. Las políticas públicas, tendientes a estimular y facilitar el desarrollo están bien orientadas, pero su implantación es muy reciente, por lo que es difícil realizar juicios. El Programa para el Desarrollo de la Industria de Software (PDIS) es uno de los medios que el Estado utiliza para definir las líneas de acción. En cualquier caso, el país necesita mejorar sus ventajas competitivas para lograr los objetivos propuestos, pero el subsector del *software* todavía está lejos de alcanzar la madurez organizativa y tecnológica, a la vez que su mercado aún se encuentra en proceso de redefinición. Algunos organismos relacionados son:

- La Asociación Mexicana de la Industria de Tecnologías de Información (AMITI), es una organización privada creada para posicionar las TIC como motor clave para aumentar la competitividad de México, promoviendo el crecimiento de la industria mediante la búsqueda de un marco reglamentario, comercial y legal, que facilite el desarrollo de negocios. Por otro lado, la Ingeniería de Software ha sido promovida desde la década de los 90, y se han obtenido logros como la creación de la Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS), el modelo MoProSoft, el apoyo al programa PROSOFT, el soporte para la norma mexicana NMX-I-059/02-NYCE-2005 y la base para la norma ISO/IEC 29110.

- El Consejo Nacional de Ciencia y Tecnología (CONACYT), es un organismo público del Sector Educativo, con personalidad jurídica y patrimonio propio, y es responsable de elaborar las políticas de ciencia y tecnología en el país. Su misión es impulsar y fortalecer el desarrollo científico y la modernización tecnológica de México, mediante la formación de recurso humano de alto nivel, la promoción y el sostenimiento de proyectos específicos de investigación y la difusión de la información científica y tecnológica.

En México, las profesiones se ejercen mediante la obtención de una licencia o patente del ejercicio profesional, que otorga la Dirección General de Profesiones, dependiente de la Secretaria de Educación Pública, a aquellas personas que han acreditado el cumplimiento de los requisitos establecidos.

3.5 PANAMÁ

- *Universidad Tecnológica de Panamá*. Ofrece los siguientes programas:

  - *Ingeniería de Sistemas de Información,* en nueve semestres. Entre otras, los egresados estarán en capacidad de:

    - Diseñar y desarrollar sistemas computacionales
    - Construir, evaluar y seleccionar software basados en los requisitos del negocio
    - Desarrollar e implementar productos software en áreas como el comercio electrónico, CRM, ERP, Inteligencia de Negocios, SCM y otros.
    - Dirigir y desarrollar proyectos de investigación y aplicación en bases de datos, ingeniería de sistemas de información, Ingeniería de Software, seguridad, auditoría de sistemas, sistemas inteligentes y sistemas para soporte y toma de decisiones.

  - *Ingeniería de Sistemas y Computación,* en nueve semestres. Una Licenciatura con un fuerte matiz hacia el desarrollo de elementos informáticos originales y a la medida de una solución específica. Se fundamenta en una sólida base matemática y computacional que permite la formación de profesionales más críticos y orientados al desarrollo de soluciones innovadoras. El egresado estará en capacidad de diseñar y generar sistemas operativos e interfaces de software, y de desarrollar software de sistemas de control de dispositivos robóticos con determinado grado de inteligencia, diseñar sistemas digitales y el software de funcionamiento, entre otros.

  - *Licenciatura en Desarrollo de Software,* en ocho semestres. Esta Licenciatura busca suplir las necesidades de las organizaciones o empresas de Software de tipo Administrativo, Comercial, Financiero y de Mercadotecnia, entre otros. El egresado de la carrera estará preparado para:

- Dirigir un Centro de Desarrollo de Software
- Liderar proyectos de Desarrollo de Software
- Programar en nuevas Tecnologías que giran en torno a Internet y los negocios por medios electrónicos
- Asesorar en la producción de Software en las empresas
- Desarrollar aplicaciones implementando herramientas multimedia
- Generar empresas orientadas a la industria de Software

Estos programas sustentan su investigación en el grupo de Ingeniería de Software, que tiene como objetivo estudiar, aplicar y mejorar los modelos, métodos y técnicas actuales de desarrollo de sistemas, aplicando rigurosamente tecnología avanzada de Ingeniería de Software.

- *Universidad Latinoamericana de Ciencia y Tecnología*. Ofrece el programa de Ingeniera Informática con Énfasis en Desarrollo de Software, en ocho semestres. La carrera está orientada a la formación de profesionales de la ingeniería, especializados en implementar sus conocimientos en las áreas de mantenimiento de equipos, desarrollo, mantenimiento y evolución de software, además de la administración de redes de computadores. Los egresados estarán en capacidad de:

  - Implantar sistemas informáticos en las funciones empresariales básicas, finanzas, mercadeo, producción, compras, recursos humanos, etc.
  - Administrar y supervisar las operaciones de un centro informático
  - Participar en la toma de decisiones con respecto a la adquisición de hardware y software, así como la creación de nuevos sistemas y/o eliminación o modificación de los existentes
  - Diseñar estrategias con el fin de mejorar los flujos de información en las diferentes entidades de la organización
  - Administrar las bases de datos corporativas
  - Desarrollar aplicaciones en Internet
  - Dominar las técnicas de programación orientadas a objetos
  - Realizar soporte técnico y asesoría a empresas

- *Universidad del Istmo*. Ofrece los siguientes programas:

  - *Ingeniería en Sistemas* con énfasis en Seguridad Informática, en siete semestres. El egresado:

    - Tendrá conocimientos sobre la aplicación de técnicas de programación para la solución computarizada de problemas de tipo comercial o empresarial
    - Mostrará conocimiento sobre las instrucciones para programar en el lenguaje de programación C
    - Programará en un lenguaje de alto nivel que esté orientado a objetos
    - Recopilará y analizará los datos que se requieren para el análisis de requisitos, el diseño y el desarrollo de un Sistema de Información, o aplicación con los diversos controles de seguridad

  - *Ingeniería en Sistemas* con énfasis en Construcción de Aplicaciones, en siete semestres. El egresado tendrá conocimientos básicos para desarrollar, conceptual y técnicamente, la creación de aplicaciones en las empresas, y aplicará los métodos, técnicas y herramientas para analizar y diseñar un Sistema de Información.

- *Universidad de Cartago*. Ofrece la Licenciatura en Informática en once semestres. Los graduados se preparan para desarrollar programas de alto nivel, en la interfaces persona–máquina, capaces de manejar recursos físicos computacionales e interconexiones de diferentes máquinas simultáneamente, y para crear programas para incrementar la productividad de información en la parte de interfaces usuario–máquina, lenguajes más simples y potentes al alcance de más personas, programas generadores de aplicaciones que resuelvan un tipo genérico de problemas.

3.5.1 Reglamentaciones relacionadas con la Ingeniería de Software

- Resolución 253 del 5 de agosto de 1998, Gaceta Oficial 23,622 de 3 de septiembre de 1998. Por medio de la cual se reglamentan las funciones correspondientes a los títulos de Ingeniero Electrónico en Computadores, Ingeniero en Sistemas Electrónicos e Ingeniero de Computadores.

- Resolución 368 del 26 de marzo de 1999, Gaceta Oficial 23,771 de 9 de abril de 1999. Por medio de la cual se Reglamenta las funciones correspondientes al título de Ingeniero Administrador de Sistemas.
- Resolución 737 del 28 de febrero de 2007, Gaceta Oficial 25,872 de 7 de septiembre de 2007. Por medio de la cual se determinan las funciones correspondientes al título de Ingeniero de Sistemas.

- Resolución 738 del 28 de febrero de 2007, Gaceta Oficial 25,885 de 26 de septiembre de 2007. Por medio del cual se determinan las funciones correspondientes al título de Ingeniero de Sistemas y Computación.

- Resolución 761 del 13 de junio de 2007, Gaceta Oficial 25,888 de 1 de octubre de 2007. Por medio de la cual se determinan las funciones correspondientes al título de Licenciatura en Desarrollo de Software.

Algunas empresas relacionadas con el desarrollo de software en Panamá son:

- Apsys. Soluciones de Software a la medida y Outsourcing. Expertos en desarrollo de aplicaciones Web y dispositivos móviles, como Android, iPhone, Blackberry y Windows Phone.

- Bios Software. Diseño, desarrollo, asesoría e implementación de sistemas de información y redes para los sectores educativos, empresariales y legislativos.

- Byte. Provee soluciones de software de clase mundial para las industrias bancaria/financiera y de telecomunicaciones.

- Desimplex, S.A. Desarrollo de software financiero, de recursos humanos, entes reguladores, Base de Datos, BI y desarrollos a la medida.

- Logic Studio. Expertos en desarrollo móvil, software a la medida, outsourcing de IT y pruebas de software.

- SvSoftware S.A. DE C.V. Outsourcing en el desarrollo de aplicaciones software.

- Plus Technologies & Innovations. Empresa proveedora independiente de Software (ISV), que desarrolla y comercializa soluciones de aplicaciones de negocios de alta calidad.

REFERENCIAS

[BAU08] Baum, G. & Artopoulos, A. (2008). Libro Blanco de la Prospectiva TIC – Proyecto 2020. Ministerio de Ciencia, Tecnología e Innovación Productiva, Buenos Aires.

[COM97] Comité Ejecutivo del Consejo de Ministros (1997). Lineamientos estratégicos para la informatización de la Sociedad Cubana. Resumen Ejecutivo. La Habana, Cuba. Junio.

[DAV11] Dávila, M. & Martínez, L.E. (2011). "Evaluación y Acreditación en Argentina y Uruguay: los sistemas de educación superior y nuevas orientaciones de política en perspectiva comparada." Universidad de Belgrano, Buenos Aires.

[MCT11] MCTI. (2011). *"TI MAIOR – Programa estratégico de Software e Serviç*os de Tecnologia da Informa*ção 2012-2015."* Ministério da Ciência, Tecnologia e Inovação, Brasil.

[MER02] MERCOSUR. (2002). "Memorandum de entendimiento sobre la implementación de un mecanismo experimental de acreditación de carreras para el reconocimiento de títulos de grado universitario en los países del MERCOSUR, Bolivia y Chile." Online: http://www.coneau.gov.ar/archivos/files/MemorandumparalaCreaciondelSistemaARCUSUR.pdf. [May 2013].

[MIN11] Ministerio de Industria de la Nación. (2011). "Plan Estratégico Industrial 2020." Argentina.

[SEL09] SELA, Secretaría Permanente. (2009). "Desarrollo de la Industria Regional de Software en América Latina y el Caribe: Consideraciones y propuestas." Venezuela.

CAPÍTULO IV

RETOS Y REALIDADES DE LA INGENIERÍA DE SOFTWARE EN EL SIGLO XXI

Paola V. BRITOS[1]; Alejandro A. HOSSIAN[2]; Ramón GARCÍA-MARTÍNEZ[3];
Martha L. PALACIOS H.[4]; Leonel NOSSA O.[5];
[1] *Universidad Nacional de Río Negro*
[2] *Universidad Tecnológica Nacional*
[3] *Universidad Nacional de Lanús*
Argentina
[4,5] *Universidad Autónoma de Colombia*
Colombia

INTRODUCCIÓN

La presente exposición se sustenta en dos ejes fundamentales, a través de los que se aborda la temática propuesta para el desarrollo de este capítulo, y con base en dos enfoques que, a juicio de los autores, constituyen una perspectiva que le permitirá al lector formarse una idea global acerca de la realidad actual y de los futuros desafíos que debe afrontar la Ingeniería de Software en los albores de este milenio, para luego centrarse en el contexto de América Latina en particular.

El Eje 1 se desarrolla en el primer apartado del documento, y se denomina *Visión Global de la IS*. En él se presenta una breve síntesis de la Ingeniería de Software a nivel general, haciendo mención a sus orígenes y a como fueron surgiendo algunas de las dificultades de mayor peso para el desarrollo de la disciplina, que derivaron en lo que popularmente se conoce como *Crisis del Software*. Crisis que, en opinión de parte de la comunidad científica y tecnológica vinculada a la investigación y a la práctica de esta ingeniería, mantiene su carácter crónico y aún es vigente.

El Eje 2 se desarrolla en el segundo apartado del documento, y se denomina *Panorama de la Ingeniería de Software en América Latina.* En él se presenta una breve síntesis de esta ingeniería a nivel general, dentro del contexto de la realidad que se vive en la región. En tal sentido, también se esbozan algunas de las líneas de carácter estratégico que le pueden brindar solidez, como para enfrentar los retos que afrontará la disciplina en el corto y mediano plazo.

Posteriormente, se hace un análisis a las tendencias y a la prospectiva de la Ingeniería de Software en la Sociedad de la información y el Conocimiento. Se trata de una descripción del contexto en el que los profesionales de esta disciplina se desempeñarán en un futuro cercano, analizando la relación intrínseca entre esta ingeniería y la gestión del conocimiento. También se presenta la prospectiva del área, desde una visión de los retos y realidades de la sociedad, y su dependencia de los productos software.

1. VISIÓN GLOBAL DE LA INGENIERÍA DE SOFTWARE

Presentar una visión global de esta ingeniería como disciplina no resulta sencillo, debido a la cantidad de información que está disponible en las diversas fuentes bibliográficas, y a los diferentes enfoques a partir de los cuales se puede abordar la temática. En este sentido, se puede establecer un enfoque más histórico, centrado en las causas que le dieron origen en la década de 1960, o quizá abordarlo más centrado en las características de los productos software, que motivaron la necesidad de comprender su construcción como un proceso ingenieril, que utiliza metodologías de desarrollo con diversos procedimientos y herramientas de implementación, o quizá un enfoque en función de los diferentes tipos de software que requieren los usuarios, o presentar una visión global de la Ingeniería de Software que esté fuertemente centrada en la construcción del producto, pero con base en un enfoque de calidad. Éstas serían algunas de las posibilidades para presentar la visión.

De acuerdo con esta exposición, el lector, conocedor de estas temáticas, podrá inferir que existen varios puntos en común entre los enfoques descritos. De esta manera, se comprende que es necesario implementar un modelo de calidad adecuado para construir un producto software robusto y fiable, y que podrá presentar diferentes características en función del tipo de producto que necesita el usuario. Además, se debe reconocer que estas consideraciones poseen una estrecha relación con un proceso histórico que, desde un punto de vista formal y de acuerdo con importantes referencias bibliográficas, se originan en la conferencia organizada por la OTAN en 1968. Este grupo estaba constituido por profesores de informática, programadores y empresarios del sector, que coincidieron en reconocer pública y oficialmente la existencia de la denominada *Crisis del Software*.

De acuerdo con este enfoque, y basado en el contexto histórico y en aras de comprender las causas de la *crisis*, resulta necesario adentrarse un poco más en el tiempo para explorar cómo se desarrollaba el software y qué características tenía. En tal sentido, la historia indica que tradicionalmente la construcción de un producto software se identificaba, fundamentalmente, como la escritura de programas para computador. Ahora bien, conforme se iba incrementando la complejidad de los problemas, que debían resolver los programadores de software, se fue tomando conciencia de que el desarrollo era sólo una pequeña, pero importante, parte del proceso de la construcción de productos software.

A modo ilustrativo de lo que se desea significar, y estableciendo una analogía con actividades de la Ingeniería Civil, es necesario comprender que aunque es importante realizar adecuados cálculos estructurales, también se deben contemplar otras cuestiones que son pilares para alcanzar el éxito del proyecto, como el cómputo de materiales, la contratación de personal, la erogación presupuestaria en la línea de tiempo y la correcta ejecución de la obra, entre otras actividades. En otras palabras, no sería posible llevar a cabo un proyecto mediano de construcción, suponiendo que sólo se deben llevar a cabo las actividades de cálculo estructural, y sin tener en cuenta las otras que, probablemente, no tendrían mayor incidencia cuando se realizan construcciones menores. Lo mismo podría analizarse para proyectos de construcción de software, actividad que ha sufrido una metamorfosis similar a lo largo del tiempo. Es decir, de la misma manera que la profesión del ingeniero civil o del arquitecto dejó de estar circunscripta sólo al cálculo estructural y a la ejecución de la obra, la labor profesional del desarrollo de software ha evolucionado e incorporado actividades que van más allá de sólo escribir líneas de código.

En consecuencia, la evolución de la informática y el incremento de la complejidad de los problemas que los usuarios necesitan resolver con productos software, ocasionaron que las líneas de

código de las soluciones informáticas se incrementaran en número, a la vez que se planteó la necesidad de estructurar, validar e implementar metodologías para construir y diseñar los sistemas que respondieran a las necesidad sociales. Además, se requirió perfeccionar los métodos para comprender y resolver cuestiones de mantenimiento, portabilidad, eficiencia, calidad y fiabilidad, entre otras tantas. Esta demanda desbordada por productos software de calidad, que resolvieran adecuadamente los diferentes problemas sociales, y que empezaron a superar los tópicos específicos de las ciencias, originó la llamada *crisis*.

Aunque es cierto que los profesionales y la industria en desarrollo de software no estaban preparados para atender este tipo de demandas, también es cierto que construir software es una cuestión de alta complejidad, debido a que el producto no se puede observar físicamente y se pierde, de alguna manera, el control que el fabricante necesita tener sobre él. Desde esta perspectiva, un asunto que se debe analizar es *qué significa construir correctamente un producto software*. Para intentar responder esta cuestión, la comunidad hace referencia a algunos atributos que debe tener ese producto:

- *Ser Fiable:* que los programas que lo componen fallen lo menos posible
- *Ser Fácil de Modificar:* que el mantenimiento de esos programas sea sencillo de realizar
- *Ser entregado a tiempo al usuario:* que se respete el período de tiempo estipulado para la entrega
- *Ser entregado conforme al costo establecido al usuario:* que se respeta el monto de dinero estipulado para la entrega
- *Ser Portable:* que los programas se puedan cambiar de su entorno hardware original
- *Ser predecible en su funcionamiento:* que responda a las expectativas que tienen los usuarios

Cuando el software empezó a carecer de la mayoría de estos atributos, se percibieron los síntomas de que se estaba manifestando una crisis en el desarrollo y construcción de los productos, lo que se constituyó en el objetivo central de la conferencia de la OTAN, considerada como un hito en la historia del desarrollo de software porque permitió delinear un posible rumbo para hacer frente a esas dificultades. Este contexto sentó las bases para que Bauer, director de la conferencia, propusiera: *…establecer y usar principios de ingeniería orientados a obtener software de manera económica, que sea fiable y funcione eficientemente sobre máquinas reales*..., lo que se considera como el surgimiento de la *Ingeniería de Software*.

Desde un punto de vista práctico y sintético, la Ingeniería de Software se ocupa de abordar el desarrollo de software mediante un proceso ingenieril, que contemple cuestiones básicas como respetar los tiempos y costos estipulados, que sea eficiente y que cumpla con las prestaciones requeridas por el usuario. Asimismo, para poder producir software de calidad, es preciso que los ingenieros de software adopten un enfoque sistemático y organizado en su trabajo [SOM05]. No obstante, si bien uno de los principios ingenieriles consiste en seleccionar los métodos y herramientas que mejor se ajusten para la resolución de un determinado problema, en ciertas circunstancias también es preciso adoptar un enfoque que abarque lo creativo y artesanal, como en el caso del desarrollo de software. Es por esto que algunas comunidades consideran que esta ingeniería es tanto arte como ciencia.

En los párrafos anteriores se aprecia la aplicación de un enfoque histórico–cronológico, mediante el cual se presenta una visión global de la Ingeniería de Software, tratando de describir el origen de la crisis del software, los principales síntomas que hicieron posible su identificación, y las circunstancias que motivaron la necesidad de hacer del software un proceso ingenieril. Es decir, el surgimiento y la aceptación de que esta actividad es una cuestión de ingeniería. Aunque se han superado muchos de los problemas que aquejaban la construcción de software de calidad, actualmente, casi medio siglo después de la conferencia, todavía no se puede afirmar con total certeza que la construcción de software ha alcanzado el nivel de madurez de un desarrollo ingenieril.

2. PANORAMA DE LA INGENIERÍA DE SOFTWARE EN AMÉRICA LATINA

En esta sección se presenta un panorama general de la realidad y las necesidades que afronta esta ingeniería en el contexto Latinoamericano, teniendo en cuenta que buena parte de las consideraciones tratadas también hacen parte de los desafíos que afronta la disciplina en el contexto mundial. Si bien el contenido se puede abordar desde diferentes enfoques, se optó por hacerlo desde tres ejes, considerados centrales, para una presentación ajustada y comprensible del objetivo que se busca. El primero se refiere a los retos de la formación académica de los profesionales en desarrollo de software, el segundo se focaliza en los actuales desafíos relacionados con los atributos de los productos software, y el tercero cubre los retos vinculados con la ética y la responsabilidad profesional del área.

2.1 La formación académica

La posibilidad de obtener un producto software de alta calidad es una cuestión que depende de diversos factores, pero uno de los que mayor influencia tiene es la capacidad de los profesionales encargados de su desarrollo. En tal sentido, su formación técnico-académica debe ser lo suficientemente sólida como para poder llevar a cabo los proyectos, y responder a las exigencias, dificultades y necesidades que plantean los problemas sociales. Además, también deben ser competentes y mantener actualizado su conocimiento, fortaleciendo la formación con cursos de posgrado y de perfeccionamiento en las áreas específicas que se los exija. De acuerdo con estas consideraciones, las instituciones que ofrecen formación académica en Ingeniería de Software en la región, deberían tener en cuenta para su proyección las siguientes recomendaciones:

- Proporcionar formación académica, que les ayude a los estudiantes a mantener sus conocimientos actualizados, conforme a las necesidades actuales y futuras de la disciplina

- Crear las condiciones necesarias para formar y ofrecer un adecuado número de profesionales, para hacerle frente a la creciente demanda de los usuarios y de los de productos software

- Mantener actualizados los planes de estudio, con mallas curriculares y contenidos en los que se vinculen temas como control y gestión de proyectos, paradigmas de calidad y mejoramiento de procesos, entre otras

- Fomentar la creación y/o fortalecimiento de centros y grupos de investigación, por medio de subsidios que permitan el desarrollo de programas de transferencia y de colaboración nacional e internacional

- Orientar las especializaciones en tópicos de Ingeniería de Requisitos, debido a que es un factor importante para la construcción de productos software que satisfagan las necesidades del usuario

- Fomentar y fortalecer la formación de posgrado, mediante programas de maestría y doctorado de alto nivel e impacto

- Implementar programas de becas, para que los estudiantes estén en condiciones de afrontar estos estudios y de forma que sean atrayente para los candidatos

- Fomentar el ingreso de recurso humano con trayectoria en la industria al sistema científico–tecnológico, para fortalecer la investigación aplicada y la transferencia de tecnología.

Es posible que se puedan seleccionar otros enfoques para el desarrollo de este eje, teniendo en cuenta que deben ser adecuados en virtud de las carencias y necesidades que afronta la Ingeniería de Software en el contexto de la realidad que atraviesa Latinoamérica, y teniendo en cuenta las diferencias lógicas que se manifiestan en cada contexto nacional.

2.2 Atributos de los productos software

Actualmente, al software se le considera como parte inherente y en ciertos casos fundamental para el desarrollo de las distintas actividades cotidianas y de negocios de la sociedad globalizada. En este sentido, la necesidad de construir productos software de alta calidad es cada vez mayor, y en consecuencia las empresas y los profesionales del sector deben velar porque las soluciones se ajusten a los parámetros de calidad acordados y requeridos por el usuario. Debido a esto, sería adecuado focalizarse en atributos de calidad que no se relacionen con la *funcionalidad del negocio* en una aplicación, como:

- *Confiabilidad*. El software confiable no genera problemas económicos, físicos o de funcionamiento, cuando tenga lugar una falla del sistema. En este sentido, adquiere relevancia la necesidad de desarrollar la funcionalidad del sistema por lapsos ininterrumpidos de tiempo.

- *Seguridad*. Esta cuestión va más allá de características estrictamente técnicas, como encriptación y controles de acceso, porque actualmente también se presentan fraudes, ataques y violaciones a la información personal, como el robo de identidad, que crecen en complejidad, y en consecuencia las estrategias para enfrentarlos deben evolucionar en la misma medida para evitarlos y prevenirlos.

- *Precisión*. La sociedad se encuentra inmersa en un mundo digital que refleja de forma muy cercana al mundo real. Las soluciones informáticas deben responder con precisión a las necesidades que surjan en este sentido; por ejemplo, que un pago con tarjeta o un depósito en ATM se refleje automática e inmediatamente en el estado de cuenta.

- *Usabilidad*. Los productos software deben ser sencillos de utilizar por los usuarios, es decir, que deben poseer una interfaz adecuada y una documentación consistente.

- *Mantenimiento*. La documentación del producto debe evolucionar en la medida que se presentan los cambios y actualizaciones, por lo que este atributo se puede considerar de carácter crítico,

debido a que un cambio en el software trae consecuencias directas en el ámbito del negocio [SOM05].

Por otro lado, es importante señalar que estos atributos dependen fundamentalmente del tipo de aplicación, es decir, un sistema bancario debe ser seguro, mientras que un sistema de juego interactivo debe poseer una buena capacidad de respuesta.

2.3 Responsabilidad ética y profesional

Como en otras áreas, los procesos de la Ingeniería de Software se deben llevar a cabo dentro del marco legal y la responsabilidad profesional que regula la actividad del sector, por lo que es necesario que estos ingenieros desarrollen su trabajo de forma que va más allá de la aplicación de sus habilidades técnicas, sino que también deben abarcar aspectos de carácter ético y profesional. Si bien es cierto que organizaciones como ACM e IEEE estructuran y promulgan códigos de conducta profesional, también lo es el hecho de que cada país y región tienen características propias e individuales, como el caso Latinoamericano. Pero el extremo es que en diversos países no se tiene una regulación adecuada ni ajustada a las responsabilidades sociales y morales del ejercicio profesional de los desarrolladores de software. Los siguientes son ejemplos de algunas acciones que no se encuentran reguladas por un marco legal, sino por normas básicas de responsabilidad profesional:

- *Competencia*. El ingeniero de software no debería aceptar trabajos que se encuentran por fuera de su esfera y capacidades profesionales

- *Confidencialidad*. El ingeniero de software debe respetar la confidencialidad del cliente/usuario, con independencia de que se haya o no firmado un acuerdo de confidencialidad

- *Uso de Hardware*. El ingeniero de software no debe utilizar sus habilidades técnicas para emplear inadecuadamente el hardware de otras personas u organizaciones.

Estructurar un código de conducta profesional como *marco regulatorio* de la actividad, ayudaría a alcanzar metas como la trazada por la Red Latinoamericana en Ingeniería de Software (RedLatinaIS), en el sentido de lograr la profesionalización del desarrollo del software [SER13].

3. PROSPECTIVA DE LA INGENIERÍA DE SOFTWARE

La Ingeniería de Software en Latinoamérica deberá tener una visualización del todo por encima de sus partes, a efecto de operar mejor la complejidad que se presente en su campo de acción; ayudando a detectar los indicadores más transcendentes e impactantes, y teniendo en cuenta la velocidad e intensidad de los cambios tecnológicos, económicos y sociales de la sociedad de este siglo. Por otro lado, debe generar un consenso en el sentido de aceptar que es necesario que los conocimientos generados en la investigación traspasen la conceptualización teórica y llevarlos a la práctica.

3.1 Retos y realidades de la Ingeniería de Software

Tratar de predecir el futuro de la Ingeniería de Software no es como leer una esfera de cristal, debido a que es complicado por la amplia variedad de variables involucradas. En este caso, se tratará de

verlo desde la prospectiva metodológica propuesta por Gaston Berger [BER79], quien argumenta que el futuro depende de las acciones del hombre, y luego éste se construye. La cuestión no es predecir sino reflexionar acerca de los cambios, y preparar las reacciones para enfrentar las diferentes circunstancias que puedan generar los derroteros de esta ingeniería; buscar un futuro deseable y responder a las ambiciones que las sociedades de ingenieros desean alcanzar. Luego se requiere un alto grado de creatividad, aplicando elementos de cambio y de transformación, para asumir una actitud activa en busca de lo que se desea de la Ingeniería de Software como disciplina. En los siguientes párrafos se describen algunas problemáticas y tendencias importantes para el futuro de esta ingeniería.

3.1.1 Calidad del software

La calidad de un producto software depende del cumplimiento de un amplio listado de variables y de indicadores, que se deben tener en cuenta en las fases iniciales del ciclo de vida. Para alcanzarla es necesario darle mayor importancia a cuestiones como los requisitos no-funcionales, y a características como interoperabilidad, flexibilidad, usabilidad, confiabilidad, desempeño, escalabilidad, seguridad, etc. Entre los atributos más relevantes se encuentran:

- *Seguridad*. Debido a iniciativas como cero papel, mejoramiento ambiental, gestión documental, y la tendencia actual hacia la digitalización de la información de las empresas y la sociedad en general, y debido a que prácticamente todos los dispositivos electrónicos tienen conectividad, se vuelve sensible la información confidencial y protegida por marcos legales. Es por esto que este atributo toma importancia, y se hace imperante mejorar la seguridad y sus atributos de calidad, como la integridad, la privacidad y la confidencialidad, incrementado el nivel de la complejidad y la efectividad de los mecanismos software que prevengan las diferentes modalidades de ataque.

- *Disponibilidad y Confiabilidad*. La dependencia social, organizacional y de gobierno de los sistemas de información, exige el aseguramiento de una funcionalidad ininterrumpida de todas sus funciones, y cada vez son menos tolerables las interrupciones o fallas.

- *Usabilidad*. El acceso cada vez mayor a todo tipo de usuario hace que el desarrollo de software sea más exigente en el diseño de interfaz y factores relacionados con la usabilidad, y cada vez se va ir agudizando más este problema en el análisis y diseño, lo cual incrementará los costos.

3.1.2 Servicios y modelos de negocio

En la misma medida que las necesidades de los clientes se vuelven complejas, la Ingeniería de Software debe evolucionar para poderlas solucionar. Es por esto que se necesita avanzar en propuestas como el modelo de software como servicio, la optimización de recursos y los cambios en las arquitecturas, integrando nuevas capas, como la de servicios e infraestructura, para ofrecer viabilidad a las nuevas demandas de productos y a un nuevo modelo de desarrollo de aplicaciones. Otro modelo que se debe potencializar es el de desarrollo en comunidades colaborativas, y corrientes como el software libre. Porque el uso y la demanda de servicios implicarán cambios en el concepto de licenciamiento y de concentración de mercados, lo que impactará a las pequeñas y medianas empresas que tendrán que orientarse a los componentes o los *frameworks*, y a incrementar la reutilización. Estos cambios también afectarán la cadena de valor de este sector productivo, que en Latinoamérica tiene una amplia participación en las economías de cada país.

3.1.3 Integración de sistemas y componentes

Dada la actual tendencia a la integración de componentes y sistemas, la demanda por la construcción de software por componentes se incrementará, y esto expondrá a las organizaciones a la pérdida del control sobre los mismos, lo que la expondrá posiblemente a perder la oferta de servicios y a cambiar la calidad y funcionalidad de los componentes y versiones. Esta dinámica se incrementará cada vez más al perder el control del cambio, lo que constituye un reto que la Ingeniería de Software debe enfrentar y resolver a corto plazo.

3.1.4 Dispositivos móviles y conectividad

La ampliación de los mercados y el crecimiento en oferta y demanda por los dispositivos móviles, hace que continuamente se replanteen las necesidades de los usuarios, tanto en conectividad como en el diseño de aplicativos. En este contexto sobresalen cuestiones como portabilidad, transportabilidad, interoperabilidad, usabilidad, seguridad y calidad de los desarrollos para empresas, las redes y comunicaciones, la web y el entretenimiento. Las implicaciones de estos cambios marcan el surgimiento continuo de nuevas empresas que ofrecen soluciones, a veces sin una aplicación práctica, pero que los mercados acogen porque ven oportunidades de negocio. La competencia en estos entornos genera cambios que impactan la industria del software, específicamente a los modelos de ingeniería que se deben aplicar para responder a las solicitudes de productos seguros y fiables. Varias multinacionales de esta industria ofrecen actualmente la posibilidad de que el usuario pueda *construir* y ensamblar el equipo con las características que necesita, a la vez que lo personaliza en hardware y software, como si fuera un Lego. A estos escenarios se unen cuestiones como la demanda por ancho de banda, porque cada equipo *armado,* sin estudios de soporte y requisitos adecuados, genera retos para las plataformas que están instaladas. Esto es trabajo futuro que también debe atender la Ingeniería de Software.

3.1.5 Recurso humano competente

La falta de control adecuado al ejercicio del desarrollo profesional de software, la tendencia a abrir programas que ofrecen perfiles en programación, los bajos niveles en comprensión y aplicación de las matemáticas, la falta de experiencia profesional de los profesores, la falta de comprensión en la industria del rol de los ingenieros de software, y el concepto facilista que tienen los estudiantes en formación, son algunas de las causas por las que actualmente se vive una crisis en la ingeniería. Mientras la oferta de programas es amplía, la demanda de los estudiantes disminuye, alentando el surgimiento de la informalidad, y de programadores ejerciendo funciones como desarrolladores. Uno de los retos en la formación en Ingeniería de Software será ofrecer y potencializar el desarrollo de software como profesión, porque, como se ha indicado varias veces, la complejidad de los problemas se incrementa constantemente y se necesitan profesionales capacitados para solucionarlos.

4. TENDENCIAS

4.1 Métodos ágiles e incrementales

Aunque en la actualidad los métodos ágiles todavía no escalan lo suficiente como para abordar el desarrollo de grandes sistemas, proporcionan nuevas perspectivas y técnicas, para atender otro tipo de desarrollos a los cambios rápidos y a productos de componentes comerciales y libres, lo que ayuda a mejorar la atención a las modificaciones que los usuarios solicitan. Otra tendencia es el trabajo incremental y distribuido geográficamente, en el que se integran nuevas tecnologías y grupos de

desarrollo desde diferentes países y zonas horarias. En estos procesos se requiere afinar métodos, metodologías y herramientas para lograr una adecuada interacción y una fuerte cohesión.

4.2 Dominio de los productos software

El dinamismo de los procesos en la actual Sociedad de la Información y el Conocimiento se orienta a la evaluación de procesos de desarrollo de productos, con tecnologías emergentes y orientadas por modelos, aspectos y agentes, que buscan mayor integración de componentes y de patrones de software. Para lograrlo adecuadamente se debe tener en cuenta la velocidad de los cambios y la obsolescencia temprana del producto, unas características que impactan el proceso de la Ingeniería de Software, por lo que los profesionales deberán tener en cuenta las técnicas, los procesos y las herramientas necesarios para gestionar los requisitos, los cambios, la Validación, la Verificación y el control de versiones.

4.3 Ingeniería de Requisitos en entornos globales

Esta fase del ciclo de vida de la Ingeniería de Software es uno de los temas que ha tenido mayor acogida desde el surgimiento mismo de esta área del conocimiento. Pero, aunque los progresos son evidentes, todavía no se logra estructurar una técnica o metodología lo suficientemente confiable como garantizar la calidad de la especificación. Con la aplicación de las metodologías de trabajo global y de proyectos conjuntos entre profesionales de diversas partes del mundo, la realización de esta fase se complica y en ocasiones genera brechas culturales y de interpretación.

4.4 La prueba del software

Aunque se esperaba que con el desarrollo y la masificación de nuevos principios, técnicas y métodos se incrementaría la calidad de los productos software, el objetivo no se ha logrado a plenitud. Los usuarios y organizaciones todavía padecen las fallas de los sistemas y no se vislumbra una solución radical a corto plazo. En este aspecto se espera que desde la Ingeniería de Software se planteen proyectos orientados a generar conocimiento acerca de cómo probar adecuadamente el software, antes de entregarlo al cliente. Es por esto que se tienen esperanzas en los trabajos de los Métodos Formales, la automatización de las pruebas y la comprensión universal de que la prueba es un proceso paralelo a todas las fases del ciclo de vida del producto, y no una fase de choque previa a la puesta en funcionamiento del programa.

5. PROSPECTIVA DE LAS NECESIDADES DE LA SOCIEDAD ACTUAL

La Ingeniería de Software se debe plantear retos y objetivos a corto plazo para atender las necesidades de la actual Sociedad de la Información y el Conocimiento. Desde una visión funcional, el conocimiento se considera hoy como uno de los activos más importantes de las organizaciones, y por tanto se debe poder salvaguardar, gestionar y utilizar de forma ágil y productiva. Las herramientas que permiten alcanzar este objetivo parten desde lo humano hasta los Sistemas de Información, creando una cadena de actores y ayudas tecnológicas que se ponen al servicio de la sociedad, para lograr que la información se convierta en el conocimiento necesario para el desarrollo. En todo este proceso median los productos software, y se espera que su fiabilidad se incremente en la misma medida que se viene presentando la dependencia social de los mismos.

5.1 Gestión del conocimiento

De acuerdo con Thomas Davenport [DAV08], la gestión del conocimiento es la función de planificar, coordinar y controlar los diferentes flujos del conocimiento, que produce una organización, para la realización de sus diferentes actividades y de su entorno con el fin de generar nuevas competencias. Este proceso enriquece el quehacer desde un enfoque sistémico y dinámico para asegurar, desarrollar y mantener recursos intangibles. Por su parte Chou y Lin [CHO02] identifican cuatro procesos en la gestión del conocimiento: 1) desarrollar nuevo conocimiento, 2) asegurar el conocimiento existente, 3) distribuir el conocimiento logrado y 4) combinar el conocimiento disponible. Además, para gestionar el conocimiento las organizaciones requieren cuatro pilares:

1. *Personas*: recursos humanos y cultura organizativa asociada
2. *Procesos*: estrategia de negocio, organización, metodología y rutinas, vinculando el conocimiento asociado a cada área
3. *Contenidos*: información crítica, interna y externa, para el éxito de la organización
4. *Tecnología*: software y hardware necesarios para recoger, almacenar y distribuir los datos, la información y el conocimiento explícito e implícito.

Por su parte, la Ingeniería de Software se ocupa de las teorías, métodos, metodologías, herramientas y técnicas para desarrollar y mantener productos software de calidad, por lo que la gestión de conocimiento está inmersa por la misma naturaleza de esos procesos, Nakakoji et al [NAK06] indican que el desarrollo de software es un proceso cognitivo que depende del conocimiento, y por ende se debe gestionar, y la importancia de esta gestión tiene las siguientes características [QUI07]:

1. Los procesos de desarrollo de software requieren conocimiento de múltiples dominios, entre ellos el de la computación y el de la aplicación

2. El conocimiento necesario para construir una aplicación se distribuye entre el grupo gestor del proyecto (analistas, diseñadores, programadores,…) y el mundo exterior, por lo que es necesario integrar las fuentes de conocimiento para desarrollar una solución de calidad

3. El objetivo de un proyecto software es dinámico y no se define totalmente al inicio del mismo

4. Existe un proceso continuo y dinámico de conocimiento, y de transformación del mismo, hasta lograr la construcción del producto

5. El ciclo de vida del desarrollo es una evolución continua de los conocimientos y los procesos de construcción

6. La construcción de software impacta diferentes grupos sociales y áreas geográficas, y el conocimiento que cada uno aporta colabora para el desarrollo del proyecto

Debido a esta estrecha relación entre el conocimiento y los procesos de la Ingeniería de Software la gestión de éste se orienta a:

1. Comprender el conocimiento que existe en el dominio de la aplicación

2. Brindarles a los analistas, arquitectos y desarrolladores una manera fácil y eficiente de encontrar la información que necesitan para el desarrollo

3. Proporcionarle al equipo de desarrollo una forma de hacer seguimiento a los objetivos planteados al inicio y a los cambios que surjan y sus implicaciones

4. Permitir la construcción y retroalimentarlo a cada uno de los integrantes para complementar sus conocimientos y lograr los resultados esperados.

Esta gestión del conocimiento, desde la Ingeniería de Software, es un área de investigación que se orienta a identificar, capturar y preservar el conocimiento y la experiencia adquirida durante el desarrollo de productos software, con el propósito de estimular y facilitar su reutilización en nuevos proyectos. Si las empresas desarrolladoras logran gestionar las experiencias de los participantes en los equipos de trabajo, este aprendizaje podría ser continuo y con el tiempo generaría buenas prácticas para aplicar en el futuro. En el desarrollo de esta gestión los equipos se encontrarán con procesos como:

- *La gestión de datos*, que consiste del desarrollo y la ejecución de arquitecturas, políticas, prácticas y procedimientos para direccionar apropiadamente las necesidades del ciclo de vida de los datos de una empresa. Para lograrlo las personas necesitan de principios y conceptos de la Ingeniería de Software como:

  1. Modelado de datos
  2. Administración de base de datos
  3. Data *warehousing*
  4. Migración de datos
  5. Minería de datos
  6. Calidad de datos
  7. Seguridad de datos
  8. Gestión de meta-datos
  9. Arquitectura de datos

  Además de un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia, al momento de analizar los aspectos de interés para una organización y sus relaciones. Este proceso es una actividad necesaria y la habilidad para lograrlo sólo se adquiere con la experiencia.

- *La gestión y vigilancia de la información*. El desarrollo de las TIC y el incremento en el volumen de información que generan y procesan las organizaciones ha generado el problema de la mezcla de datos. Es decir, no es fácil separar los datos que son fiables y tienen valor de aquellos que no tienen mucha importancia y que pueden no tener valor para la organización, lo que genera pérdida de datos a través de patrones tecnológicos y sociales que los exportan e importan sin una adecuada validación y evaluación, con inconvenientes asociados como:

  1. Búsqueda de ventajas competitivas sobre la base de procesos generativos de conocimiento aplicados a los productos y servicios, lo que depende de las necesidades y expectativas cambiantes de los clientes, que no son fáciles de determinar, que son costosas y que requieren vigilancia permanente.

2. Altos niveles de saturación, porque se presenta sobreproducción de información de calidad variable y difusión abierta.

3. Las fuentes de la información que circulan y procesan las empresas son diversas, lo que dificulta la implementación de procesos de seguimiento y vigilancia para auditarlas antes que impregnen la toma de decisiones.

Estos inconvenientes plantean la necesidad de diseñar e implementar, desde la Ingeniería de Software, sistemas de vigilancia para monitorear, analizar y contextualizar la información que sea de utilidad para las organizaciones, dependiendo de su aplicación y utilidad y de los diferentes grupos de trabajo. Estas iniciativas deben comprender acciones de búsqueda, localización, focalización, filtrado, almacenamiento y análisis de diferentes tipos, con el objetivo de generar valor agregado desde los datos identificados como potencialmente útiles.

Atender esta cuestión requiere una infraestructura tecnológica adecuada, que facilite los flujos de información y una comunicación interactiva entre las personas y los grupos que conforman la organización. En este sentido, Israel et al. [ISR06] citan la concepción de Gates [GAT99] sobre el Sistema Nervioso Digital (SND), acerca de cómo debería ser la infraestructura tecnológica al interior de las empresas: *El sistema nervioso digital… se distingue de una simple red de computadores por la precisión, inmediatez y riqueza de la información que lleva a los trabajadores del conocimiento y a la visión y colaboración que hace posible con la información. Los ganadores serán aquellos que desarrollen un sistema nervioso digital... tal que la información pueda fluir fácilmente a través de sus compañías para un aprendizaje máximo y constante*.

Al investigar con respecto a las alternativas tecnológicas de apoyo a la gestión de conocimiento en las organizaciones, se encuentra una diversidad de soluciones para el desarrollo y el control de los ambientes empresariales, que tienen relación con los diferentes dilemas asociados con la gestión del conocimiento. Entre ellas se encuentran soluciones como Content Management, Document Management y Knowledge Management, que se diferencian como áreas de interés específico, y que se refieren a soluciones particulares que afectan y son afectadas por otras soluciones de Gestión Empresarial, como Business Intelligence, Business Process, Management, Collaboration, Competitive Intelligence, Customer Relationship, Management, E-mail Management, Enterprise Application Integration, Enterprise Search, Image, Forms, Document Capture , Intellectual Property Management, Workflow, entre otras. Aunque se pueden considerar eficientes, estas iniciativas necesitan soporte paralelo como:

- *Manejo de Clúster*. Para fomentar el desarrollo de la Ingeniería de Software y la calidad y actualización de los métodos, metodologías y herramientas que igualmente eleven la calidad de los productos desarrollados y su competitividad en los mercados globales

- *Minería de Datos en la Web*. A través de web semánticas para sistemas distribuidos masivamente, software en entornos GRID y modelado y gestión de recursos

- *Innovación en los procesos formativos*. Las universidades deber tener en cuenta el impacto de los cambios sociales, demográficos y tecnológicos sobre los procesos formativos, y mantener actualizados sus planes y programas de estudio en áreas como la Ingeniería de Software, una de las de mayor impacto en este siglo. Esto se debe a que las transformaciones productivas, la

globalización, la competitividad, las innovaciones tecnológicas, la integración de tecnologías y la generación y gestión de conocimiento, impactan continuamente los ambientes en los que los profesionales ejercen sus funciones.

REFERENCIAS

[BER79] Berger, G. (1979). Recherches sur les conditions de la connaissance: essai d'une théorétique pure (Phenomenology: background, foreground & influences, 1). Garland Publishing.

[CHO02] Chou, D. & Lin, B. (2002). Development of Web-based knowledge management systems. Human Systems Management, Vol. 21(3), pp. 153-158.

[DAV08] Davenport, T. (2008). Enterprise 2.0: The New, New Knowledge Management? Harvard Business.

[GAT99] Gates, B. (1999). Business @ The Speed of Thought. Warner Books.

[ISR06] Israel, A.; Núñez, P. & Núñez, Y.G. (2006). Bases conceptuales del software para la Gestión del Conocimiento. Revista Venezolana de Información, tecnología y conocimiento, Vol. 3(2), pp. 63-96.

[NAK06] Nakakoji, K.; Yamamoto, Y. & Ye, Y. (2006). Supporting Software Development as Knowledge Community Evolution. In ACM CSCW Workshop on Supporting the Social Side of Large-Scale Software Development Banff, Canada, pp. 15-34.

[QUI07] Quiroga, J.A. (2007). Gestión de conocimiento en grupos de desarrollo de software. Revista Electrónica Paradigma en Construcción de Software, Vol. 1, pp. 1-15.

[SER13] Serna, M.E. (Ed). (2013). Manifiesto por la Profesionalización del Desarrollo de Software. Instituto Antioqueño de Investigación (IAI). Red Latinoamericana en Ingeniería de Software (RedLatinaIS).

[SOM05] Sommerville, I. (2005). Ingeniería de Software. Addison-Wesley.

CAPÍTULO V

LA FORMACIÓN EN INGENIERÍA DE SOFTWARE

Lucia CAMILLONI & Diego VALLESPIR
Universidad de la República
Uruguay

INTRODUCCIÓN

La creciente importancia del software en la vida diaria de las personas ha generado en los últimos años una fuerte y creciente demanda mundial de ingenieros de software cualificados, que ayuden a producir software de calidad, en el plazo y dentro del presupuesto especificado [HIS09]. La Ingeniería de Software es una disciplina relativamente nueva, pero ha madurado rápidamente debido a estas mismas exigencias. Este capítulo presenta los distintos esfuerzos que se han realizado para desarrollar y mejorar la formación en esta área del conocimiento.

En el siglo XXI, el software se ha convertido en un componente central que se relaciona con casi todos los aspectos de la vida diaria, como el gobierno, la banca, la educación, el transporte, la salud… Debido a esto, el número, el tamaño y los dominios de aplicación de los sistemas software ha crecido dramáticamente, y tienen una amplia aplicación para responder con eficiencia a las exigencias de la productividad [IEE04]. Pero, de la misma forma que la funcionalidad de los sistemas se diversifica, también se necesita desarrollar software de forma eficiente y correcta, de tal manera que no se frene el crecimiento y el desarrollo de las naciones. Los productos software actuales presentan problemas relacionados con el costo, los tiempos de entrega y la calidad [IEE04], lo que ha generado una creciente demanda por ingenieros de software calificados, que apliquen prácticas profesionales para desarrollar software de calidad, que satisfaga adecuadamente las necesidades de los clientes y usuarios, y que se entreguen a tiempo y dentro del presupuesto establecido [ACM05].

La Ingeniería de Software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, mantenimiento y operación del software [IEE90], y los principios y prácticas de esta disciplina son esenciales para el desarrollo de sistemas grandes, complejos y confiables [ACM05]. El estudio y la práctica de esta ingeniería se han visto influenciados por sus raíces en las Ciencias Computacionales, y por su surgimiento como disciplina ingenieril, pero la naturaleza intangible de su producto, el software, la hace diferente a las otras disciplinas ingenieriles. Entre sus elementos de aplicación busca integrar los principios matemáticos y computacionales con las prácticas desarrolladas por otras ingenierías, cuyos productos son artefactos tangibles [IEE04]. Por este origen combinado, actualmente persisten considerables tergiversaciones y confusiones acerca de las diferencias y similitudes que tiene con las Ciencias Computacionales. Parnas [PAR11] intenta clarificarlas, subrayando que los estudiantes de ingeniería se deben formar en cómo aplicar estas ciencias y las matemáticas para construir sistemas con masivos componentes software, mientras que los estudiantes de Ciencias Computacionales lo hacen en cómo crear nuevo conocimiento a partir de la información conocida. Generalmente, las universidades ofrecen cursos comunes para ambas áreas, pero con enfoques diferentes. Por ejemplo, los de ingeniería se orientan a aspectos relacionados con la fiabilidad, la calidad, los costos y el mantenimiento del software [ACM05], y si bien esto es discutible, algunos de estos aspectos pueden no ser considerados en los de las Ciencias computacionales.

Por otro lado, tradicionalmente el desarrollo de software lo llevan a cabo equipos profesionales multidisciplinares, que requieren una comunicación adecuada con los distintos *stakeholders* involucrados, por lo que las actividades de carácter social representan una parte importante del trabajo diario de los ingenieros de software, y les brindan una fuerte relación con otras disciplinas, como la sociología y la psicología; porque en su ejercicio requieren técnicas efectivas para compartir ideas y lograr la comprensión de los requisitos del producto [AHM08]. También es importante la comprensión del aspecto cooperativo y humano en el desarrollo y el mantenimiento de software [SOU09].

Entonces, los procesos formativos constituyen un pilar básico para ésta y cualquier profesión, y actualmente es difícil concebir que, si se considera madura, no cuente con formación igualmente madura, porque es el camino a través de la cual se forman profesionales altamente capacitados. Aunque se puede considerar que la Ingeniería de Software es un área del conocimiento relativamente nueva (el término se acuña en 1968, la primera maestría es de 1979 y el primer pregrado es de 1987), en la que, como se describe en este capítulo, en las últimas décadas se han generado modificaciones y actualizaciones importantes en lo que tiene que ver con sus procesos formativos. Así mismo, en este contenido también se presentan diferentes aspectos de la formación en esta ingeniería, y el estado actual de la misma, y se hace un recorrido a través de la retroalimentación entre las necesidades de la sociedad, la industria y las organizaciones, las propuestas formativas de la academia y las propuestas y necesidades de las sociedades profesionales, que han servido para construir y avanzar en la formación disciplinar. Conocer el estado actual de la formación en Ingeniería de Software es necesario para pensar y describir la actualidad y el futuro de estos ingenieros.

Se parte del concepto de que una disciplina madura debe contar con un cuerpo de conocimiento, porque sin él es difícil consensuar la formación, otorgar licencias, crear certificaciones y acreditar planes de estudio, que garanticen la adecuada formación de profesionales competentes. En la sección 1 se describe el proyecto SWEBOK, que se ha convertido en la guía para el cuerpo de conocimiento de la Ingeniería de Software. En las secciones 2.1 y 2.2 se presentan las guías curriculares propuestas por IEEE-CS y ACM para la formación a nivel de pregrado y posgrado. Estas guías, que se basan en el SWEBOK, presentan los conocimientos y habilidades consensuados, que debe tener un profesional en el área, y se utilizan como guía para elaborar los programas. En la sección 3 se describe la visión de la formación en Ingeniería de Software en contextos como la secundaria, y en pregrados diferentes de la misma área. En la sección 4 se presentan los elementos que conforman una profesión y su relación con el estado actual de la Ingeniería de Software. En particular, se discuten aspectos relacionados con los procesos formativos, como la acreditación de programas, las certificaciones y el licenciamiento, entre otros; pero también se describen aspectos necesarios para una profesión, como el código de ética y las sociedades profesionales. Por último, en la sección 5 se presentan las conclusiones.

1. GUÍA PARA EL CUERPO DE CONOCIMIENTO SWEBOK

The Guide to the Software Engineering Body of Knowledge (SWEBOK) es un proyecto de IEEE-CS [BOU14], en el que se describe el cuerpo de conocimiento, generalmente aceptado, para la Ingeniería de Software. El propósito del libro es brindar una caracterización consensuada de los límites de esta ingeniería, y proporcionar acceso por tópicos al cuerpo de conocimiento que la soportan [IEE04]. SWEBOK no se debe confundir con un cuerpo de conocimiento en sí, porque no es posible poner el cuerpo de conocimiento completo en un único documento, pero surge de la necesidad de contar con una guía para esta disciplina.

SWEBOK describe y organiza los componentes del cuerpo de conocimiento y proporciona acceso por tópicos y referencias bibliográficas. El concepto de *conocimiento generalmente aceptado* proviene del Project Management Institute (PMI), que lo define como aquel que se aplica a la mayoría de proyectos, la mayor parte del tiempo, y que tiene un amplio consenso al validar su valor y eficacia. En particular, el término se utiliza para distinguirlo del conocimiento avanzado y del que surge de la investigación y del especializado, como las prácticas utilizadas sólo para el desarrollo de determinado tipo de software. Lograr el consenso de todos los sectores importantes de la comunidad pertinente en un cuerpo de conocimiento representa un hito clave en cualquier disciplina. IEEE considera a SWEBOK como crucial para el desarrollo de la Ingeniería de Software como profesión, porque permite establecer lo que debe saber un profesional de esta disciplina. Sin un consenso, por ejemplo, no se podrían validar los exámenes para licenciamiento, ni establecer criterios para acreditar planes de estudio [IEE04]. La propuesta tiene los siguientes objetivos:

1. Promover una visión consistente de la Ingeniería de Software (IS) en todo el mundo

2. Aclarar el lugar y establecer los límites de la IS con respecto a otras disciplinas, como las Ciencias Computacionales, la gestión de proyectos, la ingeniería en computación y las matemáticas

3. Caracterizar los contenidos de la disciplina de IS

4. Proporcionar acceso por tópicos al cuerpo de conocimiento de IS

5. Sentar las bases para el desarrollo de currículos, certificaciones individuales y material para el licenciamiento

La guía caracteriza los contenidos de la disciplina de esta ingeniería, es decir, los conocimientos necesarios para su práctica [IEE04] que se prevé deben ser alcanzados, por ejemplo, a través de la formación de pregrado, y cuatro años más de experiencia. El objetivo del proyecto es aclarar el lugar y establecer los límites de la Ingeniería de Software, para lo cual organiza el área en 15 Áreas de Conocimiento (KA, por sus siglas en inglés):

1. Software Requirements
2. Software Design
3. Software Construction
4. Software Testing
5. Software Maintenance
6. Software Configuration Management
7. Software Engineering Management

8. Software Engineering Process
9. Software Engineering Models and Methods
10. Software Quality
11. Software Engineering Professional Practice
12. Software Engineering Economics
13. Computing Foundations
14. Mathematical Foundations
15. Engineering Foundations

Derivadas de esta propuesta, en lo que refiere a formación en IS, IEEE-CS y ACM propusieron guías curriculares para programas a nivel de pregrado (Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering SE2004) [IEE04a] y de posgrado (Curriculum Guidelines for Graduate Degree Programs in Software Engineering GSWE2009) [BAL09], que utilizan como fuente para la definición del conocimiento central. Otro ejemplo son Certified Software Development Professional (CSDA) y Certified Software Development Associate (CSDP), desarrolladas por IEEE-CS, y que lo utilizan como base para estructurar los exámenes de certificación.

2. LA FORMACIÓN EN INGENIERÍA DE SOFTWARE

2.1 Pregrado

En 1998, IEEE-CS y ACM establecieron un grupo de trabajo conjunto para crear una nueva versión de sus guías curriculares para programas de pregrado en computación, debido a que desde comienzos de 1990 esta área presentaba un desarrollo continuo y a se estaban conformando disciplinas dependientes e independientes de la misma [INV05]. En 2001, el proyecto Computing Curricula 2001 (CC2001) elaboró cuatro volúmenes de guías curriculares para disciplinas relacionadas con la computación: 1) Ciencias Computacionales, 2) Sistemas de Información, 3) Ingeniería en Computación y 4) Ingeniería de Software, a los que posteriormente se agregó otro relacionando la guía curricular para la disciplina de Tecnologías de la Información [ACM05]. El volumen Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering 2004 (SE2004), se convirtió rápidamente en la guía curricular para los programas desarrollados por IEEE-CS y ACM, a la vez que sirvió de base para que las instituciones académicas y organismos de acreditación estructuran la formación de pregrado [IEE04a].

El proyecto SE2004 se estructuró en tres etapas: la primera se orientó a determinar el conjunto de resultados esperados del currículo y a describir lo que debería saber un egresado de esta disciplina, la segunda implicó determinar y especificar el Software Engineering Education Knowledge (SEEK), que mínimamente deberían contener los programas de pregrado en Ingeniería de Software, y la tercera generó un conjunto de recomendaciones curriculares en las que se describe cómo estructurar, en diversos contextos, un currículo que incorpore el SEEK en esta ingeniería. Además, SE2004 establece un conjunto de siete resultados que se espera que logren los estudiantes al terminar el pregrado, como se aprecia en la Tabla 1, y que son lo suficientemente genéricos como para que se puedan adaptar a una variedad de programas relacionados [IEE04a].

Tabla 1. Resultados para pregrado en IS propuestos por SEEK

1	Mostrar dominio de los conocimientos y habilidades en IS, y de las cuestiones profesionales necesarias para comenzar la práctica como ingeniero de software
2	Capacidad para trabajar de forma individual y en equipo con el objetivo de desarrollar y entregar productos software de calidad
3	Ser capaz de conciliar objetivos conflictivos de un proyecto, encontrando compromisos aceptables dentro de las limitaciones de tiempo, costo, conocimiento y sistemas existentes
4	Diseñar soluciones apropiadas en uno o más dominios de aplicación usando enfoques de IS, que integren cuestiones éticas, sociales, legales y económicas
5	Demostrar que comprende y puede aplicar teorías actuales, modelos y técnicas que proveen una base para la identificación y el análisis de problemas, el diseño de software, el desarrollo, la implementación, la Verificación y la documentación
6	Comprender y apreciar la importancia de la negociación, los hábitos efectivos para el trabajo, el liderazgo y la buena comunicación con los *stakeholders* en un entorno típico de desarrollo
7	Ser capaz de aprender nuevos modelos, técnicas y tecnologías, cuando emergen, y apreciar la necesidad del desarrollo profesional continuo

SEEK es el cuerpo de conocimiento mínimo que se debería incluir en los programas de pregrado en Ingeniería de Software, además de disciplinas relacionadas como matemáticas, Ciencias Computacionales, ingeniería y economía [INV05]. Para la definición de este cuerpo se tomó como base SWEBOK 2004, al que se realizaron varias modificaciones; es decir, SEEK define el conocimiento a incluir en los planes de estudios de programas de pregrado, mientras que SWEBOK caracteriza el contenido de la disciplina de Ingeniería de Software utilizado en la práctica por los ingenieros de software, e incluye únicamente lo que está dentro de los límites de esta ingeniería, y omite intencionalmente el conocimiento de dominios y disciplinas relacionadas [IEE04a].

En SEEK también se definió el núcleo, una componente central que contiene el material básico que se debe obtener en una carrera de pregrado en esta área, y que es lo mínimo que se debe impartir en un pregrado que sigue la guía SE2004. En la construcción de este cuerpo de conocimiento se buscó que fuera lo más ajustado posible, para darle a las instituciones mayor flexibilidad para adaptarlo a los diferentes contextos. De esta forma, cada programa podrá agregar unidades adicionales que pertenezcan o no al núcleo. En la Tabla 2 se presentan las diez Áreas de Conocimiento que conforman el SEEK. Estas áreas se descomponen en módulos, llamados unidades, y cada unidad se subdivide en un conjunto de tópicos, que representan el menor nivel jerárquico, como se observa en la Figura 1.

Tabla 2. Áreas de Conocimiento SEEK

Sigla	**Área**
CMP	Fundamentos en Computación
FND	Fundamentos en Matemática e Ingeniería
PRF	Prácticas profesionales
MAA	Análisis y Modelado de Software
DES	Diseño de Software
VAV	Verificación y Validación de Software
EVL	Evolución del Software
PRO	Procesos de Software
QUA	Calidad del Software
MGT	Gestión del Software

Figura 1. Estructura jerárquica de SEEK

Para cada unidad del SEEK se indica el tiempo mínimo de dedicación, expresado en horas, que corresponde al dedicado en el aula para presentar un material en el formato de lectura tradicional, pero no incluye el tiempo dedicado por fuera de la misma, aunque se sugiere estimar entre dos y tres veces las horas presenciales como promedio de tiempo dedicado. Por otro lado, para cada tópico se indica la capacidad de dominio esperada, expresada en niveles de Bloom. En este caso se utilizan los primeros tres niveles de la taxonomía: 1) conocimiento: ser capaz de recordar lo aprendido, 2) comprensión: entender la información y el significado del material presentado y 3) aplicación: aplicar el conocimiento para resolver problemas. Sólo se utilizan estos niveles porque representan capacidades que pueden ser aprendidas a nivel de pregrado. Adicionalmente, para cada tópico se indica su relevancia en el cuerpo de conocimiento central: 1) esencial: cuando es parte del núcleo, 2) deseable: cuando no es parte del núcleo pero que debería ser incluido, de ser posible, en el núcleo del programa particular, o en caso contrario formar parte de los materiales electivos, o 3) opcional: cuando sólo se debe considerar como electivo. A partir de la concepción de SE2004, diferentes universidades del mundo han utilizado la guía para crear nuevas carreras de pregrado en Ingeniería de Software, y para adaptar y comparar los planes de estudio existentes [MIS11; FRE06; RAM07; DIN11]. Latinoamérica se ha movido a un ritmo lento en este sentido, y no es posible conseguir publicaciones que presenten adopciones del SE2004; pero, en 2011, en la Conferencia Latinoamericana de Informática (CLEI) se creó el Workshop en Nomenclatura y Acreditación en Programas de Computación. En este evento los académicos y profesionales conversan acerca de las guías curriculares internacionales y su acreditación en Latinoamérica. En la tercera edición se presenta un trabajo que estudia la evolución de los currículos de Latinoamérica [CUA13]. Si bien su foco son los currículos en Ciencias Computacionales, no los de la Ingeniería de Software, sí brinda un panorama interesante de la región.

Con el objetivo es mantener estas guías actualizadas, IEEE-CS y ACM iniciaron en 2011 un proyecto para revisar SE2004 [ARD13], para lo que definieron tres etapas:

1. *Proceso de consulta*. Para recolectar información y opiniones sobre la necesidad de modificar el modelo curricular, propuestas por los principales *stakeholders* pertenecientes a la academia, la industria y el Estado.

2. *Análisis y evaluación de los resultados*. Para determinar el tipo y la magnitud de los cambios necesarios.

3. *Preparación de un reporte*. Para describir el proceso de consulta realizado, el análisis y la evaluación, que se presentará a las juntas del área educativa de IEEE-CS y ACM. Además, se deben realizar recomendaciones acerca del tipo y magnitud de las revisiones requeridas, y una estimación del esfuerzo necesario.

Para la primera etapa se definió una lista de *stakeholders* y se estructuró una encuesta con el objetivo de recoger información acerca del estado actual y de uso de SE2004. Se recibieron respuestas de 42 países diferentes, pero la mayoría provino de los Estados Unidos. Los participantes estuvieron de acuerdo en que las Áreas de Conocimiento definidas aún son relevantes, y con base en estos resultados se definió que la estructura general de SE2004 es sólida y no se debería cambiar. Con respecto al SEEK, se determinó la necesidad de realizar revisiones menores, principalmente en la incorporación de métodos de desarrollo recientes, que han demostrado ser eficientes y eficaces, como las metodologías ágiles, y la necesidad de enfatizar en cuestiones como la seguridad y la computación orientada a servicios.

En 2012 se inició el proyecto SE2013, con el objetivo de realizar la revisión de las guías. Además, actualmente se están realizando otros esfuerzos relacionados con la elaboración de guías curriculares, como CS2013, una revisión a las guías para programas de pregrado en Ciencias Computacionales [ACM13], y SWEBOK V3. Se espera que los integrantes del proyecto SE2013 tengan una estrecha colaboración con los miembros de estos proyectos [ARD13].

2.2 Posgrado

Graduate Software Engineering 2009 (GSwE2009), es un plan de estudios de referencia para programas de Maestría Profesional en esta ingeniería [FRE06], que se puede utilizar como guía para que las instituciones diseñen o mejoren sus programas [STE09]. Se entiende por Maestría Profesional a los programas que se ofrecen a nivel de posgrado, para personas que buscan ejercer una carrera en la práctica, en contraposición a una Maestría Académica.

GSwE2009 se comenzó a desarrollar en 2007, como parte del proyecto Integrated Software and Systems Engineering Curriculum (iSEEc), en el Stevens Institute of Technology. Fue financiado por la oficina del Departamento de Defensa de Estados Unidos, y por más de dos años participaron cerca de 40 autores pertenecientes a la academia y a la industria. En 2009, IEEE-CS y ACM adoptaron GSwE2009 como parte de sus recomendaciones curriculares conjuntas en computación [ARD11], lo que significa que cumple con las expectativas en cuanto a la calidad del proceso de desarrollo de las guías y del producto como tal. Las futuras versiones de GSwE2009 serán mantenidas por estas dos sociedades profesionales. La base de la propuesta es un conjunto de recomendaciones para la creación de planes de estudio para programas de maestría, provenientes del Software Engineering Institute (SEI) [ARD13a], SWEBOK 2004 y la propuesta SE2004 [IEE04a]. Además, incluye una descripción de la arquitectura curricular, los conocimientos esperados al ingreso, el Core Body of Knowledge (CBOK) y los resultados esperados al egreso. También proporciona orientación para la construcción, el mantenimiento y la evolución de las recomendaciones. A su vez, cuenta con un anexo que describe las áreas de conocimiento definidas en el CBOK, no contenidas en SWEBOK 2004. La arquitectura de la guía curricular está compuesta por un contenido preparatorio, un contenido central, un contenido específico de la institución, un contenido electivo y una experiencia final, como se observa en la Figura 2.

- *Contenido preparatorio*. Es aquel que debe dominar el estudiante antes de entrar al programa de maestría

- *Contenido central*. Son las habilidades y conocimientos básicos que todo graduado de maestría en Ingeniería de Software debería tener; definen el cuerpo de conocimiento central (CBOK)

- *Contenido específico de la institución*. Corresponde a los contenidos que la institución podría incluir para adaptar su programa y cumplir con sus objetivos específicos

- *Contenido electivo*. Permite que los estudiantes se enfoquen en sus intereses particulares dentro del enfoque establecido por el programa

- *Experiencia final*. GSwE2009 recomienda que las maestrías cuenten con una experiencia final, que puede ser un proyecto, una práctica o una tesis

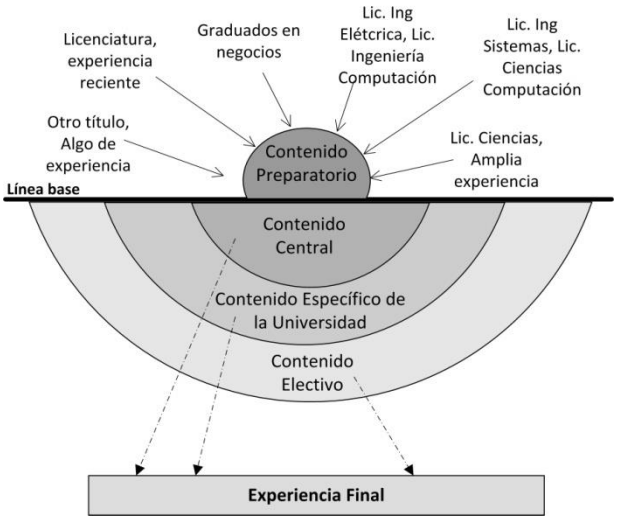


Figura 2. Arquitectura del GSwE2009

GSwE2009 asume que los estudiantes que ingresan a la maestría cumplen con las siguientes condiciones: 1) ser egresados de una carrera de pregrado en informática, ingeniería o científica con algún estudio en computación, 2) haber realizado algún curso introductorio de Ingeniería de Software, y 3) tener por lo menos dos años de experiencia práctica en algún aspecto de esta ingeniería. Los conocimientos esperados para el ingreso se definen de tal forma que se puedan lograr los resultados definidos para el egreso. Al igual que en el CBOK, el conocimiento previo se presenta dividido en KA, y para cada una se establece el nivel de Bloom que el estudiante debería tener al ingreso.

El CBOK es una descripción de las principales habilidades, conocimientos y experiencias que se espera que el estudiante adquiera para lograr cumplir los resultados de egreso. Está organizado de forma jerárquica en tres niveles: KA, unidades y tópicos. Para cada KA se indica el nivel de conocimiento que se espera logre el estudiante en aproximadamente 200 horas de contacto, expresada en la Taxonomía Bloom. El CBOK fue desarrollado tomando como base principal el SWEBOK 2004, al que se realizaron algunas modificaciones, como la adicción de nuevas KA y la modificación de algunas unidades y tópicos. Estos cambios se realizaron con el objetivo de dar soporte al logro de los resultados esperados, establecidos en GSWE2009, y para adaptarse a las necesidades y opiniones de la academia, la industria y las sociedades profesionales. En la Tabla 3 se describen las KA del CBOK, comparadas con las de SWEBOK 2004. Se añadieron las KA de Ética y Conducta Profesional e Ingeniería de Sistemas, y se quitó la de Herramientas y Métodos de la Ingeniería de Software.

Tabla 3. Comparación entre CBOK y SWEBOK 2004

KA CBOK	KA SWEBOK 2004
Ética y Conducta Profesional	
Ingeniería de Sistemas	
Ingeniería de Requisitos	Requisitos del Software
Diseño de Software	Diseño de Software
Construcción de Software	Construcción de Software
Testing	*Testing* de Software
Mantenimiento de Software	Mantenimiento de Software
Gestión de la Configuración	Gestión de la Configuración del Software
Gestión de la Ingeniería de Software	Gestión de la Ingeniería de Software
Procesos de la Ingeniería de Software	Procesos de la Ingeniería de Software
Calidad del Software	Calidad de Software
	Herramientas y métodos de la Ingeniería de Software

En el currículo se establecen diez resultados esperados para el egreso, es decir, lo que se espera que asimile el estudiante al culminar la maestría. Estos resultados cubren diversos aspectos técnicos, éticos y de aprendizaje. En la Tabla 4 se describen los resultados esperados.

Tabla 4. Resultados esperados de egreso

Resultado	Descripción
CBOK	Dominar el CBOK. El CBOK especifica niveles de Bloom que deberían ser cumplidos para cada KA
Dominio	Dominar la Ingeniería de Software en un dominio y tipo de aplicación particular
Profundidad	Dominar al menos una KA o sub-área del CBOK en el nivel de Bloom de Síntesis
Ética	Ser capaz de tomar decisiones éticas y practicar con comportamiento ético profesional
Ingeniería de Sistemas	Entender la relación entre la IS y la ingeniería en sistemas. Ser capaz de aplicar principios y prácticas de la ingeniería de sistemas en la IS
Equipo	Ser un integrante efectivo de un equipo, pudiendo liderar un área del desarrollo o mantenimiento de software
Conciliar	Ser capaz de conciliar objetivos conflictivos de un proyecto, encontrando compromisos aceptables dentro de las limitaciones de tiempo y costo
Perspectiva	Entender y valorar el análisis de factibilidad, la negociación y las buenas comunicaciones con los *stakeholders*
Aprender	Ser capaz de aprender nuevos modelos, técnicas y tecnologías cuando estas emergen. Apreciar la necesidad del desarrollo profesional continuo
Tecnología	Ser capaz de analizar tecnologías de software actuales, compararlas con tecnologías alternativas, y especificar y promover mejoras o extensiones a esas tecnologías

Al igual que sucede con la guía SE2004, la adopción de la guía curricular GSwE2009 es lenta en esta región, y la revisión bibliográfica acerca de su uso arroja pocos resultados. En artículo se presenta cómo cuatro Universidades de tres países distintos han usado esta guía para la construcción y adaptación de programas de Maestrías [ARD13a], y otro describe una adaptación realizada en Uruguay para la creación de un programa de Maestría en Ingeniería de Software [VAL12].

3. OTROS NIVELES DE FORMACIÓN EN INGENIERÍA DE SOFTWARE

El futuro de esta profesión depende de los diversos niveles de formación, formales e informales, que se ofrecen en todo el mundo para la próxima generación, porque los jóvenes no deberían tener que esperar hasta ingresar a la universidad para tener sus primeros acercamientos a las Ciencias Computacionales [THI13]. Diversos estudios ratifican lo que la lógica difunde, en el sentido de que los seres humanos asimilan mejor cuando tienen acceso al conocimiento a edades tempranas. Es por

esto que las Ciencias Computacionales se deben apreciar como una forma de alfabetización, y deben hacer parte de los conocimientos básicos de la formación en este siglo [SOB12]. En algunos países se están realizando esfuerzos para incluir esta área en los ciclos tempranos de formación de los niños, o generación Net [KEN09]. Por ejemplo, en Estonia se imparte programación a los adolescentes de primer grado [THI13]; en Estados Unidos se crearon escuelas secundarias, como *Academy for Software Engineering* (AFSE), para orientar a los estudiantes en carreras relacionadas con las Ciencias Computacionales o la Ingeniería de Software [ARD12], y en Uruguay se creó la opción de Bachillerato Tecnológico en Informática, para introducir a los estudiantes en temas relacionados con la computación, como la programación, los sistemas operativos y las bases de datos.

Para los estudiantes no siempre es sencillo tomar la decisión de comenzar una carrera universitaria, por lo que deben ser alentados para seguir sus intereses, pero teniendo en cuenta sus particularidades y factores como la capacidad, el potencial para el empleo y el costo al tomar esa decisión [SOB12]. Las habilidades e intereses de las personas son diversos, y para muchas de ellas no es prioritario ni necesario obtener un título universitario. Es común encontrar instituciones que ofrecen programas de capacitación informal, pero sin ningún respeto por la profesión misma, y sin tener en cuenta las iniciativas y modelos que se describen en este capítulo, por lo que se debe regular y controlar la oferta informal de formación, especialmente en las áreas que aquí nos interesan.

4. NIVEL DE MADUREZ DE LA INGENIERÍA DE SOFTWARE COMO PROFESIÓN

4.1 Elementos que conforman una profesión

De acuerdo con el Manifiesto por la Profesionalización del Desarrollo de Software [SER13], en la industria y en la academia se visualiza a la Ingeniería de Software como es una profesión todavía inmadura, y desde hace algún tiempo se ha buscado entender y estudiar esta afirmación. El su reporte, Gary Ford y Norman Gibbs [FOR96] estudiaron diversas profesiones establecidas, como la medicina, la arquitectura, la ingeniería civil y la contabilidad. A partir de este estudio observaron que, a pesar de ser disciplinas diferentes, existen componentes comunes a todas. El reporte presenta un modelo que permite caracterizar el nivel de madurez de una profesión en términos de ocho componentes:

1. *Formación profesional inicial*. Se refiere a los estudios completados previamente al comienzo de la práctica profesional; es decir, generalmente, los profesionales comienzan a ejercer sus carreras al completar un programa universitario en el campo elegido

2. *Acreditación*. Los programas universitarios deben ser acreditados por organismos de control, que determinan si ofrecen una formación adecuada. Esto busca asegurar que los profesionales que egresen puedan comenzar su vida profesional con los conocimientos necesarios

3. *Desarrollo de habilidades* (competencias). En la mayoría de profesiones no es suficiente con la formación para desarrollar completamente las competencias necesarias, por lo que los profesionales recién egresados necesitan la práctica para aplicar sus conocimientos, antes de estar preparados para asumir la responsabilidad primaria en la realización de sus trabajos

4. *Certificación*. Es un proceso voluntario y administrado por una profesión, que permite determinar quién está plenamente capacitado para participar de un área específica. Una vez se termina la

formación profesional inicial y el desarrollo de habilidades, el profesional debe pasar una o más pruebas para asegurar que ha adquirido un mínimo nivel de conocimiento

5. *Concesión de licencias*. El licenciamiento es similar a la certificación, excepto que es obligatorio y administrado por una autoridad gubernamental. La concesión de licencias tiene como objetivo proteger al público, la vida, la salud, la propiedad y promover el bienestar general. En diversos países la ingeniería es una profesión que tiene licenciamiento, sin embargo, no es obligatorio que todos los ingenieros estén licenciados

6. *Desarrollo profesional*. La formación profesional continua permite mantener o mejorar el conocimiento y las habilidades de los profesionales, luego de haber comenzado la práctica profesional

7. *Sociedades profesionales*. A medida que una profesión se desarrolla tienden a surgir sociedades profesionales. En general, comienzan promoviendo el intercambio de conocimiento; luego evolucionan e incluyen funciones como la definición de criterios de certificación, el establecimiento de estándares para acreditación y la definición de códigos de ética

8. *Código de ética*. Muchas profesiones han definido su código de ética para asegurar que los profesionales ejerzan sus funciones responsablemente

Ford y Gibbs [FOR96] caracterizan cada uno de estos componentes en alguno de los siguientes niveles de madurez:

0. *No existente*: el componente no existe de ninguna forma

1. *Ad hoc*: existe una forma relacionada del componente, pero no se identifica con la profesión dada

2. *Específico*: el componente existe y está claramente identificado con la profesión

3. *Maduración*: el componente ha existido por muchos años, ha estado bajo custodia de un órgano apropiado y se está mejorando continuamente

Además de definir el modelo, para 1996 se pudo determinar que los componentes de la profesión de Ingeniería de Software estaban en el nivel 1. *Ad hoc*, salvo el de Desarrollo Profesional, que estaba en el nivel 2. *Específico*.

4.2 Nivel de madurez actual

Desde la evaluación realizada por Ford y Gibbs la Ingeniería de Software ha tenido avances como profesión en su nivel de madurez. En 1999, McConnel y Tripp [MCC99] analizaron la evaluación realizada en 1996 y estudiaron el estado de los componentes de la profesión para ese año, y en 2004 nuevamente actualizaron su estudio [THA05]. En 2013 se hace una nueva actualización, en la que se presenta el nivel de madurez de los componentes de la profesión [MCC13]. En este informe, los autores actualizan la lista de referencias, e incluyen acontecimientos recientes que influyen sobre estos niveles. En esta serie de artículos de McConnell y Tripp, a diferencia de la evaluación realizada por Ford y Gibbs, no se indica qué valor de las etapas evolutivas le corresponde a cada una de las

componentes. A continuación se presentan los resultados expuestos por estos autores y se incluyen nuevas referencias actualizadas relacionadas a dichos componentes.

4.2.1 Formación profesional inicial

Actualmente, la mayoría de practicantes tienen una carrera de pregrado en Ciencias Computacionales o en Tecnologías de la Información. Aunque el número de egresados de carreras de pregrado en Ingeniería de Software aún es pequeño, en las últimas décadas se ha venido incrementando. A modo de referencia, en 2010 en Estados Unidos existían 244 programas de pregrado, 70 programas online, 230 programas de maestría y 41 programas de doctorado [MCC13]. En 1987, el *Imperial College* de Londres introdujo la primera carrera de pregrado en Ingeniería de Software, y desde entonces se han establecido carreras en diferentes universidades del mundo. La primera maestría fue establecida en 1979 en la universidad de Seattle, y en 1988 el primer programa de doctorado en la *US Naval Postgraduate School* (NPS). Al igual que con los pregrados, actualmente existe una variedad de universidades en el mundo que ofrecen posgrados relacionados [MCC13].

4.2.2 Acreditación

La acreditación de programas es un asunto al que actualmente las instituciones le prestan mayor atención, y cada vez están más interesadas en lograrla [MCC13]. Por ejemplo, el criterio para acreditación de programas en Ingeniería de Software en Estados Unidos se desarrolló entre 1998-1999, y los primeros programas lo lograron en 2003 [HIS09]. Además, se crearon organizaciones para operiacionalizar el trabajo, como Accreditation Board for Engineering and Technology (ABET), dedicada a la acreditación de programas de educación universitaria o terciaria en disciplinas de ciencias aplicadas, Ciencias Computacionales, ingeniería y tecnología. Por este mismo estilo, los países latinoamericanos cuentan con organismos, la mayoría estatales, para otorgar registros y para acreditar programas. La acreditación se ha convertido en un indicador importante para las instituciones, porque de forma muy generalizada se presume que un programa acreditado es de calidad.

4.2.3 Desarrollo de habilidades

Dada la naturaleza dinámica de la tecnología, la industria dedica recursos significativos para la formación de los ingenieros de software [MCC99]. IEEE-CS trabajó en un proyecto para determinar cuál es el conocimiento que debe adquirir un ingeniero de software profesional, y que al final llevó a la elaboración de la guía del cuerpo de conocimiento de la Ingeniería de Software SWEBOK 2004 [IEE04], en el que se definen esos conocimientos y que se utiliza para la construcción de guías curriculares para programas de pregrado y posgrado, así como para definir las pruebas de certificación. Es decir, actualmente se tiene mayor claridad acerca de las habilidades que debe poseer un ingeniero de software, lo que también se ha reforzado en el *Manifiesto por la profesionalización del desarrollo de software*, que publicó en 2013 la Red Latinoamericana en Ingeniería de Software (RedLatinaIS) [SER13].

4.2.4 Certificación

Las certificaciones las toman los profesionales que desean demostrar el dominio en un cuerpo de conocimiento en particular, reconocido por la comunidad [SEI09]. Generalmente existen dos tipos de certificación: 1) las de base general, que se fundamentan en los cuerpos de conocimiento que cubren toda una disciplina profesional o una especialidad dentro de la disciplina, y 2) las específicas de productos, cuyo objetivo es demostrar dominio en un producto o línea de productos específicos [SEI09]. Las primeras son otorgadas por sociedades profesionales y se rigen por normas nacionales e internacionales. Por ejemplo, IEEE-CS ofrece *Certified Software Development Professional* (CSDP) y

Certified Software Development Associate (CSDA) para la Ingeniería de Software. Los candidatos que aplican a este tipo de certificación deben cumplir con requisitos establecidos en experiencia y formación. Comúnmente se realizan evaluaciones, aunque algunos programas de certificación también aplican revisión por pares. Además, para mantener la certificación, la mayoría de instituciones requieren que el profesional certificado demuestre su actividad profesional y que realiza formación continua [SEI09].

En diferentes países se ofrecen otras certificaciones, por ejemplo, en Alemania, *International Software Quality Institute* (iSQI) brinda certificaciones a ingenieros de software profesionales, en especialidades como arquitectura de software, gestión de proyectos y *testing* [SEI09]; en el Reino Unido, *Institution of Electrical Engineers* lleva a cabo un programa de certificación de ingenieros de software, y en Australia, *Australian Computer Society* ofrece un programa de certificación en tecnologías de la información con una sub-especialización en Ingeniería de Software [MCC13]; por otro lado, *International Software Testing Qualifications Board* (ISTQB) ha creado un esquema para la certificación internacional de *testers* de software, y a marzo de 2013 había emitido más de 295.000 certificaciones en 70 países. En lo que refiere a Ingeniería de Requisitos, *International Requirements Engineering Board* (IREB) ha creado *Certified Professional for Requirements Engineering*, que hasta 2013 había otorgado más de 13.000 certificaciones, y *Software Engineering Institute* (SEI) ha elaborado tres certificaciones relacionadas con la arquitectura de software. Las certificaciones relacionadas con un producto, o con líneas de productos, generalmente aplican pruebas para evaluar la familiaridad del candidato con éstos y con su uso [SEI09]. Por ejemplo, Microsoft y Apple brindan certificaciones para sus productos.

4.2.5 Concesión de licencias

La concesión de licencias ha sido un tema controversial para la Ingeniería de Software, porque existen distintas opiniones sobre si el conocimiento actual es lo suficientemente maduro como para poder licenciarla. Es decir, todavía se duda si un profesional con licencia de ingeniero de software tendrá el conocimiento necesario como para proteger al público de los riesgos del software, o si el cuerpo de conocimiento de esta ingeniería es lo suficientemente maduro como para servir de apoyo para cumplir con esta responsabilidad [HIS09]. En Estados Unidos, Texas es el único estado que otorga licencias a ingenieros de software desde 1998 [SPE99], y lo mismo que en Florida también regula el uso de los títulos de *ingeniero en computación* e *ingeniero de software* [MCC13]. Para obtener la licencia de ingeniero de software en este estado se necesita cumplir con los siguientes requisitos [SPE99]:

- Poseer un título de pregrado en Ciencias Computacionales, ingeniería u otras titulaciones de pregrado que la junta considere adecuada
- Al menos 16 años de experiencia comprobada en la realización de trabajos de ingeniería, o que al menos tenga 12 años en una carrera de pregrado aprobada por ABET
- Referencias de al menos nueve personas, cinco de las cuales deben ser ingenieros licenciados
- Credenciales educativas y de otro tipo que se consideren necesarias

Además, desde 2006 se exige que todos los candidatos pasen el examen general *Professional Engineer* (PE). Otros países han procedido con relativa facilidad al otorgamiento de licencias para ingenieros de software, por ejemplo, en Canadá las provincias de Ontario, Alberta y British Columbia

licencian ingenieros de software, y en el Reino Unido los graduados de programas acreditados por *British Computer Society* (BCS), y que posean un nivel de experiencia apropiado, pueden aplicar para obtener el estado de *Chartered Engineer* [30].

Las sociedades profesionales IEEE-CS y ACM han marcado distintas posturas en lo que se refiere al licenciamiento de ingenieros de software, mientras que la primera está a favor, la segunda está en contra. ACM considera que no conviene licenciar a ingenieros de software, porque es prematuro, y que no sería efectivo para hacer frente a los problemas de calidad y fiabilidad del software [BAL09]. IEEE-CS ha estado trabajando activamente en una iniciativa para licenciar ingenieros de software que pasen una prueba de competencias. El examen *Principles and Practices of Software Engineering* comenzó a aplicarse en abril de 2013: lo presentaron 12 personas de las que lo aprobaron seis. Para poder licenciarse como ingeniero de software se deben pasar dos exámenes: 1) *Fundamentals of Engineering* (FE), que ACM no considera interesante para los ingenieros de software porque cubre un espectro grande de temas, y porque generalmente se toma luego de graduarse de programas acreditado por ABET; 2) *Principles and Practices of Sotware Engineering* (PPSE), que es específico de la disciplina y que se puede tomar sólo después de acumular varios años de experiencia laboral en el área. Se debe tener presente que aún en los estados americanos que licencian ingenieros de software, éste sólo se requiere para trabajar en sistemas de software críticos, por ejemplo la salud, la seguridad o el bienestar público. En particular requerirán licencia los ingenieros que ofrecen servicios directamente al público, ya sean profesionales independientes o en pequeñas empresas.

4.2.6 Desarrollo profesional

La formación profesional continua permite mantener o mejorar el conocimiento y las habilidades de los egresados, luego de comenzar la práctica profesional. Estos requisitos ayudan a asegurar que una vez lograda la formación profesional inicial, y luego de haber desarrollado las habilidades necesarias para ejercer la profesión, los egresados mantienen un nivel mínimo de competencia a lo largo de su carrera [MCC13]. Actualmente existen diversas formas de desarrollo profesional para ingenieros de software, como la oferta de educación continua, los programas de especialización, y diferentes conferencias y talleres a nivel mundial.

4.2.7 Sociedades profesionales

IEEE-CS y ACM son las dos sociedades profesionales más importantes relacionadas con la ingeniería de software, y ambas disponen de conferencias, revistas, grupos de interés y bibliotecas digitales; además, trabajan activamente por el establecimiento de la ingeniería de software como profesión [MCC99]. En algunos países de América Latina también se han conformado sociedades con este objetivo, pero la mayoría no pasa de ser una especie de club para diagnosticar la Ingeniería de Software, y no para trabajar activamente en los temas que se consideran importantes. Así mismo, se han estructurado redes de profesionales e investigadores con el objetivo de trabajar en pro de la profesionalización y el reconocimiento de esta ingeniería, como la Red Latinoamericana en Ingeniería de Software (RedLatinaIS).

4.2.8 Código de ética

SWECC fue el responsable de la creación del grupo de trabajo *Software Engineering Ethics and Professional Practice* (SEEP), que desarrolló el Código de Ética y Práctica Profesional [THO01]. En 1998 fue adoptado por IEEE-CS y ACM, lo que representó un hito en la computación, porque fue el primer código de conducta desarrollado por un equipo internacional y examinado a lo largo de su creación por practicantes de todo el mundo. A su vez, este código fue avalado por diferentes cuerpos profesionales y asociaciones en diversos países [ROG02]. Actualmente sirve como apoyo para lograr

una profesión madura, debido a que define en forma explícita las obligaciones éticas de un ingeniero de software [MCC13]. El código está disponible de dos formatos: 1) una versión corta, que resume las aspiraciones a un alto nivel de abstracción y 2) una versión completa, que incluye cláusulas adicionales.

4.3 Evolución de la profesión

En la Figura 3 se presenta la línea del tiempo con los principales hitos y acontecimientos asociados a la Ingeniería de Software como profesión, desde su establecimiento en 1968 hasta setiembre de 2013. En la misma se pueden apreciar los diversos acontecimientos sobresalientes que han sucedido en este tiempo, desde que se acuñó el término por primera vez, y la aceleración importante que se presentó a partir de 1996. Esto indica que la profesión y la formación en Ingeniería de Software recorren un camino de maduración claro y sostenido.

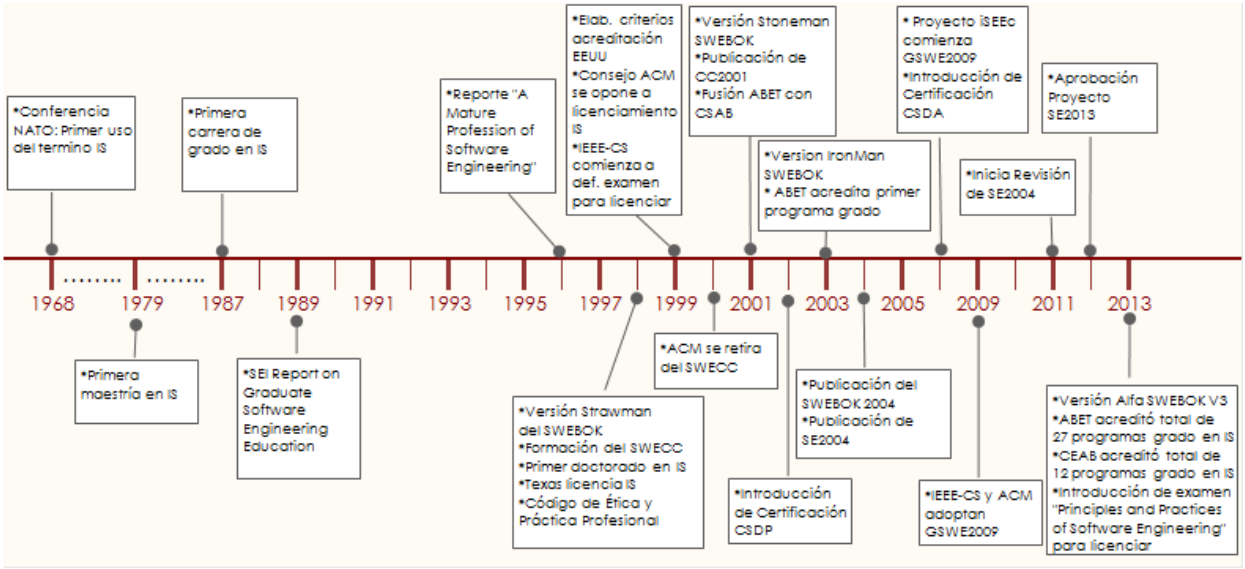


Figura 3. Evolución de la profesión en Ingeniería de Software

5. CONCLUSIONES

En este capítulo se presentó el estado actual de la formación en Ingeniería de Software, y los distintos aspectos relevantes para esta disciplina, a partir del momento en que se acuño el término por primera vez en 1968, y que desde entonces ha evolucionado y madurado de manera sostenida, acompañado por las necesidades que surgen de la academia, la industria y las sociedades profesionales. Varios hitos importantes sucedieron desde entonces hasta el presente, como el desarrollo de la guía para el cuerpo de conocimiento SWEBOK, para establecer qué es lo que tiene que conocer y cuáles habilidades debe tener un profesional de la disciplina. Además, a partir del mismo se pueden elaborar planes de estudios, certificaciones y licencias.

En los últimos años se han elaborado guías curriculares propuestas por IEEE-CS y ACM para la formación en pregrado (SE2004) y posgrado (GSwE009). Ambas guías se basan en SWEBOK, y cada una define un cuerpo de conocimiento central, que mínimamente se debe incluir en los programas

respectivos. Estas guías han sido utilizadas por distintas universidades del mundo para crear nuevas carreras en Ingeniería de Software, y para adaptar y comparar las existentes.

Diversos investigadores y profesionales concuerdan en que aún la disciplina es inmadura en varios aspectos, en parte porque todavía la industria y la academia utilizan *modas* para desarrollar y formar en desarrollo de software. Se debe entender por *moda* al uso de técnicas, métodos, y procesos de software, entre otros, que no han sido efectivamente probados o que se conozcan realmente los efectos de aplicarlos. Explícitamente se refiere al hecho de utilizar algo únicamente porque *otros dicen que da buenos resultados*. De todas formas, esto no se debe concebir como dramático, porque todos los involucrados en la ingeniería debemos comprender que es una disciplina reciente, aunque sus avances y progresos son vertiginosos.

Se puede suponer que los avances en esta profesión serán semejantes a los que se han dado en otras profesiones, pero probablemente de forma más vertiginosa debido a la propia aceleración de la historia (La aceleración de la historia es un concepto tratado, entre otros, por varios autores del materialismo dialéctico. Por ejemplo, Grompone [GRO01] presenta claramente el concepto de aceleración de la historia con datos empíricos). Entre estos avances se tendrá que definir aún más la formación en la ingeniería y en las diversas especialidades de esta área. Por ejemplo, en medicina existen diferentes especialidades relacionadas, pero no todos los egresados son médicos, como los enfermeros, los técnicos en radiografía, los administradores de salud, los especialistas médicos, entre otros. Para la Ingeniería de Software se podría pensar desde ahora en especialidades como arquitectos de software, responsables de calidad de software, probadores profesionales, desarrolladores profesionales, arquitectos de sistemas, técnicos en programación, administradores de bases de datos, y muchos más. No todos tendrían que ser ingenieros de software, y no todos tendrían por qué tener un título universitario. No se puede asegurar que esto vaya a acontecer exactamente así, sin embargo, lo que es seguro es que el futuro, e incluso el futuro cercano, deparará muchos cambios en esta área. Es importante estar preparados, y para eso no hay nada mejor que conocer nuestra propia historia.

REFERENCIAS

[ACM13] ACM/IEEE-CS Joint Task Force (2013). The Joint Task Force on Computing Curricula. Computer Science Curricula (CS2013). ACM-IEEE.

[ACM05] ACM; AIS & IEEE-CS (2005). Computing Curricula 2005 - The Overview Report covering undergraduate degree programs in Computer Engineering, Computer Science, Information Systems, Information Technology, Software Engineering. Association for Computing Machinery & IEEE Computer Society.

[AHM08] Ahmadi, A. et al. (2008). A Survey of Social Software Engineering. In 23rd IEEE/ACM International Conference on Automated Software Engineering - Workshops, 2008. ASE Workshops 2008, 1-12. 15-16 September, L'Aquila, Italy.

[ARD11] Ardis, M. et al. (2011). Advancing Software Engineering Professional Education. IEEE Software, 28(4), 58-63.

[ARD12] Ardis, M. & Henderson, P. (2012). Software Engineering Education (SEEd). Software Engineering Notes, 37(3), 8-9.

[ARD13] Ardis, M. (2013). Revisions to Software Engineering 2004. In 26th International Conference on Software Engineering Education and Training (CSEE&T), 356-358. May 19–21, San Francisco, USA.

[ARD13a] Ardis, M. et al. (2013). Using GSwE2009 in the Creation and Modification of Graduate Software Engineering Programs and Related Curricula. In 26th Conference on Software Engineering Education and Training (CSEET 2013), 109-118. 19-21 May, San Francisco, USA.

[BAL09] Baldwin, K. & Pyster, A. (2009). The Integrated Software and Systems Engineering Curriculum Project: Creating a Reference Curriculum for Graduate Software Engineering Education. Stevens Institute of Technology.

[BLO56] Bloom, B.S. et al. (1956). Taxonomy of educational objectives - The classification of educational goals, Handbook 1: Cognitive domain. Longmans Green.

[BOU14] Bourque, P. y Fairley, R. (Eds.). (2014). Guide to the Software Engineering Body of Knowledge SWEBOK®. IEEE Computer Society.

[CUA13] Cuadros-Vargas, E. et al. (2013). Evolution of the Computing Curricula for Computer Science in Latin America 2013. CLEI, 1-10. IEEE.

[DIN11] Ding, E. et al. (2011). Research and practice on software engineering undergraduate curriculum NJU-SEC2006. In IEEE-CS Conference on Software Engineering Education and Training, 492-496. 22-24 May, Honolulu, USA.

[FOR96] Ford, G. & Gibbs, N. (1996). A Mature Profession of Software Engineering. Technical Report CMU/SEI-96-TR-004. Software Engineering Institute, Carnegie Mellon University.

[FRE06] Frezza, S.; Tang, M-H. & Brinkman, B. (2006). Creating an Accreditable Software Engineering Bachelor's Program. IEEE Software, 23(6), 27-35.

[GRO01] Grompone, J. (2001). La Danza de Shiva - Tomo 5: La Construcción del Futuro. La Flor de Itapebí.

[HIS09] Hislop, G.W. (2009). Software Engineering Education: Past, Present, and Future. In Ellis, H.; Demurjian, S. & Naveda, J.F. (Eds.), Software Engineering - Effective Teaching and Learning Approaches and Practices, 1-13.

[IEE04] IEEE-CS (2004). SWEBOK, Guide to the Software Engineering Body of Knowledge. IEEE Computer Society.

[IEE04a] IEEE-CS (2004). Computer Engineering 2004 - Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering. Association for Computing Machinery.

[IEE90] IEEE (1990). IEEE Standard Glossary of Software Engineering Terminology. IEEE Std. 610.121990. The Institute of Electrical and Eletronics Engineers.

[INV05] Inverardi, P. & Jazayeri, M. (Eds.). (2005). Software Engineering Education in the Modern Age. In Software Education and Training Sessions at the International Conference, on Software Engineering, ICSE 2005, Vol. 4309, 11-27. 15-21 May, St. Louis, USA.

[KEN09] Kennedy, G. et al. (2009). Educating the Net Generation. Melbourne University.

[MCC13] McConnel, S. & Tripp, L. (2013). Software Engineering Professional Practices. In Software Engineering Essentials, Volume II: The Supporting Processes. Thayer, R. & Dorfman, M. (Eds.), 159-164.

[MCC99] McConnell, S. & Tripp, L. (1999). Professional Software Engineering-Fact or Fiction? IEEE Software, 16(6), 13-18.

[MIS11] Mishra, A. & Yazici, A. (2011). An Assessment of the Software Engineering Curriculum in Turkish Universities: IEEE/ACM Guidelines Perspective. Croatian Journal of Education, 13(1), 188-219.

[PAR11] Parnas, D. (2011). Software Engineering - Missing in Action: A Personal Perspective. IEEE Computer Society, 44(10), 54-58.

[RAM07] Ramakrishnan, S. (2007). Accreditation of Monash University Software Engineering (MUSE) Program. International Journal of Issues in Informing Science and Information Technology, 4, 73-89.

[ROG02] Rogerson, S. (2002). The Software Engineering Code of Ethics and Professional Practice: A case for being proactive. In 26th Annual International Computer Software and Applications Conference, COMPSAC 2002, 344-345. 26-29 August, Oxford, England.

[SEI09] Seidman, S. (2009). An International Perspective on Professional Software Engineering Credentials. In Ellis, H.; Demurjian, S. & Naveda, J.F. (Eds.), Software Engineering Effective Teaching and Learning Approaches and Practices, ch. XVIII. Idea Group Reference.

[SER13] Serna, M.E. (ed.) (2013). Manifiesto por la Profesionalización del Desarrollo de Software. Red Latinoamericana en Ingeniería de Software. Instituto Antioqueño de Investigación.

[SOB12] Sobel, A. (2012). Should Everyone Go to College? Computer, 45(10), 82-83.

[SOU09] De Souza, C. et al. (2009). Cooperative and Human Aspects of Software Engineering (CHASE 2009). In 31st International Conference on Software Engineering - Companion Volume, 2009. ICSE-Companion 2009. 16-24 May, Vancouver, Canada.

[SPE99] Speed, J. (1999). What Do You Mean I Can't Call Myself a Software Engineer? IEEE Software, 16(6), 45-50.

[STE09] Stevens Institute of Technology (2009). Integrated Software & Systems Engineering Curriculum (iSSEc) Project. Graduate Software Engineering 2009 (GSwE2009). Curriculum Guidelines for Graduate Degree Programs in Software Engineering.

[THA05] Thayer, R. & Christensen, M. (2005). Software Engineering - The Development Process, Volume 1. Wiley-IEEE Computer Society Press.

[THI13] Thiruvathukal, G. (2013). The Education Issue. Computing Now.

[THO01] Thompson, J. (2001). A Long and Winding Road (Progress on the Road to a Software Engineering. In 25th Annual International Computer Software and Applications Conference, COMPSAC 2001, 39-45. 8-12 October, Chicago, USA.

[VAL12] Vallespir, D. & Camilloni, L. (2012). Uso del Currículo GSwE2009 en la Universidad de la República. IX Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento (JIISIC'12), 7-14.

CAPÍTULO VI

LA RUTA DE LA PROFESIONALIZACIÓN DE LA INGENIERÍA DE SOFTWARE

Red Latinoamericana en Ingeniería de Software (RedLatinaIS)
Medellín, Colombia

PRESENTACIÓN

El propósito de este capítulo es describir la situación, los objetivos y el alcance de una iniciativa para lograr la profesionalización del ejercicio de los ingenieros de software. Aquí es importante aclarar nuestra comprensión acerca de la Ingeniería de Software, ya que es un término con amplio uso pero con diferentes significados para cada persona y situación. En pocas ocasiones la visión global de esta ingeniería ha sido completamente positiva, por lo que es un área temática y una profesión que sufre de una sobrecarga de tecnología y de falta de rigurosidad histórica.

Para la RedLatinaIS, *la Ingeniería de Software tiene por objeto mejorar la práctica del desarrollo de software de alta calidad*. Esta área del conocimiento es el centro de las Ciencias Computacionales, porque el objetivo es generar software de alta calidad que corra eficientemente con un hardware determinado, lo que se convierte además en el interés principal de la investigación en este campo. Paradójicamente, y debido a su importancia, puede ser difícil identificar qué es exactamente, y sobre todo, la forma en que se diferencia de otras áreas de investigación de estas ciencias. Sin embargo, creemos firmemente que todas las áreas científicas, especialmente aquellas en las que es esencial el manejo de información, están altamente relacionadas con la Ingeniería de Software y con sus productos. Además, la sociedad tiene una interacción directa con todo tipo de sistemas y software —de ahí la denominación de *software-dependiente*—, por lo que la calidad y la fiabilidad de estos productos son la piedra angular del desarrollo de software. Pero, al parecer, esta ingeniería se ha convertido en una especie de *patito feo*, porque se le invita a las reuniones familiares pero nunca a una noche de paseo por la ciudad. Varias causas han originado y enfatizado esta situación, como el hecho de que gran parte de la industria y de los *programadores* de software han interpretado a su amaño el *Manifiesto para el Desarrollo de Software Ágil*, y se olvidan de las ventajas que esta iniciativa le aporta al logro de productos a tiempo y dentro del presupuesto [INT09]. Algunas de estas interpretaciones son:

1. *Lo mejor es saltar inmediatamente a escribir código y no preocuparse por escribir requisitos y especificaciones de diseño*. Para ellos esta es una *buena práctica,* teniendo en cuenta que la mayor parte del código que escriben, al faltarles formación adecuada, no tiene el arte ni la esencia científica que se requiere para lograr alta calidad y adecuada fiabilidad. De todos modos, siempre quisieron hacer las cosas a su manera, e *interpretan* que existe un principio que les dice que está bien.

2. *No tenemos que perder el tiempo escribiendo documentos para explicar lo que estamos haciendo*. En todo caso, sino comprenden lo que necesitan, entonces sobra el material técnico.

3. *Estamos seguros de que nuestros empleadores estarán encantados de recibir lo que producimos*, porque les entregamos a tiempo, cuando antes no sabían si iban a conseguir nada en absoluto.

4. *Siempre admiramos a aquellos raros programadores ingeniosos que pueden hackear nuestras bases de código y escribir otro tan brillante que nadie siquiera lo puede comprender*. Ahora tenemos un nuevo nombre para ellos: *codificadores ágiles* (como parodia).

En caso de que no haya quedado claro hasta el momento, estas interpretaciones son muy, muy perjudiciales para la calidad de los productos software. Las personas creen que la Ingeniería de Software trata sólo y exclusivamente del dominio de la sintaxis y las *APIs* de un lenguaje, y que la calidad y la tasa de éxito de los proyectos sigue siendo el mismo año tras año. Creemos que se debe considerar cada metodología y cada principio que aporte al logro de productos de calidad, pero dentro de los lineamientos de un profesión en proceso de maduración, y no como espejos para distraer del objetivo de desarrollar software serio y responsable que satisface las necesidades del cliente y de la sociedad.

Por eso estamos de acuerdo con McConnell [MCC03), cuando escribe que las prácticas necesarias para crear buen software fueron establecidas y están fácilmente disponibles desde hace más de 20 años, pero que han sido manipuladas e interpretadas a la luz de intereses personales, y los productos resultantes, a pesar de algunos éxitos sorprendentes, no están a la altura de las necesidades. Existe un amplio abismo entre las prácticas normales y las mejores, y muchas de uso generalizado son peligrosamente obsoletas y de poca utilidad, por lo que el desempeño promedio de los proyectos software, que se trabajan bajo está mirada, dejan mucho que desear, y diversos desastres conocidos lo constatan. Es necesario que trabajemos para que la comunidad del software comprenda, aplique y experimente los principios de cada propuesta, antes de aplicarla sin razón, porque aunque las buenas prácticas están claras en la teoría desde hace años, pero la puesta en práctica, la contextualización y el compromiso con el que se llevan adelante son el problema principal, y es lo que en parte genera las crisis y las falencias de la industria software, no las tendencias metodológicas en sí.

Por último, creemos que la Ingeniería de Software es una disciplina práctica, por lo que su investigación obliga a participar en el trabajo experimental. Por lo tanto, debemos trabajar con empresas y organizaciones en proyectos reales, a menudo como parte de proyectos de investigación financiados por la academia, la industria o el Estado, pero también en proyectos de consultoría y de formación, incluso dentro de las mismas empresas. Esto nos permite estar en sintonía con sus problemas y poner a prueba nuestras ideas en el campo, porque sólo la investigación, que se valida en el mundo real, se puede decir que demuestra su eficacia.

INTRODUCCIÓN

El software es un elemento clave que se ha involucrado en los principales desarrollos tecnológicos. Además, la rápida evolución de la tecnología informática ha propiciado la definición de nuevos servicios y esquemas de trabajo en las organizaciones, y les ha permitido mejorar la calidad del servicio que ofrecen a sus clientes, a definir nuevos servicios, a conquistar nuevos mercados, y en general a ser más competitivas. La operación de las compañías se encuentra cada vez más soportada en sistemas de información intensivos en software, que son fundamentales para apoyar su liderazgo estratégico en el mercado o, por el contrario, para propiciar su fracaso [ANA06]. Igualmente, el software ha incursionado en los campos del entretenimiento y el hogar, modificando el estilo de vida y abriendo posibilidades a nuevas formas de trabajo y nuevos modelos de negocio.

En relación con otras ingenierías la Ingeniería de Software es una disciplina relativamente joven, pero al igual que para las demás su reto es solucionar problemas en un mundo en constante cambio, y lograr una adecuada relación costo-beneficio aplicando principios matemáticos y de las Ciencias Computacionales [SEI90]. Las características particulares del producto software, la complejidad y la dinámica del contexto, y la rápida evolución de la electrónica y estas Ciencias, le imponen una complicación particular a la disciplina que la diferencian de las demás [MAI97; BRU02]. Hoy en día, el software es reconocido como un sistema socio-técnico que comprende uno o más sistemas, pero crucialmente, también incluye conocimiento acerca de cómo utilizarlo para alcanzar un objetivo más amplio. Este tipo de sistemas incluye a las personas como partes inherentes del mismo, son gobernados por políticas y reglas organizacionales, y se pueden ver afectados por restricciones externas, como las leyes nacionales y las políticas reguladoras [SOM06].

Al analizar el estado actual de la Ingeniería de Software se podrían tomar dos posturas, por un lado está la *pesimista*, planteada desde la conferencia de la NATO en 1968 cuando se habló por primera vez de la *crisis del software*, y que aún hoy muchos consideran que no se ha superado debido los resultados poco halagadores que con frecuencia se observan en los proyectos: incumplimiento sistemático de los plazos de entrega, desfase en los tiempos y presupuestos, y baja calidad del producto final. Pressman [PRE05] define esta problemática como *aflicción crónica*, pero afortunadamente existen movimientos que buscan solucionarla, como Software Engineering Method and Theory (SEMAT). Por otro lado se encuentra la *optimista,* que sin pretender desconocer las problemáticas que actualmente se presentan, busca enfrentar la alta complejidad del desarrollo de software en entornos altamente cambiantes, entiende los desafíos propios de los sistemas socio-técnicos, y reconoce la evolución positiva y los resultados logrados por la Ingeniería de Software hacia su madurez. Tal como lo planteó Ambler [AMB10] en Zurich: "*Somos más éxitos de lo que pensamos; la definición de éxito de un proyecto debe evaluarse en función de su contexto; cada organización define sus criterios de éxito del proyecto que van más allá de los criterios de tiempo, presupuesto y alcance.*"

Actualmente, no es posible concebir al mundo moderno sin software: las infraestructuras nacionales y todas sus utilidades están controladas por sistemas basados en computadores, y los productos eléctricos y electrónicos de control incluyen a un procesador y al software necesario; la fabricación y distribución industrial están completamente informatizadas, y la industria del entretenimiento, como la música, los video-juegos, el cine y la televisión, utilizan software intensivo. Por lo tanto, la Ingeniería de Software es esencial para el funcionamiento de la sociedad de este siglo, pero, debido a que los sistemas software son abstractos e intangibles, y a que no están limitados por las propiedades de los materiales, ni gobernados por las leyes físicas o por los procesos de

fabricación, se pueden convertir en extremadamente complicados, difíciles de entender, y costosos de modificar.

Aunque muchas personas en diferentes contextos escriben programas, como en los negocios para simplificar el trabajo, los científicos e ingenieros para procesar sus datos experimentales, y los aficionados para su propio interés y disfrute, esto no las califica como *desarrolladores*, y en muchos casos ni siquiera como *programadores*. Por otro lado, la mayor parte del desarrollo de software es una actividad profesional, en la que los productos se desarrollan para fines específicos, para incluirlos en otros dispositivos, o como productos de los sistemas de información que están destinados a ser utilizados por alguien diferente a su creador, y que generalmente los desarrollan equipos en lugar de individuos.

Esta situación ha hecho que muchos piensen que *software* no es más que otra palabra que se relaciona con computadores. Sin embargo, cuando se habla de Ingeniería de Software, el término no sólo tiene relación con programas, sino también con la documentación asociada y con los datos de configuración que se requiere para hacer que los programas funcionen correctamente. A menudo, un sistema software es la combinación de varios programas, porque suele consistir de una serie de componentes independientes y de archivos de configuración, utilizados para implementarlos en un entorno de trabajo. Es por esto que si alguien escribe un programa por sí mismo que nadie más va a usar, no tendrá por qué preocuparse de escribir las guías de uso, ni la documentación de diseño; sin embargo, si está escribiendo software que otras personas utilizarán y otros ingenieros modificarán, por lo general tendrá que proporcionar información adicional, además del código del programa. Este es uno de los principios de las actividades de Verificación y Validación (V&V), porque al hablar de la calidad de software se debe tener en cuenta que el producto lo utilizan y modifican personas, diferentes a sus desarrolladores. La calidad no sólo tiene que ver con lo que hace el software, porque también incluye el comportamiento mientras se ejecuta, y la estructura y organización de los demás programas del sistema, lo mismo que la documentación asociada.

Generalmente, los *desarrolladores* de software adoptan un enfoque sistemático y organizado para realizar su trabajo, porque es la manera más eficaz para producir software de alta calidad, sin embargo, la mayoría de *programadores* parece no querer seleccionar un método adecuado, y a veces ni siquiera lo tienen en cuenta, lo que genera una serie de circunstancias a través de *interpretaciones* poco creativas, informales y más rápidas, de las propuestas existentes, que luego aplican en sus proyectos. En conclusión, el desarrollo profesional de software es importante por dos razones:

1. *Los individuos y la sociedad dependen cada vez más de sistemas software avanzados*, por lo que se requiere suficiente capacidad para producirlos de forma fiable, segura, eficiente y fidedigna.

2. Generalmente, *a largo plazo es más económico aplicar métodos y técnicas de Ingeniería de Software* para desarrollar sistemas, en lugar de sólo escribir programas como si fuera un proyecto personal. Para la mayoría de sistemas los costos más elevados tienen que ver con las actualizaciones y modificaciones, posteriores a la implementación.

Por todo lo expuesto, es que mantenemos nuestra posición de que se necesita profesionalizar la Ingeniería de Software; además, porque a diferencia de otras actividades, el desarrollo profesional parece no tener una regulación adecuada en cuanto a requisitos de empleabilidad de sus practicantes, ni de las responsabilidades sociales, y mucho menos de una formación acorde con las exigencias de este siglo.

1. LA INGENIERÍA DE SOFTWARE COMO PROFESIÓN

Por décadas, los investigadores han tratado de encontrar la *bala de plata* necesaria para matar al monstruo de la *generación de software ineficaz*. Estos esfuerzos se reflejan en la continua búsqueda de metodologías, lenguajes, notaciones, y otras fórmulas, casi *alquimistas*, cada una comprometida en resolver el problema, y aunque muchas han hecho aportes sustanciales, todavía no se ha logrado descubrir la bala faltante. Desde hace algún tiempo se inició un movimiento a nivel mundial que busca no encontrar una solución alquimista sino profesionalizar la Ingeniería de Software.

La forma tradicional, y ampliamente extendida, de desarrollar software en diferentes industrias no se puede considerar una actividad profesional, porque en la mayoría de casos sus practicantes no aplican un modelo estructurado, en el que empleen los niveles de destreza y las habilidades necesarios para lograr un producto con las características de calidad que la sociedad requiere. Este es el caso del software comercial, como los sistemas operativos y los ofimáticos, en los que el fabricante, al poco tiempo de salir al mercado, debe ofrecer actualizaciones para corregir errores que se pudieron evitar desde el desarrollo. A diferencia de ingenierías como la Civil, en la que están claramente establecidos los requisitos formales para que sus practicantes se puedan considerar profesionales, la Ingeniería de Software todavía no logra el consenso acerca de las reglas para que su ejercicio se considere profesional. Aunque algunas personas tienen instinto natural para *programar*, sin haber recibido capacitación formal para hacerlo, no quiere decir que tengan la formación necesaria para entregar productos fiables a la sociedad, sobre todo cuando se trata de sistemas críticos, como en las áreas aeroespacial, de la salud y la aviación. Tradicionalmente se acepta que un programador es *bueno* porque entrega el código esperado, pero hacen falta estándares de clasificación, aceptados de forma generalizada en la industria, la academia y el Estado, acerca de qué debería ser un *desarrollador profesional*, y de las características que se deben exigir para considerarlo como tal.

Para ilustrar esta cuestión basta con leer, por ejemplo, la declaración de la misión de Visual Basic para democratizar la programación: …*cualquiera que pueda arrastrar y soltar controles y comprender un mínimo de material técnico podría juntar una solución a un problema con un esfuerzo relativamente pequeño*. Además, existen herramientas y servicios públicos que les permiten a los usuarios crear sofisticadas hojas de cálculo, sin necesidad de conocimientos de programación práctica. Otros han decidido apoyar iniciativas como la *Programación Ágil*, sin tener un conocimiento adecuado de ellas y sus principios, sólo con el objetivo de encadenar servicios en línea sin los elementos necesarios para un desarrollo de software fiable y de alta calidad. Estas interpretaciones, en muchos casos amañadas, perjudican y generan información incorrecta en contra de otros esfuerzos que, continuamente, buscan incrementar la calidad y la seguridad de los productos software.

Por otro lado, la industria de TI es relativamente joven, de hecho sólo ha sobrevivido a un par de generaciones, pero también es una especie de mina de oro que, comparada con otras industrias y de acuerdo con creencias muy generalizadas, es relativamente bien remunerada, no requiere esfuerzos físicos excesivos y se ejerce sin *requisitos mínimos de entrada*. Esto significa que economías enteras han crecido viendo a las TI como un juego de números: *si acumulas suficientes personas alrededor de un problema, es posible que desaparezca, y si esa cantidad no genera costos elevados entonces se puede acumular un montón mayor*. Por esto es que, desde una perspectiva puramente demográfica, es posible afirmar que la mayoría de personas está en esta industria porque: 1) se trata de un trabajo bien remunerado, en comparación con otros empleos de carácter intelectual o incluso manual o 2) no hay incentivos para hacer algo diferente a jugar cómodamente a los números.

Claro que también existen aquellos que optan por sobresalir construyendo buenos productos software, que entienden que desarrollar bien es una habilidad, de hecho, toda una gama de habilidades: comprender y modelar el dominio del problema, entender y aplicar los lenguajes de programación, las librerías, los paradigmas, los idiomas, elegir qué aplicar en una situación dada, aprender y desarrollar algoritmos, comprender y dominar las fases de la Ingeniería de Software, automatizar procesos, conocer las teorías esbeltas para el suministro, la producción y el desarrollo de productos, aplicar la concurrencia y el paralelismo, destacar las bondades y potencialidades de las metodologías recientes, y así podría seguir la lista. De alguna manera, estos profesionales se quieren diferenciar, pero persisten algunos interrogantes: ¿cómo potencializar sus habilidades y capacidades? ¿Cómo pueden ayudar a otros en la industria a que tengan verdadero aprecio por el software que escriben? Las respuestas pueden ser que se necesita algún tipo de modelo de formación y una manera de identificar estas fortalezas, tanto entre los principiantes como en los experimentados. La cuestión es que en muchos casos el software se valora por la utilidad que proporciona, y no importa lo *feo* que sea internamente, siempre y cuando se entregue a tiempo. Un programador de software puede construir un producto que *enamora* desde sus interfaces de usuario, pero lo que verdaderamente importa es que su interior, el *código*, se haya construido de acuerdo con las necesidades del cliente y con la seguridad y fiabilidad necesarias. *Ese es el producto de un desarrollador profesional*.

1.1 Desarrollar software es ciencia y es arte

Desarrollar software es ciencia, porque en el proceso se deben considerar los principios de las Ciencias Computacionales y de la Ingeniería, y es arte, porque son tantas y tan diversas las variables involucradas en el desarrollo del producto que no se puede tener sólo un proceso totalmente prescriptivo y repetitivo. De acuerdo con Brooks [BRO86], debido a las complejas relaciones humanas de comunicación y colaboración, en el desarrollo de software se articulan aspectos esenciales que corresponden a los elementos inherentes y abstractos relacionados a la naturaleza del producto, y aspectos accidentales correspondientes a las dificultades para llevar a cabo el desarrollo mismo, es decir, a lo que Sommerville [SOM06] clasifica como sistemas socio-técnicos.

En el arte, los productos tienen una belleza intrínseca en sí mismos. Una catedral realmente es una cabaña grande para que las personas se reúnan con un fin, y generalmente se construye de piedra para que dure más que una cabaña de madera, pero ¿por qué todo el material decorativo es de lujo? Por supuesto que está ahí para generar un sentido de grandeza y de imponencia, y llama la atención de aquellos que aprecian la belleza y magnificencia, por lo que entran con reverencia y humildad, listos para el culto. Lo que la hace arte es el trabajo y la belleza estética, por encima de su utilidad intrínseca. Entendemos el arte como un proceso creativo individual o colectivo y una profesión disciplinada, donde los autores impregnan en la creación más que su intelecto, suman su alma y su impronta.

Existe una diferencia entre la mentalidad de un escultor, que esculpe la expresión en la cara de una gárgola, y un maestro de obra, que utiliza bloques básicos para construir un parqueadero de varios pisos. En este caso, lo que menos se necesita es que el maestro esculpa su personalidad en la obra, de tal forma que los bloques tengan diferentes tamaños y que no sean intercambiables, lo que interesa es la funcionalidad. Cuando esto sucede, es decir, cuando el maestro de obra pierde de vista el objetivo de lo que está construyendo y coloca su firma, puede que logre una magnifica representación artística, pero que no responde a las necesidades del cliente. Está claro entonces que la Ingeniería de Software necesita más que esto, porque sobre todas las cosas, el arte, como cierto y

de calidad, implica un compromiso y un resultado que trasciende aspectos básicos y cotidianos. Esto no implica dejar de lado las cuestiones técnicas ni el predominio de la funcionalidad, porque el arte es una manera de hacer las cosas a nuestro entender y no sólo la búsqueda de la belleza, que es un concepto temporal y altamente subjetivo.

En los años 60, la comunidad del software aplicaba metáforas mecánicas para el proceso del desarrollo, pero hoy la Ingeniería de Software es un área académica aceptada y un campo activo de investigación. La perspectiva de esta ingeniería para ejecutar proyectos se define como la aplicación de un enfoque disciplinado, cuantificable y sistemático para desarrollar, operar y mantener productos software, es decir, es la *aplicación de la ingeniería al software* [IEE90); un enfoque que ha demostrado ser eficaz en el desarrollo de sistemas críticos de seguridad.

El software suficientemente *bueno* es la extensión lógica de las ideas de la Ingeniería de Software, y de quienes la practican, con un componente artístico adjunto; es una labor de ingeniería en la que se intercambian recursos, calendarios, características y defectos. Por ejemplo, la seguridad en el software del transbordador espacial es una cuestión crítica, por lo que se tienen que minimizar los defectos, y las cuestiones artísticas pasan a segundo plano; pero las aplicaciones comerciales, como navegadores y editores gráficos, requieren una cantidad de características que se deben desarrollar rápidamente, y la seguridad no es tan incidente, pero asumen protagonismo cuestiones como combinación de colores, tipos de letra y la distribución del escritorio, es decir, la parte artística; además, los recursos son limitados por la necesidad de obtener un beneficio, por lo que su desarrollo se realiza pensando en la reducción de costos y de tiempo. Ambos enfoques de trabajo generan un mismo producto, software, pero con niveles diferenciados en seguridad, fiabilidad, consistencia, ciencia y arte. Por eso es que el desarrollo de software es tanto ciencia como arte: *es una profesión* con igual o mayores responsabilidades que la Ingeniería Civil.

1.2 Profesiones y profesionales

Lo primero en este proceso es comprender *qué es una profesión* y *qué es un profesional*, aunque la búsqueda de una definición universalmente aceptada para la primera es en sí misma una empresa prácticamente imposible. Mientras que las sociedades occidentales apuntan a la medicina como el ejemplo más sólido de profesión, otras no tienen una visión unificada. El propósito aquí no es entrar en discusiones sin sentido, sino aclarar algunos conceptos de acuerdo con opiniones generalizadas.

En los diccionarios se define *profesión* como una ocupación, especialmente una que requiere formación avanzada. Desafortunadamente, utilizar el término como sinónimo de *ocupación* es superficial y simplista, y no aporta beneficios para el objetivo buscado. Otros la definen como una *vocación,* cuya práctica se basa en la comprensión de la estructura teórica de algún área del saber o de la ciencia, y sobre las habilidades que la acompañan, y otra definición dice que son grupos que aplican *conocimientos especiales* al servicio de un cliente. Estas últimas definiciones presentan deficiencias, porque casi todas las ocupaciones modernas, basadas en servicios, involucran algún conocimiento teórico, y pierden su significado distintivo al calificarlas como profesiones. Una profesión tiene que ser algo más que una reunión de proveedores de servicios.

En los años 70 se intentó definir el término con base en el control que tenían las asociaciones sobre sus miembros y el mercado, y se creía que la autonomía profesional era un índice útil para valorar la situación relativa de las diversas ocupaciones profesionales. El consenso era que la distinción más estratégica reside en la autonomía legítima y organizada, y que una profesión era

diferente de otras en el sentido que se le ha dado el derecho de controlar su propio trabajo. Posteriormente, esta cuestión tomó un enfoque diferente, cuando se discutió si la odontología podía ser considerada como una profesión, y entonces se argumentó que una característica común a todas las profesiones es la *promesa de altruismo que hacen con la sociedad*. La asociación de una profesión con la autoridad moral y el bien público se basa en que los profesionales tienen el deber, no sólo el derecho, de hablar sobre cuestiones relacionadas con su experiencia para alentar a los ciudadanos a tomar decisiones informadas.

Una definición con amplia acogida dice que una profesión es un colectivo de proveedores de servicios expertos, que conjunta y públicamente se comprometen a priorizar las necesidades existenciales y los intereses del público, al que sirven sobre los suyos propios, y que a su vez reciben la confianza de éste para que lo hagan. Cuando una ocupación es reconocida como una profesión se firma un contrato social entre sus miembros y el público en general, y se sustenta en la confianza, el respeto, la condición social, el monopolio de la práctica y el derecho de auto-gobierno. Otra manera generalizada de definir una profesión es mediante el estudio de las características de las ya reconocidas, y examinar si la ocupación que se evalúa posee los mismos rasgos. Este enfoque, también conocido como de los rasgos o inventarios, se ha ampliado y utilizado para clasificar las ocupaciones en profesiones o semi-profesiones mediante el análisis a la ausencia, o la presencia, de ciertos atributos. Desafortunadamente, este método tiene una desventaja evidente: *los atributos se pueden convertir en una cortina de humo que sólo permite observar la apariencia y no el fondo de la cuestión*.

Como se puede observar el término *profesión* tiene varias definiciones: 1) es una *vocación* que requiere conocimiento especializado y una larga e intensiva preparación académica, y su práctica se basa en una comprensión de la estructura teórica de alguna área científica y de las habilidades que acompañan esa comprensión, 2) es una *capacidad* principal, una vocación, o un empleo y 3) es todo el *cuerpo de personas* que participan en una vocación. Por desgracia, la segunda definición describe con mayor precisión lo que actualmente se acepta para la Ingeniería de Software, sin embargo, un creciente número de organizaciones, personas e instituciones están convencidas de que la primera definición es la más apropiada, y están trabajando por lograr su aceptación como tal. Estos movimientos han descubierto que asumir un enfoque ingenieril para el ciclo de vida del software es menos caótico, produce mejores productos y también es rentable.

Ahora bien, ¿qué pasa con el término *profesionales*? En pocas palabras, los profesionales son miembros de una profesión reconocida, independientemente de si individual o colectivamente se comportan profesionalmente. Pero una mirada más profunda a esta definición, aparentemente obvia, revela dificultades elementales; por ejemplo, la medicina es una profesión, pero ¿todos sus practicantes son automáticamente profesionales? Los médicos que tienen sus licencias suspendidas, que continúan en ejercicio y diagnostican pacientes y prescriben medicamentos, dejan serias dudas acerca de si encajan como profesionales. Hay quienes piensan que debe ser la sociedad quien decida qué ocupaciones califican como profesiones, pero en lo que tiene que ver con la responsabilidad social y el bienestar de las comunidades debe ser el Estado el que decida acerca de estas cuestiones. Esto se puede lograr a través de licencias o certificaciones que emita un organismo oficial o un consejo independiente, las cuales, dentro de una profesión reconocida, trazan la línea entre los profesionales de confianza y los practicantes.

Entonces, al igual que el término *profesión*, un profesional puede ser: 1) alguien que se dedica a una ocupación o actividad profesional, es decir, alguien que se ajusta a los estándares técnicos y

éticos de una profesión y 2) persona que ejerce una actividad por la cual recibe un beneficio económico. Si se tiene alguna duda acerca de que la segunda definición es la que refleja con mayor precisión a la mayoría de profesionales del software de hoy, basta con preguntarle a alguno de ellos acerca de las normas técnicas o éticas de su *profesión*.

1.3 Profesionalizar

Típicamente se acepta que profesionalizar es organizar una ocupación dentro de la forma asumida por los paradigmas de profesiones establecidas, como la medicina. En otros términos, es el proceso para promover una ocupación a una profesión, en el que se busca mejorar las habilidades de una persona con el fin de hacerla más competitiva en asuntos relacionados con su área de conocimiento. Los beneficios se reflejan al trasladar una serie de recursos —especialmente conocimientos y habilidades— a otros, que originarán recompensas sociales y económicas —para los profesionales— . A medida que una profesión adquiere mayor movilidad social y control de mercado podrá reflejar sus habilidades, experiencias y normas éticas. Esto se logra con productos fiables y seguros, que les permiten a sus miembros lograr el reconocimiento de la sociedad y afianzar su labor como profesión reconocida.

El trabajo de los profesionales se basa en proyectos, lo que implica que deben tener un principio y un fin claros, y un proceso metodológico de acciones deliberadas, planeadas e implementadas por los integrantes de los equipos. Esto es precisamente lo que se propone en este capítulo, porque el fin es obtener las recompensas que acompañan a la condición de que la Ingeniería de Software se reconozca como profesión. Para lograrlo, el primer paso consiste en llegar a un consenso acerca del cuerpo de conocimiento, trabajo que se viene desarrollando desde varias organizaciones. El segundo paso es legitimar la autoridad profesional, lo que consta de tres componentes distintivos: 1) que el conocimiento y la competencia de los profesionales hayan sido validados por una comunidad o por sus pares o iguales, 2) que ese conocimiento validado descanse en fundamentos racionales y científicos y 3) que el juicio profesional y el asesoramiento estén orientados hacia un conjunto de valores sustantivos, como la salud, en el caso de la medicina. Estos aspectos de legitimidad corresponden a los tipos de atributos, colegiales, cognitivos y morales, usualmente incorporados en el término *profesión*. En este sentido, la Ingeniería de Software como profesión se caracteriza por:

1. Una formación profesional inicial en un programa de estudios, validado por la sociedad a través de un registro y una acreditación de calidad

2. Registro de la aptitud para la práctica, mediante una certificación voluntaria o licencia obligatoria

3. Desarrollo continuo de habilidades especializadas y de formación profesional

4. Apoyo público a través de una asociación profesional

5. Compromiso con las normas de conducta prescritas en un código de ética

Además, para el reconocimiento de una profesión es necesario articular un conjunto de conocimientos, porque esto representa un amplio consenso respecto a lo que debe conocer un profesional. Este consenso también es un requisito previo a la adopción del desarrollo de competencias coherentes y de programas de formación continua. La profesionalización se puede definir, aunque vagamente, como el establecimiento de, y la adhesión a, un modelo profesional que no existe o no se ha arraigado con firmeza. En este modelo los individuos, que se refieren a sí mismos

como profesionales, exhiben la mayoría de, si no todas, las características de los profesionales del área específica. Además, existe una *atmósfera profesional,* establecida conjuntamente, para los miembros individuales y para las asociaciones profesionales existentes.

El proceso para la profesionalización se realiza en una serie de fases, que no siempre se ejecutan de forma secuencial, es decir, algunas pueden ocurrir simultáneamente y otras lo hacen fuera de orden, pero siempre aprovechan el trabajo previo. La siguiente es una lista de esas fases:

1. *Reconocimiento de la necesidad de la profesionalización*. Esto suele ocurrir cuando se tiene y se siente la necesidad de unos individuos, altamente calificados y uniformemente capacitados, es decir, cuando el trabajo que realizan quienes ya están en esa área del conocimiento se convierte en fundamental y crítico en la naturaleza de las sociedades, en este caso, el software.

2. *Definición formal de la profesión*. Esto incluye establecer una terminología normalizada para la profesión, creación de títulos y descripciones para los miembros, y la identificación de relaciones, por ejemplo, profesión con profesión, profesional con cliente y profesión con público en general. Algo en lo que se pueden encajar iniciativas como SWEBOK.

3. *Identificación de asociaciones profesionales clave*. Se espera que estas sociedades hayan alcanzado cierto grado de estatus formal, por ejemplo, el respeto de la comunidad profesional, la publicación de buenas revistas y la realización de reuniones locales y nacionales. Además, serán muy valiosas para avanzar en el resto del proceso de la profesionalización. Asociaciones como ACM e IEEE ya han adelantado estas cuestiones.

4. *Establecer criterios mínimos de ingreso*. Esto debe incluir temas como cierto grado de formación formal, experiencia demostrable y una certificación. La mayoría de asociaciones en el mundo tienen sus reglamentos bien establecidos.

5. *Establecer y reconocer programas de formación y de entrenamiento*. Que incluyen programas de estudios universitarios existentes y programas profesionalizantes de formación continua aprobados. Los Ministerios de Educación y los organismos de control tienen esto cubierto.

6. *Establecer procedimientos formales de certificación y renovación*. El proceso de certificación se debe basar en la mínima cantidad de habilidades y conocimientos útiles que necesita un profesional tipo. El proceso de renovación de la certificación se debe dirigir hacia las trayectorias profesionales de los asociados. Obviamente, los procesos de certificación y formación se afectan directamente el uno al otro, y para el caso de la Ingeniería de Software, tal vez sea una de las fases que mayor trabajo requiere en este momento.

7. *Recolección y promulgación de estándares profesionales*. Estos estándares cubren temas como procedimientos, metodologías, métricas, niveles aceptables de rendimiento y herramientas. Actualmente, se tienen diversas propuestas, pero es necesario unificarlas y aceptarlas ampliamente.

8. *Identificar y divulgar un código de ética profesional*. Se debe establecer un código de ética que respeten todos los miembros, y lo ideal sería que se incorporará en los procesos de certificación. Para la Ingeniería de Software ya existe un código ampliamente aceptado.

9. *Establecer objetivos de rendimiento cuantificables para la profesión y los profesionales*. Se espera que los profesionales satisfagan y superen un conjunto de objetivos de rendimiento para mantener su estatus profesional, para lo que deben conocer qué tan bien se están desempeñando en su área de conocimiento. La profesión se debe esforzar continuamente para mejorarse a sí misma como un todo, por ejemplo, disminuyendo la tasa de error promedio por miembro. Ya existen criterios, como las buenas prácticas, que se podrían estructurar para alcanzar este objetivo.

10. *Identificar los requisitos y procesos de formación continua*. Las profesiones no son estáticas, especialmente las relacionadas con TI, y la vida útil de los conocimientos profesionales continúa disminuyendo. Por ejemplo, se dice que el conocimiento técnico de la raza humana se duplica cada seis meses, por lo que un profesional está obligado a tomar continuamente cursos de actualización para mantener su estatus. El protagonismo aquí lo deben asumir las universidades, porque la Ingeniería de Software requiere permanentemente de especializaciones y cursos de actualización.

11. *Identificar las responsabilidades y obligaciones profesionales*. Los profesionales deben ser conscientes de sus responsabilidades para con sus clientes, sus empleadores, otros profesionales y con la profesión en general. Además, deben ser consecuentes con las obligaciones que esas responsabilidades generan. Sólo las personas que están legalmente dementes no son responsables por sus acciones. Será el Estado quien legisle a este respecto.

12. *Establecer mecanismos legales de ingreso y de permanencia*. El estado de una profesión se ve disminuido por la inclusión de personas que no cumplen con las normas establecidas, y por las violaciones que sus asociados cometen. El Estado debe reglamentar el ejercicio profesional, que a su vez influye para que las asociaciones delimiten los mecanismos.

13. *Establecer y mantener una imagen pública positiva*. Una profesión con este tipo de imagen puede conseguir mejores beneficios para sus miembros, incluidos niveles salariales más altos. Se debe hacer público el trabajo y los logros de los profesionales en Ingeniería de Software, tanto los positivos como los errores, para que la sociedad mantenga una imagen reciente de los beneficios de esta profesión.

Obviamente, aunque lograr todo esto toma tiempo, se puede llevar a cabo en un período relativamente corto, por ejemplo, unos tres o cuatro años, tiempo que se puede reducir mediante un esfuerzo concentrado de parte de los propios profesionales. En el caso de la Ingeniería de Software, y como se evidencia en las fases, ya se han hecho importantes avances en pro de la profesionalización.

1.4 Responsabilidad profesional de los ingenieros de software

Las fases en pro de la profesionalización de la Ingeniería de Software han evolucionado, aunque no de la forma esperada. Inclusive, actualmente parecen estar en retroceso debido a la falta de unificación de criterios y a la necesidad de desmontar los intereses individuales. Las consecuencias se reflejan en proyectos con sobrecostos, pérdidas por incumplimiento, disminución en los ingresos, mayores costos de oportunidad, baja moral del personal y mala calidad de software en general. Desafortunadamente, poco se está haciendo para cambiar este *statu quo*, y la industria continua contratando *programadores*, poco cualificados y a bajos salarios, mientras que los *desarrolladores* experimentados pierden apreciación en la industria y la academia.

Diversas investigaciones han concluido que hasta el 80% de los proyectos software no llegan a terminarse, o son rápidamente retirados y sustituidos por otro de *última y mejor* versión, sólo para continuar una tradición de pérdida de oportunidades, expectativas no satisfechas y desgastes monetarios. Se puede decir entonces que el desarrollo de software continua en crisis, pero estamos convencidos que con un poco de orgullo y compromiso de parte los involucrados en esta industria será posible alcanzar su recuperación. De acuerdo con estas apreciaciones, a continuación se describen algunas responsabilidades profesionales de los ingenieros de software:

1. *Comprender los requisitos reales del proyecto y mantenerse centrado en ellos*. Pocos proyectos software tienen éxito si previamente no se han definido y documentado las fronteras y los límites del entorno en el que se ejecutarán, y las necesidades de los usuarios que los utilizarán. En ausencia de esta base, se tomarán muchas y malas decisiones de parte del equipo de desarrollo, y como resultado, el producto tendrá prioridades inadecuadas y soluciones mal ajustadas. La comunicación abierta, honesta y frecuente con el usuario es esencial para recoger esta valiosa información, desde el principio en el ciclo de vida del proyecto.

2. *Comprender quién es el cliente*. El cliente para un proyecto dado no siempre es una persona o un grupo obvio, por lo que el ingeniero debe tener la habilidad necesaria para identificarlo y reconocerlo, en beneficio del proyecto y del futuro producto. Es importante tener en cuenta quién es el cliente a fin de mantener la perspectiva correcta de las prioridades del proyecto.

3. *Mantener comunicación abierta, frecuente y directa con el cliente*. Comúnmente, el enfoque que aplican los equipos para construir productos software es contratar el desarrollo con una empresa, del mismo modo que un inventor contrata a un fabricante para manufacturar su producto. Irónicamente, con el software, este enfoque regularmente falla, debido a una comunicación deficiente entre el cliente y el equipo de desarrollo. Los ingenieros de software profesionales tienen la experiencia y el conocimiento para hacer las preguntas correctas, para determinar los requisitos adecuados, para proporcionar información de calidad y oportuna, y para realizar el asesoramiento y la consultoría necesarias para mantener el proyecto en marcha.

4. *Producir un diseño para todos y no sólo para proyectos simples*. Un programa no especificado no puede ser incorrecto, sólo puede ser sorprendente.

5. *Generar un plan para implementar el diseño*. Sin un plan es posible que no se alcancen los objetivos, que se pierda el rumbo y que se elimine cualquier posibilidad de alcanzar pruebas significativas.

6. *Elegir las mejores herramientas para el trabajo, no las más populares,* y de acuerdo con las necesidades del proyecto. Las nuevas tecnologías se deben elegir por la oportunidad de aprender, no porque sean las mejores para el trabajo. A menudo, la tecnología subyacente que se utiliza para poner en práctica un proyecto se elige por razones equivocadas, y otras veces sólo por ser la más popular del momento. Estos criterios son comunes, pero perjudiciales para el éxito potencial del proyecto. No se debe elegir una herramienta sin una revisión exhaustiva de los requisitos.

7. *Mantener un alto nivel de profesionalismo*. La formación permanente es fundamental para lograrlo.

8. *Mantener la disciplina necesaria para alcanzar estándares consistentes de codificación y de prueba*. Los estándares de desarrollo y de prueba son necesarios para ahorrar tiempo y dinero a largo plazo, y conducen a tener menos errores, que generalmente son más fáciles de rastrear y de

resolver si se comienza a probar en las fases tempranas del ciclo de vida, por lo tanto se acorta el tiempo de desarrollo global y el del ciclo de prueba.

9. *Ejecutar pruebas completas y adecuadas al código* para asegurar los más altos niveles de fiabilidad y facilidad de mantenimiento en el producto. Uno de los objetivos de la Ingeniería de Software es generar productos con la calidad y fiabilidad suficientes para que la sociedad los acepte y utilice, pero para lograrlo se debe implementar un adecuado plan de pruebas.

10. *Comprender que la fase de mantenimiento es la más cara de cualquier proyecto software*. Por lo que se deben seguir estándares que ayuden a reducir este costo, mediante una documentación adecuada y de comentarios precisos en el código.

1.5 Recomendaciones para alcanzar la profesionalización

La Ingeniería de Software se ocupa de la creación y de la aplicación de fundamentos ingenieriles para el desarrollo de software y del uso intensivo de productos técnicos, procesos que se logran mediante un análisis sistemático y trabajo en equipo. El concepto de la formación de ingenieros de software profesionales requiere sujetos, secuencias sistemáticas y flexibilidad para hacerles frente a las necesidades del mercado, a los requisitos de la industria y a los rápidos y abundantes desarrollos tecnológicos. Uno de los objetivos debe ser integrar conceptos como formación, formalización y profesionalización en todos los ámbitos relacionados con el desarrollo de software. Aunque no se puede asegurar que el simple seguimiento de un algoritmo sea suficiente para alcanzar la profesionalización de un área de conocimiento, a continuación se plantean unos pasos que pueden ayudar a lograrlo:

1. *Iniciar el proceso de una vez*, con el objetivo de alcanzar resultados positivos visibles en no más de dos años. Específicamente, las funciones del desarrollador y del administrador promedio se deben ver afectadas por el proceso de profesionalización antes de esa línea de tiempo.

2. *Invitar a muchos y diversos actores a participar y a opinar*. Incluyendo a desarrolladores, analistas, gestores, profesores, gobierno, sociedades profesionales y a profesiones establecidas, entre otros.

3. *Dar a conocer el avance del proceso*, y mantenerlo visible y asequible para todos los interesados, de forma que con sus opiniones y aportes se enriquezca y perfeccione en todo momento.

4. *Establecer un comité de mantenimiento para la profesionalización*, cuyo trabajo incluirá el seguimiento a los cambios en el proceso e introducirlos de manera ordenada, y actuar como un *tribunal supremo* de las controversias que surjan.

5. *Fomentar el establecimiento de planes de estudio* para el programa de Ingeniería de Software. Por otra parte, promover el concepto de profesionalización en todos los niveles de formación relacionados con el software.

2. SITUACIÓN ACTUAL DE LA PRÁCTICA PROFESIONAL

Como se ha mencionado antes, el software se relaciona con todos los aspectos de la vida cotidiana: manufactura, banca, viajes, comunicaciones, defensa, medicina, investigación, gobierno, educación,

entretenimiento, leyes, entre otros. Es parte esencial de los sistemas militares, y se utiliza en diferentes sectores civiles, incluyendo a los de misión crítica. Desafortunadamente, los *sistemas de educación* parecen no seguirle el paso a los cambios y desarrollos que originan esta relación, por lo que los programas de ciencia e ingeniería, en pregrado y posgrado, necesitan incorporar mayor y mejor capacitación en Ingeniería de Software.

El diseño y la construcción de software debe seguir algunos procesos y procedimientos que se aplican en otras disciplinas ingenieriles [SOM06; GLA06]. En primer lugar, definir cuidadosamente los requisitos, para luego desarrollar la arquitectura del sistema y comenzar el desarrollo de código, a la vez que paralelamente se diseña e implementa el plan de pruebas. De acuerdo con el tipo de modelo de ciclo de vida utilizado, y el tipo de sistema desarrollado, existen diversas formas de llevar a cabo estos pasos, pero en el proceso también es necesario considerar los costos, interactuar con las personas y monitorear las responsabilidades del equipo de desarrollo. Si bien este proceso sigue los mismos pasos comunes para diseñar y construir cualquier sistema complejo, como puentes, aviones o equipo informático, la naturaleza intangible, maleable y cambiante con alta dependencia del contexto de su producto, obliga a trabajar con una alta flexibilidad entre las fases. En ese sentido, es difícil asimilar el desarrollo de software a una línea de producción, donde todo está controlado, porque la *fabricación* de cada producto software sigue una línea de desarrollo difícilmente repetible en otro proyecto [SOM06].

De acuerdo con la National Academy of Science [JOR06], el software no es más que un artículo de comercialización básico que, de hecho, incorpora en sí mismo a la función productiva de la economía; pero, a pesar de la capacidad de los productos generados, el National Institute of Standards and Technology (NIST) estima que sus errores tienen un alto costo para la economía de los países [NIS10]. Aun así, la formación en Ingeniería de Software en el mundo no recibe la atención que merece, a pesar de esta importancia manifiesta para la economía. Para ayudar a disminuir estos problemas se requieren ingenieros de software mejor formados, a través de programas profesionales rigurosos, que estén al mismo nivel de cualquiera de las ingenierías con mayor historia. El asunto del reconocimiento de la Ingeniería de Software como una profesión no es nuevo, y en los últimos años los debates se han incrementado [IEE04; VAU00). IEEE y ACM la definen como la *aplicación de un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento de software* [IEE04]. Un buen resumen acerca de esta cuestión se puede encontrar en Cook y Dupaix [COO99].

2.1 Naturaleza del proceso del software

El estudio de esta cuestión se inició en los años 80 como una línea de trabajo autónoma, y actualmente es uno de los elementos de la Ingeniería de Software más complejos de entender y analizar, entre otras porque existen diferentes variantes, niveles de abstracción y connotaciones para lograrlo:

- *Modelos de ciclo de vida*, que plantean guías generales de las maneras como puede evolucionar un producto durante su desarrollo

- *Métodos o técnicas*, que definen formas de trabajo para tareas con un alcance controlado

- *Marcos metodológicos*, que recopilan un conjunto de técnicas y definen roles, guías, lineamientos y artefactos para realizar diversas actividades relacionadas con el desarrollo de software

- *Modelos de procesos de alto nivel*, que recopilan un conjunto de prácticas reconocidas por comunidades y especialista. En esta categoría se reconoce el esfuerzo, aunque con intencionalidades diferentes, de entidades como el Software Engineering Institute y el European Software Institute, y de entidades de estandarización como la ISO, que definen modelos y estándares de calidad que representan referentes internacionales, como CMMI®, METRICA, ISO12207 e ISO15504.

En tal sentido, uno de los hitos que marcaron un nuevo rumbo a la concepción de este proceso fue el *Movimiento Ágil*, que plantea una reacción justificada a los métodos robustos y a la burocracia de los marcos metodológicos existentes, y que rescató el valor de las personas como los principales actores del desarrollo de software [AGI01]. Afortunadamente, las tendencias de modelos y estándares robustos y los enfoques ágiles, que parecían ir por rumbos diferentes e irreconciliables, han madurado y buscan acortar las brechas. Hoy se conocen resultados positivos de empresas que han adoptado enfoques ágiles, a la vez que se han articulado al esfuerzo de madurez y mejora de procesos con CMMI, surgido inicialmente para atender las necesidades de las grandes empresas [NAV10]. También se realizan esfuerzos para escalar otros enfoques, desde los asociados con equipos y proyectos pequeños, a proyectos más complejos [AMB09].

Uno de los retos propuesto por Boehm [BOH11] está justamente asociado a dos características en el proceso software: 1) *capacidad para generar resultados a corto plazo* y 2) *capacidad para adaptarse a las condiciones particulares de un proyecto*. Surge entonces el interrogante de ¿cómo darle al proceso las características de agilidad, flexibilidad y adaptabilidad, sin perder el control sobre el objetivo final, y generar una solución software con los niveles de calidad y satisfacción esperados, permitiendo a la vez una disciplina de trabajo orientada hacia la madurez del equipo y de las organizaciones de software? Para buscar una respuesta, los esfuerzos de investigación alrededor del proceso se orientaron bajo la premisa de que el *proceso software también es software* [FUG00]. Es decir, el problema de modularidad y reusabilidad, que ha derivado nuevos enfoques de desarrollo, es el mismo que se debe enfrentar desde la perspectiva del proceso software. En lugar de definir metodologías pesadas y poco flexibles, que difícilmente se pueden aplicar en todos los proyectos, se requiere definir activos software que describan formas de trabajo con un nivel de granularidad controlado, y que se puedan integrar y adaptar a un proyecto particular.

Visto de esta manera, la línea base del proceso software, que se aplique en un proyecto particular, surge como una estrategia de composición a partir de activos software, que son definidos y almacenados en bibliotecas. Esta visión del proceso ágil y flexible a partir de componentes predefinidos, es el principio que rige líneas de trabajo como la Ingeniería del Método Situacional [MIR06] y la Ingeniería del Proceso Software [RUI07], además, es la motivación básica sobre la cual se soportan.

2.2 Las prácticas en la Ingeniería de Software

Uno de los elementos centrales que facilita esta nueva perspectiva del proceso, y que promete la articulación de enfoques que parecían irreconciliables, es el concepto de las prácticas, concebidas como *formas de trabajo propuestas para atender una necesidad particular*. Los modelos de calidad, como CMMI, ISO-SPICE e ISO9000, han articulado un conjunto de prácticas, consideradas útiles y reconocidas por la industria de software, que generalmente se consideran como *mejores prácticas,* además, por su lado la comunidad ha realizado trabajos que buscan evaluar cuáles son la prácticas aplicadas en la industria de software y su incidencia en las condiciones organizacionales [CAT04].

Una práctica proporciona una manera de abordar, de forma sistemática y verificable, un aspecto particular de un problema, con un inicio y un fin claramente definidos y con mecanismos que permiten verificar si se han logrado los objetivos. Las prácticas pueden ser desarrolladas, aprendidas y adoptadas por separado, y utilizadas en conjunción con otras para crear formas de trabajo comprensibles y coherentes [JAC07]. Desde los enfoques ágiles se prefiere el uso de *contextual practice* por encima de *best practice*. En este sentido, Scott Ambler + Associates (http://www.ambysoft.com/essays/bestPractices.html) afirman que como profesionales, los ingenieros de software se deben esforzar para comprender el contexto en el que se encuentran, para luego aplicar la práctica que mejor se adapte al mismo. En otras palabras, se deberían centrar realmente en *las prácticas contextuales* y no en las *mejores prácticas.*

Otro aspecto relacionado con las prácticas es su nivel de adopción. No es de extrañar que algunas propuestas en la literatura sean valiosas en sí mismas, pero que no han llegado a ser aplicadas en contextos reales. Existen barreras para la adopción de ciertas prácticas, y algunas innovaciones son más complejas e imponen una carga mayor de conocimientos, por lo que la importancia percibida afecta su tasa de adopción. Es aquí donde la disposición del equipo de trabajo y su apertura al cambio juegan un papel importante para la adopción adecuada de nuevas formas de trabajo.

Las condiciones de contexto en las que actualmente se desarrolla el software son altamente complejas, a la vez que tienen una influencia relevante en el éxito o fracaso de los proyectos. Continuando con la argumentación anterior, acerca de que la aplicación de las prácticas deben ser consecuentes con el contexto en el cual se van a aplicar, algunos de los factores que ayudan a contextualizarlas son el tamaño del equipo, la distribución geográfica, las normas regulatorias, la complejidad del dominio, la distribución organizacional, la complejidad técnico-organizacional y la disciplina de la empresa. Entre ellos se considera de factor crítico a la *distribución organizacional*, que está asociada a la modalidad del software, y dentro de la cual se pueden distinguir diversos tipos de desarrollo: en áreas internas de la organización, a la medida por terceros, y de soluciones genéricas estandarizadas para un dominio particular.

En los esquemas de desarrollo a la medida por terceros se crea una relación comercial, técnica y política entre el proveedor del software y el cliente, tan compleja de gestionar que, bien manejada, puede representar una alianza estratégica de beneficio mutuo que repercute en el crecimiento del sector, y que mal manejada puede representar el fracaso de la empresa proveedora o la pérdida de ventaja competitiva del cliente. Actualmente, el Global Software Development (GSD) es uno de los retos de la globalización para el desarrollo de software [HER07], porque en este esquema de trabajo los equipos se encuentran geográficamente distribuidos, y es necesario implementar mecanismos de coordinación que garanticen la integridad del producto en construcción y la integración del equipo de trabajo en sus diferentes roles.

3. LA FORMACIÓN DEL INGENIERO DE SOFTWARE

El incremento de la importancia y la complejidad del software, combinado con el conocimiento acerca de cómo *construirlo,* han generado la necesidad de actualización permanente para algunos profesionales, que han recibido una formación centrada en cómo diseñar y fabricar productos fiables pero poco especializada en el diseño, la construcción, las pruebas y el mantenimiento de productos software. Las características de este contexto no facilitan el logro de productos de buena calidad,

porque ya no es suficiente con recibir algunos de los cursos que se imparten en los programas de ingeniería tradicional o en los de Ciencias Computacionales [MIT04].

A nivel internacional existen estudios realizados por ACM, IEEE, MIT, Carnegie Mellon University, y otros organismos privados y estatales, en los que se propone que la Ingeniería de Software se debería ofrecer como un programa independiente. Sin embargo, en el contexto latinoamericano el enfoque actual de la formación en esta ingeniería no es satisfactorio. Mientras que muchos consumidores denuncian la baja calidad de los productos software, las empresas desarrolladoras sufren por la escasez de personal cualificado [THO01]. Además, los mismos empresarios no tienen claro lo que un graduado en ingeniería, relacionada con el software, pueda o no conocer acerca de desarrollarlo [SER11].

En las ofertas de empleo, muchas empresas utilizan el término *Ingeniero de Software* como un eufemismo para *programador*. La mayoría parece asumir que la única responsabilidad de estos ingenieros es escribir código; supuestos que reflejan desconocimiento acerca del significado histórico y jurídico de ser *Ingeniero*, es decir, un profesional responsable de construir productos aptos para su consumo y que, para lograrlo, debe comprender ampliamente el entorno en el que cada producto será utilizado. Consecuentemente, los ingenieros de software necesitan conocer y adquirir conocimientos que no hacen parte de la formación que se imparte en la academia, porque el software no se utiliza aisladamente de otros productos, hace parte de sistemas con componentes físicos donde computa información de los mismos. Entre los objetivos de los programas en esta área se debería tener en cuenta: 1) desarrollar la *capacidad lógico-interpretativa y abstractiva* de los estudiantes, para que comprendan y modelen los problemas antes de presentarles una solución [SER13], 2) especializarlos en el diseño de software fiable y de calidad y 3) brindarles el conocimiento suficiente acerca de otras áreas relacionadas, porque las deben conocer para solicitarle ayuda a otros profesionales, y para conformar y trabajar en equipos armónicos [MCC99]. En términos generales, los ingenieros de software se deben formar en:

1. Lo que es cierto y útil en la especialidad elegida, es decir, su cuerpo de conocimientos

2. Cómo aplicar ese conjunto de conocimientos

3. Cómo adquirir y utilizar conocimientos necesarios desde otras áreas para construir productos completos, que deben funcionar en entornos reales

4. El diseño y el análisis disciplinar, que deben seguir para cumplir con las responsabilidades que adquieren quienes construyen productos para otros.

Es decir, los ingenieros se forman en ciencia, además de en los métodos ingenieriles necesarios para aplicarla. Este proceso comienza con una aceptación de las tareas que los profesionales son capaces de realizar [SER11a]:

- Comprensión del entorno, en el que hay necesidad de cerrar una brecha o aprovechar una oportunidad de negocio, y que requiere una solución basada en software

- Identificación de los requisitos, de la solución y de las tecnologías de apoyo

- Diseño de los componentes de la solución, para determinar cuáles funciones se implementarán en el hardware y cuáles en el software, y para seleccionar los componentes básicos

- Análisis del desempeño del diseño propuesto, ya sea analíticamente o por simulación, para asegurar que el sistema cumple con los requisitos y necesidades establecidos

- Diseño de la estructura básica del software, para dividirlo en módulos y para diseñar las interfaces entre ellos y la estructura de los programas individuales, mientras se documenta con precisión todas las decisiones

- Integración del nuevo software al sistema existente

- Realización de pruebas integradas y paralelas al proceso de desarrollo
- Análisis de la estructura del software, en cuanto a completitud, consistencia e idoneidad, para la solución propuesta

- Implementación del producto, como un conjunto de programas bien estructurados y documentados

- Revisión, mejoramiento y mantenimiento de los productos software, manteniendo su integridad conceptual y preservando los documentos completos y precisos

Además, estos ingenieros son responsables por la usabilidad, seguridad y fiabilidad de sus productos, y deben ser capaces de aplicar matemática, lógica y ciencia para asegurar que el sistema que diseñan realice sus funciones cuando se entregue al cliente. Surge entonces el interrogante de si en América Latina se tienen procesos y principios estructurados para formar de esta manera a los ingenieros de software y, si existen, si se están aplicando en las instituciones de formación. En este sentido, la recomendación que le hace la comunidad, a la academia y a la industria, es que tengan en cuenta la experiencia de los profesionales que realizan trabajo práctico de desarrollo, cuando estén diseñando planes de estudio y programas de formación.

Por otro lado, existen entidades que otorgan licencias, universidades que diseñan planes de estudio y empresas centradas en mejorar la formación de su personal, e IEEE está llevando a cabo esfuerzos, como SWEBOK, para definir el cuerpo de conocimiento que estos profesionales deben adquirir. Pero la mayoría de estos grupos basan sus decisiones acerca del plan de estudios de la Ingeniería de Software en las opiniones de *expertos* en el área, en vez de estar más interesados en escuchar lo que los profesionales en la materia tienen que decir acerca de lo que es vivir y desarrollar la profesión. Interesado en aportar en este tema, Lethbridge [LET00] realizó una investigación en la que entrevistó a 186 desarrolladores profesionales, y les planteó interrogantes acerca de 75 temas relacionados con la Ingeniería de Software. Los resultados de este trabajo demuestran que es necesario entablar un diálogo entre la academia, la industria y estos profesionales, porque al parecer los objetivos formativos no logran satisfacer las necesidades de la vida real. De la revisión a los trabajos en esta temática se concluye que, debido a que las instituciones que imparten programas en Ingeniería de Software son responsables de *formar* los profesionales que van a diseñar, construir y mantener sistemas para la sociedad, deberían ampliar sus horizontes para atender de mejor forma como honrar esta responsabilidad.

El objetivo de las instituciones de formación superior debe ser el de garantizar que la formación en ingeniería sea eficaz, y de que se lleve a cabo dentro del contexto de un examen exhaustivo a todos los aspectos relevantes del sistema académico-industrial, interrelacionado los sistemas formativos con las prácticas y el sistema económico global. La formación en Ingeniería de Software se debe

reajustar para promover el logro de las características deseadas en la práctica profesional del desarrollo de software, y se debe hacer en un contexto de mayor énfasis en la base de investigación de la conducta subyacente a la práctica de la ingeniería en general. Para ello es necesario que las partes interesadas ejecuten roles protagónicos, en particular las facultades y las asociaciones profesionales de ingeniería. Además, la academia debería fomentar y ejemplificar una ética profesional, e incentivar a modificar los estados corruptos al interior de los proyectos software —sea en empresas, organismos estatales, educativos u otros contextos—, con el objetivo de remediar este flagelo, que lastima a la profesión de la Ingeniería en general y que inclusive perjudica a la misma sociedad.

3.1 Competencias del ingeniero de software

Históricamente, la forma más común para desarrollar competencias ha sido mediante procesos formativos, pero no fue sino hasta 1916 cuando más del 50% de los ingenieros profesionales lograron algún tipo de título universitario [FOR96]. Actualmente, mucha parte del desarrollo de competencias se adquiere durante la formación profesional inicial, y si bien el objetivo de los procesos formativos es impartir los conocimientos necesarios para el ejercicio de una profesión, cada curso tiende a introducir el desarrollo de habilidades a través del trabajo de laboratorios, proyectos de aula y competiciones internas [FOR96; MEN98]. Cal Poly hace hincapié en la filosofía de *aprender haciendo* [BAG98], donde los estudiantes se forman principalmente a través de laboratorios y otras actividades prácticas, en las que desarrollan competencias, generalmente en periodos diversos de tiempo, pero en todo caso el objetivo es lograrlo antes de finalizar la formación profesional inicial. Estos procesos les permiten a los estudiantes desarrollar habilidades en diversos entornos, antes de adentrarse plenamente en la práctica profesional [LET00].

Aunque se espera que las competencias se adquieran con anterioridad a la práctica profesional, la mayoría se desarrolla durante el ejercicio profesional en el lugar de trabajo [FOR96]. Los ingenieros de software recién graduados pasan algún tiempo como ingenieros en entrenamiento, antes de adquirir la experiencia o la certificación que los acredite como ingenieros profesionales. Sólo entonces comienzan su desarrollo profesional, que incluye todas las actividades destinadas a mejorar o mantener actualizados los conocimientos y las competencias de su profesión, luego de lograr el título universitario. Este proceso involucra actividades diversas, como la lectura de revistas profesionales, tomar parte de diversos programas de formación y de entrenamiento continuo, y la asistencia a eventos científicos relacionados. Dado que el desarrollo profesional de software cubre una amplia cantidad de posibilidades y alternativas, no es fácil encontrar ejemplos coherentes comunes en las profesiones relacionadas, sin embargo, se pueden describir dos patrones generales [FOR96]:

1. El desarrollo profesional es más importante en profesiones que tienen un cuerpo de conocimiento tecnológico en rápida evolución, sobre el que se basa su práctica como profesión. Por ejemplo, la Ingeniería de Software se nutre constantemente de nuevos conocimientos sobre las bases de las diferentes áreas profesionales de su desempeño, de las necesidades sociales y del desarrollo tecnológico y científico, lo que supone una demanda constante por ingenieros de software con actualización permanente en conocimientos y competencias.

2. El desarrollo profesional tiende a centrarse en las actividades con pequeñas ganancias a corto plazo para proyectos concretos, en lugar de un desarrollo a largo plazo de la profesión. En el caso de la Ingeniería de Software es más común que los profesionales tomen cursos o entrenamientos

cortos sobre una herramienta o técnica específica, que luego podrán utilizar para sus funciones laborales inmediatas, pero están poco interesados en tomar cursos largos en los avances científicos relacionados con su área.

Las competencias son habilidades, técnicas y atributos de rendimiento, que se asocian con la actualización permanente del conocimiento de los profesionales, y quizás sea el factor más importante para determinar el éxito del desarrollo de su trabajo, por lo que es necesario identificar las competencias profesionales esenciales para ellos. El proyecto Tuning Educational Structures define competencia como *una combinación dinámica de atributos, en relación a procedimientos, habilidades, actitudes y responsabilidades, que describen los encargados del aprendizaje de un programa educativo, o lo que los estudiantes son capaces de demostrar al final de un proceso formativo* [BEN07]. El mismo proyecto define dos tipos de competencias: 1) *genéricas,* que en principio son independientes del área de estudio y 2) *específicas,* para cada área temática. Normalmente, las competencias se adquieren en diferentes unidades de estudio, y por tanto pueden no estar ligadas a una sola unidad. En la Tabla 1 se detallan las competencias que debería demostrar un ingeniero de software.

Tabla 1. Competencias del ingeniero de software

Capacidad para trabajar en equipo
Capacidad para actualizar permanente su conocimiento
Habilidad para trabajar en contextos internacionales
Capacidad para resolver problemas de forma sistémica y sistemática
Habilidad para experimentar con nuevos métodos y herramientas
Habilidad para procurar el mejoramiento continuo de su trabajo y productos
Habilidad para planificar, combinar y adaptar
Capacidad de comunicación en un segundo idioma
Responsabilidad social y compromiso ciudadano
Capacidad para identificar, modelar y resolver problemas
Habilidades para buscar constantemente el mejoramiento de la calidad
Poseer sólidas habilidades analíticas
Capacidad para tomar decisiones
Habilidad para elaborar y utilizar prototipos
Capacidad para aplicar teorías, modelos y técnicas
Habilidad para planificar, combinar y adaptar
Capacidades para identificar, evaluar e implementar tecnologías
Capacidad para responder adecuadamente bajo presión
Capacidad para formular y gestionar proyectos
Habilidades para buscar, procesar y analizar información
Habilidades para la comunicación verbal y escrita
Capacidad para seleccionar especificaciones y crear modelos abstractos
Habilidades para dirigir y liderar recurso humano
Capacidad para diseñar y dirigir experimentos y para analizar e interpretar datos
Poseer disciplina, terquedad, compulsión, dedicación y voluntad de trabajo
Ser proactivo y tener iniciativa
Presentar y defender ideas creativas e innovadoras
Poseer habilidades de liderazgo, persuasión y de asesoría

3.2 Roles del ingeniero de software

Los ingenieros de software se especializan en el diseño y el desarrollo de aplicaciones software, y están formados principalmente en matemáticas y Ciencias Computacionales; y los conocimientos y

habilidades adquiridas a partir de esta formación los aplican en el diseño, construcción y despliegue de esas aplicaciones. Por otro lado, los Sistemas de Información se han convertido en un componente muy importante de las organizaciones, y las aplicaciones que estos profesionales diseñan las utiliza una amplia gama de usuarios. También necesitan formación adecuada en hardware, además de los conocimientos teóricos en software, para poder hacer frente a los complejos problemas de aplicación. De acuerdo con diversas fuentes que investigan y difunden acerca de esta cuestión, en su desempeño profesional un ingeniero de software puede llevar a cabo los roles que se describen en la Tabla 2.

Tabla 2. Roles del ingeniero de software

Rol	Definición	Tareas
Ingeniero/Analista de requisitos	Aplicar la ingeniería de requisitos con el objetivo de descubrir, analizar y documentar el propósito del producto, mediante la identificación de las necesidades de las partes interesadas	Comprender, elicitar, modelar, analizar, especificar, validar, gestionar, documentar, comunicar, integrar
Arquitecto de software	Razonar sobre si el producto cumple con sus requisitos en un contexto determinado, y sobre sus limitaciones y mejoras. Debe encontrar el balance entre el contexto de las fuerzas y las limitaciones que moldean la solución	Abstraer, visionar, conceptualizar, experimentar, crear, innovar, investigar, especificar, validar, documentar, tomar decisiones
Desarrollador	Interpretar una especificación de diseño para implementar una solución, haciendo uso de una arquitectura de referencia establecida y utilizando adecuadamente las librerías de un lenguaje o las funcionalidades provistas en un *framework*	Investigar, desarrollar prototipos, modificar, reutilizar y mantener código, re-ingeniería, probar, integrar, prototipar, modificar, reutilizar y mantener código
Analista de calidad	Aplicar los principios y prácticas de aseguramiento de la calidad del software durante todo el ciclo de vida. Sus funciones están orientadas a asegurar la calidad del proceso de desarrollo del producto	Gestión de QMS, aprobar los documentos, realizar auditorías de calidad, mantener y actualizar bases de datos de entrenamiento y auditoría, identificar problemas o deficiencias en los productos, resolver los problemas de QMS
Ingeniero de pruebas	Realizar investigación y experimentación para proporcionarles a los interesados información acerca de la calidad del producto. Proporciona una visión objetiva e independiente del software para permitirle a la empresa apreciar y comprender los riesgos del producto antes de ponerlo en producción. Asume el desafío de detectar la mayor cantidad de fallas con el mínimo esfuerzo, y participa de todas las etapas del proceso de desarrollo, colaborando para asegurar la máxima calidad del producto	Planificar, diseñar, ejecutar y administrar pruebas, conocer el negocio y su modelo, proyectar, analizar
Analista de Sistemas	Los analistas de sistemas utilizan la metodología matemática para obtener los detalles de los sistemas que analizan	Analizar sistemas y sus interacciones resultantes. Las organizaciones los requieren para que mejoren sus sistemas e incrementen su eficiencia
Diseñador	Realizar el proceso de definición de la arquitectura, los componentes, las interfaces y otras características de los productos. Describe cómo se descompone el software y cómo se organizan en componentes. Se refiere a la identificación de los principales componentes hardware y software de un producto, que proporcionan las características y atributos de calidad del mismo	Abstraer, acoplar, cohesionar, descomponer, modularizar, encapsular, diseñar, integrar, verificar

3.3 Prácticas del ingeniero de software

Las prácticas son formas de trabajo reconocidas como válidas y aceptadas por una comunidad, y se pueden entender como declaraciones de alto nivel que establecen guías generales, o como formas de trabajo que marcan ciertos *estilos*, de la misma forma que los grupos de trabajo realizan sus actividades. Desde la perspectiva de los modelos de calidad, como CMMI, las prácticas establecen guías generales —*qué*— sin entrar a definir estilos de trabajo —*cómo*—. Se podría afirmar que las prácticas de alto nivel son las más cercanas las competencias en la realización de tareas —procedimentales—, que se espera realicen los ingenieros de software, mientras que las prácticas específicas definen tendencias de trabajo y están estrechamente asociadas al contexto de aplicación en el que marcan una tendencia. En la Tabla 3 se presenta un listado de prácticas que pueden realizar los ingenieros de software en la industria, y que sirven para identificar tendencias o estilos de trabajo en el sector, respaldados y comunicados por la misma industria.

Tabla 3. Prácticas del ingeniero de software

Documentar la toma de decisiones para la solución
Aplicar las diferentes etapas de la ingeniería de requisitos
Diseñar soluciones con base en los requisitos
Codificar soluciones siguiendo los estándares y especificaciones de la arquitectura
Verificar y Validar el cumplimiento de los requisitos en la alternativa de solución
Utilizar prototipos para validar que la solución propuesta cumple los requisitos
Utilizar esquemas gráficos para comprender y explicar requisitos
Documentar las soluciones
Utilizar herramientas unificadas para todos los procesos
Mantener un repositorio unificado para conservar y garantizar la integridad de la solución
Comprender, moldear y representar los problemas
Generar código a partir de modelos
Proponer diferentes alternativas de solución
Representar los componentes estructurales y dinámicos de la solución
Utilizar patrones de diseño
Interactuar con el equipo a través de herramientas para trabajo colaborativo
Diseñar y aplicar planes de pruebas
Diseñar, aplicar y validar casos de prueba
Verificar y Validar software
Gestionar cambios a través de interacción directa
Gestionar riesgos
Reutilizar componentes
Gestionar y administrar la calidad
Realizar capacitación y entrenamiento permanente
Diseñar, aplicar y divulgar mejoras al proceso
Gestionar los cambios
Realizar instructivos que especifiquen y guíen el despliegue de la solución
Hacer control de seguimiento
Asignar roles como líder de proyectos
Diseñar e implementar mecanismos eficientes de comunicación

REFERENCIAS

[AGI01] Agile Alliance. 2001. The agile alliance. Online [Mayo 2013].

[AMB09] Ambler, S. 2009. The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments. Online [Junio 2013].

[AMB10] Ambler, S. 2010. Positions Paper. Proceedings of the Semat Zurich Workshop. Online [Mayo 2013].

[ANA06] Anaya, R. 2006. Una visión de la enseñanza de la Ingeniería de Software como apoyo al mejoramiento de las empresas de software. Revista Universidad EAFIT, 42(141), 60-76.

[BAG98] Bagert, D. 1998. The Challenge of Curriculum Modeling for an Emerging Discipline: Software Engineering. Proceedings 28th Annual Frontiers in Education Conference FIE '98, 910-915.

[BEN07] Beneitone, P. et al., 2007. Reflexiones y perspectivas de la educación superior en América Latina. Proyecto Tuning. Spain/impreso.

[BOH11] Bohem, B. 2011. Some Future Software Engineering Opportunities and Challenges. Online [Mayo 2013].

[BRO86] Brooks, F. 1986. No Silver Bullet - Essence and Accident in Software Engineering. Proceedings of the IFIP Tenth World Computing Conference, 1069-1076.

[BRU02] Bruegge, B., Dutoit, A. 2002. Ingeniería de software orientado a objetos. Prentice Hall.

[CAT04] Cater-Steel B. & Bus, A. 2004. An Evaluation of Software Development Practice and Assessment-Based Process Improvement in Small Software Development Firms. PhD Thesis. School of Computing and Information Technology Faculty of Engineering and Information Technology, Griffith University.

[COO99] Cook, D.A. & Dupaix, L. 1999. A Gentle Introduction to Software Engineering. Software Technology Support Center.

[FOR96] Ford, G. & Norman, G. 1996. A Mature Profession of Software Engineering. Technical Report. CMU/SEI-96-TR-004, ESC-TR-96-004. Carnegie Mellon University.

[FUG00] Fugetta, A. 2000. Software Process: A Roadmap. In Finkelstein, A. (Ed.), The Future of Software Engineering. ACM Press.

[GLA06] Glass, R.L. 2006. Facts and Fallacies of Software Engineering. Addison-Wesley.

[HER07] Herbsleb, J.D. 2007. Global Software Engineering: The Future of Socio-technical Coordination. Proceedings Future of Software Engineering FOSE'07, 188-198.

[IEE04] IEEE & ACM. 2004. Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. National Science Foundation.

[IEE90] IEEE. 1990. Standard Computer Dictionary.

[INT09] Intelligent Tings. 2009. The New Software Engineering Manifesto. http://intelligentthings.com/content/new-software-engineering-manifesto, [Abril 2013].

[JAC07] Jacobson, I.; Wei, P. & Spence, I. 2007. Enough of Processes - Lets do Practices. Journal of Object Technology, 6(6), 41-66.

[JOR06] Jorgenson, D.W. & Wessner, C.W. 2006. Measuring and Sustaining the New Economy, Software, Growth, and the Future of the U.S. Economy. National Academies Press.

[LET00] Lethbridge, T.C. 2000. What knowledge is important to a software professional? Computer, 33(5), 44-50.

[MAI97] Maibaum, T.S.E. 1997. What We Teach Software Engineers in the University: Do We Take Engineering Seriously? ACM SIGSOFT Software Engineering Notes, 22(6), 40-50.

[MCC03] McConnell, S. 2003. Professional Software Development: Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers. Addison Wesley.

[MCC99] McConnell, S. & Tripp, L. 1999. Professional Software Engineering: Fact or Fiction? IEEE Software, 16(6), 13-18.

[MEN98] Mengel, S.A. 1998. Guidelines Proposal for Undergraduate Software Engineering Education. Proceedings 28th Annual Frontiers in Education Conference FIE '98, 916-919.

[MIR06] Mirbel, I. & Ralyté, J. 2006. Situational Method Engineering: Combining Assembly-based and Roadmap-driven Approaches. Requirements Engineering 11(1), 58-78 (2006).

[MIT04] Mitchell, W. 2004. Is software engineering for everyone? Proceedings of second annual conference on Mid-south College computing, 53-64.

[NAV10] Navarro, J.M. & Garzás, J. 2010. Experiencia en la implantación de CMMI-DEV v1.2 en una micro-pyme con metodologías ágiles y software libre. REICIS Revista Española de Innovación, Calidad, 6(1), 6-15.
[NIS10] National Institute of Standards and Technology (NIST). 2010. Software Testing Metrics Portal. Online [Junio 2013].
[PRE05] Pressman, R. 2005. Software Engineering: A Practitioner's Approach. McGraw-Hill Science.
[RUI07] Ruiz, F. 2007. Software process engineering. De una gestión de procesos Contemplativa a una Productiva. Online [Junio 2013].
[SEI90] SEI. 1990. SEI Report on Undergraduate Software Engineering Education. Online [Junio 2013].
[SER11] Serna, M.E. 2011. Systems engineering for the XXI century: A proposal from the academy. Proceedings Ninth LACCEI Latin American and Caribbean Conference (LACCEI'2011), Paper 5. August 3-5, Medellin, Colombia.
[SER11a] Serna, M.E. 2011. 'Software Engineering´ is Engineering. Rev. Anti. de las Ciencias Computacionales y la Ing. de Soft. RACCIS, 1(1), 34-43.
[SER13] Serna, M.E. & Zapata, A.L. 2013. Necesidad de la Lógica y la Abstracción en la Formación de Ingenieros. En XX International Conference on Learning. University of the Aegean, Rhodes, Greece. 11-13 july 2013.
[SOM06] Sommerville, I. 2006. Software Engineering. Addison-Wesley.
[THO01] Thompson, J.B. 2001. A Long and Winding Road (Progress on the Road to a Software Engineering Profession). Proceedings of 25th International Computer Software and Applications Conference on Invigorating Software Development, 39-45.
[VAU00] Vaughn, R. 2000. Software Engineering Degree Programs. CROSSTALK, 13(3), 7-9.

CAPÍTULO VII

OFERTA DE PROGRAMAS EN DESARROLLO O INGENIERÍA DE SOFTWARE

Carlos A. CASTRO C. & Jaime A. MONSALVE M.
Universidad de San Buenaventura
Medellín, Antioquia, Colombia

PRESENTACIÓN

En este capítulo se relaciona el listado de programas, que ofrecen las instituciones Latinoamericanas, cuyo nombre permite relacionarlos directamente con el desarrollo profesional de software. El proceso consistió de una amplia búsqueda en cada uno de los organismos que centralizan esta información en los países de la región, con el objetivo de determinar, mediante el nombre del programa, su relación con la Ingeniería de Software. Algunos países no aparecen debido a que el resultado de la búsqueda no arrojó nombres de programas que cumplieran la condición establecida. Puede ser que como área de profundización y/o perfil de egreso las instituciones establezcan que su orientación es hacia a la Ingeniería de Software o al desarrollo profesional de software, pero la idea es poder orientar a los estudiantes que se interesan por ingresar a programas que les indiquen directamente que su profesión será la de Técnico, Tecnólogo, Ingeniero, Especialista, Magister o Doctor en Software, y no en otras áreas relacionadas, como Ingeniería de Sistemas, Ingeniería Informática, Sistemas de Información, o cualquier otro título que puedan obtener.

COLOMBIA

Institución	Ciudad	Programa	Nivel
Tecnológico de Antioquia	Medellín	Ingeniería en Software	Pregrado
Universidad de Medellín	Medellín	Tecnologías en Desarrollo de Software Maestría en Ingeniería de Software Especialización en Ingeniería de Software	Pregrado Posgrado Posgrado
Universidad EAFIT	Medellín	Especialización en Desarrollo de Software	Posgrado
Corporación Universitaria Empresarial de Salamanca	B/quilla	Tecnología en Desarrollo de Software	Pregrado
Corporación Universidad de la Costa	B/quilla	Tecnología en Desarrollo de Software y Redes Telemáticas	Pregrado
Universidad del Norte	B/quilla	Especialización en Ingeniería del Software	Posgrado
Fundación CEDINPRO	Bogotá	Técnica Profesional en Desarrollo de Software y Redes	Pregrado
CUN	Bogotá	Tecnología en Desarrollo de Software y Redes	Pregrado
Politécnico Internacional Institución de Educación Superior	Bogotá	Técnica Profesional en Desarrollo de Software	Pregrado
Fundación Para la Educación San Mateo	Bogotá	Tecnología en Desarrollo de Software Técnica Profesional en Programación de Software	Pregrado
Fundación de Educación Superior San José	Bogotá	Técnica Profesional en Desarrollo de Software y Redes	Pregrado
Corporación de Educación Tecnológica Colsubsidio	Bogotá	Tecnología en Desarrollo de Software	Pregrado
CTB	Bogotá	Tecnología en Desarrollo de Software	Pregrado
Politécnico Gran Colombiano	Bogotá	Tecnología en Desarrollo de Software	Pregrado
Fundación Universitaria del Área Andina	Bogotá	Técnica Profesional en Desarrollo de Software para Dispositivos móviles	Pregrado
Fundación Universitaria Panamericana	Bogotá	Tecnología en Diseño de Software Ingeniería de Software Técnica Profesional en Desarrollo de Software Técnico Profesional en Programación de Software Tecnología en Desarrollo de Software	Pregrado
Corporación Universitaria UNITEC	Bogotá	Tecnología en Desarrollo de Software	Pregrado
Fundación Universitaria-INPAHU	Bogotá	Ingeniería de Software	Pregrado
UNIAGUSTINIANA	Bogotá	Tecnología en Desarrollo de Software.	Pregrado
Universidad Manuela Beltrán	Bogotá	Ingeniería de Software	Pregrado
Universidad San Buenaventura	Bogotá	Tecnología en Programación y Desarrollo de Software	Pregrado
Universidad Antonio Nariño	Bogotá	Especialización en Ingeniería de Software Ingeniería de Sistemas con Énfasis en Software	Posgrado
Universidad de los Andes	Bogotá	Maestría en Ingeniería de Software Especialización en Construcción de Software	Posgrado
Universidad INCCA de Colombia	Bogotá	Especialización en Ingeniería de Software	Posgrado
Corporación Universitaria de Sucre	Cartagena	Tecnología en Desarrollo de Software	Pregrado
Corporación Universitaria Rafael Núñez	Cartagena	Tecnología en Sistemas de Información y de Software	Pregrado
Universidad Tecnológica de Bolívar	Cartagena	Tecnología en Desarrollo de Software	Pregrado

UNISINU	Cartagena	Técnica Profesional en Desarrollo de Software y Redes de Cómputo	Pregrado
Universidad Autónoma de Manizales	Manizales	Especialización en Ingeniería de Software Maestría en Gestión y Desarrollo de Proyectos de Software	Posgrado
Universidad Cooperativa de Colombia	Manizales	Tecnología en Desarrollo de Software	Pregrado
Colegio Mayor del Cauca	Popayán	Tecnología en Desarrollo del Software	Pregrado
Corporación Universitaria Autónoma del Cauca	Popayán	Técnica Profesional en Desarrollo de Software	Pregrado
Institución Universitaria ComfaCauca	Popayán Santander de Quilichao	Técnica Profesional en Desarrollo de Software Técnico Profesional en Programación de Software	Pregrado
Universidad Cooperativa de Colombia	Popayán	Especialización en Desarrollo de Software	Posgrado
Fundación Universidad del Norte	Valledupar	Especialización en Ingeniería del Software	Posgrado
Universidad de Cundinamarca	Soacha	Técnico Profesional en Desarrollo de Software	Pregrado
Universidad Sur Colombiana	Neiva La Plata Garzón Pitalito	Tecnologías en Desarrollo de Software	Pregrado
Universidad del Magdalena	Santa Marta	Especialización en Desarrollo de Software	Posgrado
Universidad De Los Llanos	Villavicencio	Especialización en Ingeniería del Software	Posgrado
Universidad de Santander	Cúcuta	Especialización en Ingeniería de Software	Posgrado
Universidad Francisco De Paula Santander	Cúcuta	Especialización en Desarrollo de Software	Posgrado
Escuela de Administración y Mercadotecnia del Quindío	Armenia	Tecnología en Desarrollo de Software Ingeniería de Software Técnica Profesional en Desarrollo de Software	Pregrado
Universidad Católica de Pereira	Pereira	Tecnología en Desarrollo de Software	Pregrado
UDI	Barrancabermeja Bucaramanga	Técnicas Profesionales en Desarrollo de Software	Pregrado
UNAB	Bucaramanga	Maestría en Gestión, Aplicación Y Desarrollo de Software	Posgrado
CUN	Sincelejo	Tecnología en Creación de Soluciones de Software y Redes de Datos	Pregrado
CORPOSUCRE	Sincelejo	Tecnología en Desarrollo de Software	Pregrado

ARGENTINA

Institución	Programa	Ciudad	Nivel
Universidad Argentina de la Empresa	Técnico Universitario en Desarrollo de Software	Capital Federal	Otros pregrados
Universidad Nacional del Sur	Ingeniero en Sistemas de Software	Bahía Blanca	Grado
Universidad del Aconcagua	Licenciado en Informática y Desarrollo de Software	Mendoza	Grado
Universidad Empresarial Siglo 21	Ingeniero en Software	Córdoba	Grado
Pontificia Universidad Católica Argentina	Especialista en Ingeniería de Software	Capital Federal	Especialidad
Universidad CAECE	Magister en Ingeniería de Software	Capital Federal	Maestría
Universidad Nacional de La Plata	Especialista en Ingeniería de Software Magister en Ingeniería de Software	La Plata	Especialidad Maestría

Universidad Nacional de Jujuy	Magister en Ingeniería de Software	Salvador De Jujuy	Maestría
Universidad Nacional de La Rioja	Magister en Ingeniería de Software	La Rioja	Maestría
Universidad Nacional de San Luis	Especialista en Ingeniería de Software Magister en Ingeniería de Software	San Luis	Especialidad Maestría

CHILE

Institución	Título	Tipo	Región
CFT CENCO	Técnico De Nivel Superior En Programación Y Diseño Informático	Técnico	Valparaíso
CFT CRECIC	Técnico Analista Programador Computacional	Técnico	Bio-Bio
CFT DUOC UC	Analista Programador Computacional	Analista	Metropolitana
CFT INACAP	Analista Programador	Analista	Arica Tarapacá Iquique Antofagasta Atacama Coquimbo Valparaíso Metropolitana Libertador B. O'Higgins Maule Araucanía Los Ríos Aysén Del G.C. Ibáñez Los Lagos
CFT LAPLACE	Programación En Computación E Informática	Programador	Metropolitana
CFT Magnos	Informática Mención Desarrollo De Aplicaciones	Técnico	Metropolitana
CFT Magnos	Informática Mención Desarrollo De Aplicaciones Y Diseño Web	Técnico	Metropolitana
CFT Massachusetts	Programación Computacional Mención Análisis De Sistemas	Programador	Maule
CFT PRODATA	Técnico En Informática con Especialización En Desarrollo De Software Y Multimedia	Técnico	Valparaíso
CFT Prodata	Técnico En Informática Con Especialización En Desarrollo De Software Y Multimedios	Técnico	Los Lagos
CFT San Agustín De Talca	Analista Programador	Analista	Maule
CFT Simón Bolívar	Programación Y Diseño Informático	Programador	Metropolitana
IP AIEP	Programación Computacional	Programador	Antofagasta Valparaíso Libertador Bio-Bio
IP AIEP	Programación Y Análisis De Sistemas	Programador	Antofagasta Coquimbo Valparaíso Metropolitana Libertador Maule Bio-Bio Araucanía Los Lagos

IP AIEP	Ingeniería De Ejecución En Informática Mención Desarrollo De Sistemas	Ingeniero	Metropolitana Libertador Maule Bio-Bio
IP CIISA	Técnico En Programación Computacional	Técnico	Metropolitana
IP De Chile	Analista Programador Computacional	Analista	Coquimbo La Serena
IP De Chile	Analista Programador Computacional	Analista	Metropolitana Libertador
IP DR. Virginio Gómez G.	Técnico Analista Programador	Técnico	Bio-Bio
IP DUOC UC	Analista Programador Computacional	Analista	Valparaíso Viña Del Mar Metropolitana Bio-Bio
IP Santo Tomas	Analista Programador	Analista	Coquimbo Ovalle Metropolitana Araucanía
Universidad De Atacama	Tecnología Universitaria En Análisis Y Programación De Sistemas Computacionales	Tecnólogo	Atacama Vallenar
Universidad De Playa Ancha De Ciencias De La Educación	Programador En Aplicaciones Computacionales	Programador	Valparaíso
Universidad Técnica Federico Santa María	Técnico Universitario En Programación De Computadores	Técnico	Valparaíso
Universidad Técnica Federico Santa María	Ingeniería Ejecución En Software	Ingeniero	Metropolitana Libertador Bio-Bio

MÉXICO

Institución	Título	Ciudad	Nivel
CETYS Universidad	Ingeniería De Software	Ensenada	Pregrado
Universidad Autónoma De Chihuahua	Ingeniero De Software	Chihuahua	Pregrado
Universidad Autónoma De Chihuahua	Ingeniero De Software (Virtual)	Ciudad Juárez	Pregrado
Universidad Autónoma De Guadalajara	Ingeniero En Software	Guadalajara	Pregrado
Universidad Autónoma De Guadalajara	Especialidad En Ingeniería De Software	Guadalajara	Posgrado
Universidad Autónoma De Querétaro	Ingeniería En Software	Santiago De Querétaro	Pregrado
Universidad Autónoma De Sinaloa	Licenciatura En Ingeniería De Software	Mochis	Pregrado
Universidad Autónoma De Tlaxcala	Maestría En Ingeniería De Software	Xicohtencatl	Posgrado
Universidad Autónoma De Yucatán	Licenciatura En Ingeniería De Software	Mérida	Pregrado
Universidad Del Norte	Diplomado En Ingeniería De Software	Monterrey	Posgrado

Universidad Del Valle De Atemajac	Ingeniería Con Especialidad En Desarrollo De Software	Guadalajara	Posgrado
Universidad Del Valle De Atemajac	Ingeniería Con Maestría En Ingeniería De Software	León	Posgrado
Universidad Del Valle De Atemajac	Ingeniería Con Especialidad En Desarrollo De Software	León	Posgrado
Universidad Madero	Ingeniería De Software	San Andrés Cholula	Pregrado
Universidad Nacional Autónoma De México	Técnica Profesional En Desarrollo De Aplicaciones De Software	Coyoacán	Pregrado
Universidad Popular Autónoma Del Estado De Puebla	Ingeniería De Software	Puebla	Pregrado
Universidad Popular Autónoma Del Estado De Puebla	Ingeniería De Software	Puebla	Posgrado
Universidad Popular Autónoma Del Estado De Puebla	Ingeniería De Software	Puebla	Posgrado
Universidad Tecnológica De La Mixteca	Maestría En Ingeniería De Software	Huajuapan De León	Posgrado

URUGUAY

Institución	Título	Ciudad	Nivel
Universidad ORT Uruguay	Analista Programador	Montevideo	Técnica/Pregrado

PARAGUAY

Institución	Título	Ciudad	Titulo	Nivel
Universidad Americana	Ingeniería En Informática Con Énfasis En Ingeniería Del Software	Asunción	Ingeniero	Pregrado
Universidad Autónoma De Asunción	Programación De Computadoras	Asunción	Programador	Técnico/ Pregrado

BOLIVIA

Institución	Título	Ciudad	Titulo	Nivel
Universidad Técnica De Oruro	Ingeniería Informática Mención Desarrollo De Software	Oruro	Ingeniero	Pregrado

ECUADOR

Institución	Título	Ciudad	Titulo	Nivel
Universidad De Cuenca	Maestría En Software	Cuenca	Ingeniero	Posgrado

VENEZUELA

Institución	Título	Ciudad	Nivel
Universidad Centro Occidental Lisandro Alvarado	Maestría En Ciencias De La Computación Ingeniería De Software	Barquisimeto	Maestría/ Posgrado

EL SALVADOR

Institución	Título	Departamento	Nivel
Universidad Capitán General Gerardo Barrios	Técnico en Programación de Computadoras	San Miguel	Técnico
Universidad Tecnológica de el Salvador	Técnico en Ingeniería de Software	San Salvador	Técnico

BRASIL

Institución	Nivel	Título	Ciudad
Fundacao Universidade Federal Do Pampa-Unipampa-Unipampa	Pregrado	Engenharia De Software	Alegrete
Centro Universitario Univates-Univates	Pregrado	Engenharia De Software	Lajeado
Universidade Da Regiao De Joinville-Univalle	Pregrado	Engenharia De Software	Joinville
Universidade Do Oeste De Santa Catarina-UNOESC	Pregrado	Engenharia De Software	Videira
Centro Universitario De Maringá-UNICESUMAR-UNICESUMAR	Pregrado	Engenharia De Software	Maringá
Pontificia Universidade Católica De Minas Gerais-PUC MINAS	Pregrado	Engenharia De Software	Belo Horizonte
Universidade Federal De Goiás-UFG	Pregrado	Engenharia De Software	Goiania
Universidade De Rio Verde-FESURV	Pregrado	Engenharia De Software.	Rio Verde
Universidade Federal Do Amazonas-UFAM	Pregrado	Engenharia De Software.	Itacoatiara
Universidade Federal Do Rio Grande Do Norte-UFRN	Pregrado	Engenharia De Software.	Natal
Universidade Federal Do Ceará-UFC	Pregrado	Engenharia De Software.	Quixadá

********** | **********

SI TRABAJAMOS JUNTOS, LO LOGRAREMOS

La iniciativa “Libro Blanco” tuvo su origen en el año 2010, en el marco del Congreso Ingeniería 2010 Argentina, que se llevó a cabo en la ciudad de Buenos Aires. En una de las reuniones de trabajo se planteó una discusión acerca de los diferentes enfoques con los que se ofrecen los programas en Ingeniería de Sistemas en el mundo, y de las similitudes que algunos de ellos tienen con los de Ingeniería de Software. En el auditorio se interrogó a los participantes por sus puntos de vista acerca de este tema, y cada uno presentó la visión que desde su conocimiento reflejaba la situación en su país. Entre los 22 participantes, provenientes de Estados Unidos, Colombia, Argentina, Chile, Brasil, España, Inglaterra, Nigeria, Suráfrica, Francia y Arabia, quedó el sin sabor de no encontrar unanimidad de criterios para llegar a conclusiones finales acerca de la situación de estas ingenierías. En definitiva, y para concluir el taller, se tomó la decisión de crear capítulos de trabajo por regiones representativas con el objetivo de investigar acerca de esta cuestión. Se organizaron las comisiones para África, Europa, Norte América, Asía y Latinoamérica.

Red Latinoamericana en Ingeniería de Software (RedLatinaIS)
Instituto Antioqueño de Investigación (IAI)
©2013