Educación en Ingeniería de Software

Lucía Camilloni
Universidad de la República
Montevideo, Uruguay
lcamilloni@fing.edu.uy

Diego Vallespir
Universidad de la República
Montevideo, Uruguay
dvallesp@fing.edu.uy

La creciente importancia que ha tomado el software en la vida diaria hace que exista una fuerte y creciente demanda mundial de ingenieros de software calificados que ayuden a producir software de calidad en plazo y dentro del presupuesto [1].

La ingeniería de software es una disciplina relativamente nueva, pero que ha madurado muy rápidamente. Este capítulo presenta los distintos esfuerzos que se han realizado para desarrollar y mejorar la educación en ingeniería de software.

1. Introducción

Hoy en día el software es un componente central en casi todos los aspectos de la vida diaria tales como el gobierno, los bancos, la educación, el transporte y la agricultura. El número, tamaño y los dominios de aplicación de los sistemas que contienen software ha crecido dramáticamente y éstos han ayudado a aumentar la eficiencia y la productividad [2].

Así como las funcionalidades de los productos y de los sistemas crecen, también se necesita implementar eficientemente y correctamente el complejo software que posibilita este crecimiento. En la actualidad existen serios problemas en lo que refieren al costo, tiempo y calidad de muchos productos de software [2]. Debido a esto, existe una creciente demanda de ingenieros de software calificados que puedan aplicar prácticas profesionales para lograr producir software de calidad, cumpliendo con las necesidades de los usuarios, en tiempo y dentro del presupuesto establecido [3].

La Ingeniería de Software (IS) es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, mantenimiento y operación del software [4]. Los principios y prácticas de esta disciplina son esenciales para el desarrollo de sistemas grandes, complejos y confiables [3].

El estudio y la práctica de la IS se ha visto influenciada tanto por sus raíces provenientes de las ciencias de la computación así como por su surgimiento como una disciplina ingenieril. La naturaleza intangible del software hace que la IS sea diferente a otras disciplinas ingenieriles. La IS busca integrar los principios de la matemática y las ciencias de la computación con las prácticas ingenieriles desarrolladas para artefactos tangibles [2].

Al día de hoy, aún existen considerables malentendidos y confusiones con respecto a las diferencias y similitudes entre la IS y la ciencia de la computación. Parnas clarifica esto subrayando que los estudiantes de ingeniería deben aprender cómo usar las ciencias de la computación y las matemáticas para construir sistemas con una alta componente de software, mientras que los estudiantes de ciencias de la computación deben aprender cómo crear nuevo conocimiento a partir de información ya conocida [5]. Por lo general los programas en IS y

ciencias de la computación tienen muchos cursos en común pero el enfoque es distinto; por ejemplo los estudiantes de IS se enfocan más en aspectos relacionados a la fiabilidad, calidad, costos y el mantenimiento del software [3]. Si bien esto es discutible, algunos de estos aspectos pueden no ser considerados desde la perspectiva de la ciencia de la computación.

El desarrollo de software es comúnmente llevado a cabo en equipos y requiere de una buena comunicación con los distintos *stakeholders* involucrados. Debido a esto, las actividades de carácter social representan una parte importante del trabajo diario de los ingenieros de software. Por esto la IS está fuertemente relacionada con otras disciplinas tales como la sociología y la psicología, ya que existe la necesidad de contar con técnicas efectivas para compartir ideas y lograr el entendimiento de los requerimientos [6]. A su vez resulta muy importante la comprensión del aspecto cooperativo y humano en el desarrollo y el mantenimiento de software [7].

La educación es un pilar fundamental para cualquier profesión. Es difícil concebir una profesión madura sin contar con una educación madura en el área debido a que sin una educación de este tipo, es imposible concebir profesionales altamente capacitados. El término IS se acuña en 1968, la primera maestría en IS es de 1979 y la primera carrera de pregrado es de 1987. Es decir, la educación en el área y el área en sí misma son extremadamente nuevas. Sin embargo, como se muestra en este capítulo, los últimos 20 años presentan grandes cambios en lo referente a la educación en IS.

En este capítulo se presentan distintos aspectos de la educación en IS y el estado actual de la misma. Es a través de una retroalimentación entre las necesidades de la industria y las organizaciones, las propuestas educativas de la academia y las propuestas y necesidades de las sociedades profesionales que se construye y avanza la educación en una disciplina. Conocer cuál es el estado actual de la educación en IS es imprescindible para pensar sobre el estado actual de esta profesión.

Una disciplina madura debe contar con un cuerpo de conocimiento. Sin este es difícil consensuar la educación, otorgar licencias, crear certificaciones y acreditar planes de estudio para que puedan garantizar la formación de profesionales competentes. En la sección 2 se presenta el SWEBOK, la guía para el cuerpo de conocimiento de la ingeniería de software.

En las secciones 3 y 4 se presentan las guías curriculares propuestas por la IEEE-CS y la ACM para educación a nivel de pregrado y de posgrado. Estas guías, que se basan en el SWEBOK, presentan los conocimientos y habilidades consensuados que debe tener un profesional en IS (ingeniero de software) y un Magister en IS. La guía es una ayuda para elaborar programas de estudio.

En la sección 5 se presenta la educación en ingeniería de software en otros contextos. Otros contextos considerados son, por ejemplo, la educación en ingeniería de software, o en ciencias de la computación, introducida de forma temprana a nivel secundario y la educación en ingeniería de software por fuera de los programas de nivel secundario o terciario.

En la sección 6 se presentan los elementos que componen a una profesión y el estado actual de la profesión de ingeniería de software. Discutimos principalmente aspectos que se encuentran fuertemente vinculados con la educación, como la acreditación de programas educativos, las certificaciones y licencias, entre otros. Pero, también presentamos brevemente otros aspectos

que son vitales para una profesión, como el código de ética y las sociedades profesionales. Por último en la sección 7 presentan las conclusiones.

2. Guía para el Cuerpo de Conocimiento: el SWEBOK

La guía para el cuerpo de conocimiento de la ingeniería de software (SWEBOK[1]) es un proyecto de la IEEE-CS que describe el conocimiento generalmente aceptado de la ingeniería de software. Actualmente la guía se encuentra en la versión SWEBOK 2004.

El SWEBOK tiene como propósito brindar una caracterización consensuada de los límites de la disciplina de IS y proveer un acceso por tópicos al cuerpo de conocimiento que la soporta [8].

El SWEBOK no debe confundirse con el cuerpo de conocimiento en sí. No es posible poner el cuerpo de conocimiento completo en un único documento, por lo que surge la necesidad de contar con una guía.

El SWEBOK describe y organiza las porciones del cuerpo de conocimiento que son generalmente aceptadas, provee acceso por tópicos y referencias bibliográficas. La definición del término conocimiento 'generalmente aceptado' proviene del *Project Management Institute* (PMI)[2]. Éste establece que el conocimiento generalmente aceptado es aquel que se aplica a la mayoría de los proyectos, la mayor parte del tiempo, y un amplio consenso valida el valor y la eficacia. En particular este término se utiliza para distinguir del conocimiento avanzado y de investigación (prácticas innovadoras usadas por pocas organizaciones o que se encuentran en la etapa de investigación) y del conocimiento especializado (prácticas usadas sólo para el desarrollo de determinados tipos de software).

Lograr el consenso por parte todos los sectores importantes de la comunidad pertinente en un cuerpo de conocimiento representa un hito clave en cualquier disciplina. La IEEE - CS considera que este hito es crucial para el desarrollo de la ingeniería de software como una profesión, ya que permite establecer qué es lo que debe saber un profesional de esta disciplina. Sin un consenso, por ejemplo, no se podría validar los exámenes para licenciamiento ni se podrían establecer criterios para acreditar planes de estudios [8].

El SWEBOK tiene los siguientes objetivos:

i. Promover una visión consistente de la IS en todo el mundo.
ii. Aclarar el lugar y establecer los límites de la IS con respecto a otras disciplinas tales como ciencias de la computación, gestión de proyectos, ingeniería en computación y matemáticas.
iii. Caracterizar los contenidos de la disciplina de IS.
iv. Proporcionar acceso por tópicos al cuerpo de conocimiento de IS.
v. Sentar las bases para el desarrollo de currículos, certificaciones individuales y material para el licenciamiento.

2.1 Historia del SWEBOK

A continuación se describe las distintas fases por las que pasó el proyecto SWEBOK desde su concepción hasta la versión actual del mismo.

---

1 SWEBOK por su sigla en inglés correspondiente a ***S**oftware **E**ngineering **Bo**dy of **K**nowledge*

2http://www.pmi.org/

La IEEE-CS contrató al *Software Enginering Management Research Laboratory* de la Université du Québec à Montréal (UQAM) para gestionar el proyecto SWEBOK con el objetivo de contar con personas dedicadas a tiempo completo. Este proyecto buscó desde sus inicios contar con una amplia participación y con el apoyo de una variedad de comunidades.

El equipo del proyecto se basó en tres importantes principios para guiar el proyecto: la transparencia (documentación y publicación del proceso de desarrollo de la guía), el consenso y la amplia distribución. Para lograr esta última, se determinó que la guía deberá permanecer de acceso libre, en al menos un formato[3].

El plan de proyecto del SWEBOK constó de tres fases sucesivas llamadas *Strawman, Stoneman* y *Ironman*.

La fase *Strawman* tuvo como principal objetivo definir la estrategia, demostrar cómo el proyecto podía ser organizado y entregar un prototipo. Este prototipo, conocido como la versión *Strawman* del SWEBOK, fue publicado en Setiembre de 1998. Consiste en un reporte inicial que presenta un borrador de la lista de áreas de conocimiento (KA[4]) y de las disciplinas que interactúan con la IS. Como su nombre lo indica, esta versión buscó estimular el debate y el cuestionamiento [9].

Desde 1993 hasta Junio del 2000 la *Association for Computer Machinery* (ACM) trabajó en conjunto con la IEEE-CS en proyectos para guiar la evolución de la ingeniería de software como una profesión.

En el otoño de 1998 se comenzó con la fase *Stoneman.* A su vez, este mismo año se forma el *Software Engineering Coordinating Commitee* (SWECC), establecido en forma conjunta por la IEEE-CS y la ACM para actuar como una entidad permanente que busca fomentar la evolución de la IS como una profesión. El SWECC lanzó proyectos tales como: el SWEBOK, el desarrollo de recomendaciones para currículas en IS y la definición del código de ética [10].

En el momento que el SWECC se estaba estableciendo recibió una solicitud de la Junta para el Licenciamiento de Ingenieros Profesionales de Texas pidiendo ayuda para definir el criterio de desempeño para los exámenes de licenciamiento para Ingenieros de Software. A partir de este pedido, el tema del licenciamiento de Ingenieros de Software se convirtió en un motivo de discusión a nivel del SWECC.

En Junio del 2000, la ACM se retiró del SWECC como parte de la decisión de tomar una posición en contra del licenciamiento de Ingenieros de Software. El consejo de la ACM sintió que las actividades del SWECC se habían vuelto demasiado relacionadas a la promoción del licenciamiento de ingenieros de software [10]. De esta forma el proyecto SWEBOK quedó en las manos de la IEEE-CS.

El desarrollo de la versión *Stoneman* pasó por tres ciclos de revisiones públicas en los cuales intervinieron cerca de 500 revisores de 42 países y se registraron cerca de 9000 comentarios. Los principales entregables de esta versión fueron [11]:

- Una lista de KA.
- Una lista de tópicos y referencias bibliográficas relevantes para cada KA.

3 http://www.computer.org/portal/web/swebok/objectives

4 KA por su sigla en inglés correspondiente a *Knowledge Area*

- Una lista de disciplinas relacionadas con la IS. Las KA y tópicos en común entre la IS y las disciplinas relacionadas.

La publicación de versión *Trial* de la guía en el 2001 marcó el fin de la fase *Stoneman* y el proyecto comenzó con una etapa de uso de ensayo.

La fase *Ironman* duró hasta el año 2003 y consistió en 2 sub-fases. En la primer sub-fase se realizó la experimentación y promoción de la versión *Trial* de la guía y se desarrollaron las normas para el rendimiento para los profesionales en ingeniería de software. Durante este período la versión *Trial* de la guía fue producto de extensivas revisiones y comentarios; participando más de 120 revisores de más de 21 países. La segunda sub-fase consistió en el desarrollo de la versión *Ironman* de la guía basada en la retroalimentación obtenida en la primer sub-fase.

En el año 2004 la IEEE-CS aprobó formalmente y publicó la guía actual, el SWEBOK 2004, marcando de esta forma el fin de la fase *Ironman* del proyecto. Esta guía ha logrado el consenso a través de una amplia revisión y uso de evaluación.

El proyecto SWEBOK cuenta con varios *stakeholders* entre los que se encuentran: la *Industrial Advisory Board* (integrada por representantes de la industria), agencias de investigación, el Consejo Canadiense de Ingenieros Profesionales y la IEEE-CS.

2.2 SWEBOK 2004

El SWEBOK 2004 es la versión actual de la guía. Esta guía caracteriza los contenidos de la disciplina de ingeniería de software, es decir los conocimientos necesarios para la práctica de la ingeniería de software [8]. Estos conocimientos están previstos para ser alcanzados, por ejemplo, a través de una educación de pregrado y cuatro años más de experiencia.

El objetivo del SWEBOK de aclarar el lugar y establecer los límites de la ingeniería de software motivó la forma de organizar la guía. El material que se reconoce como dentro de la disciplina está organizado en diez KA, estas se presentan en el Cuadro . Estas KA, que se definieron cada una como un capítulo de la guía, resumen los conceptos básicos e incluyen una lista de referencias que permiten obtener información detallada.

Requerimientos de Software	Gestión de la Configuración del Software
Diseño de Software	Gestión de la Ingeniería de Software
Construcción de Software	Procesos de la Ingeniería de Software
Testing de Software	Herramientas y métodos de la Ingeniería de Software
Mantenimiento de Software	Calidad de Software

Cuadro – Áreas de Conocimiento del SWEBOK 2004

Cada KA comienza con una breve descripción de la misma y un resumen del alcance y relación con otras KA. La guía usa una organización jerárquica para descomponer las KA en un conjunto de tópicos. Esta descomposición tiene dos o tres niveles de desglose, lo que permite que sea más sencilla la búsqueda de los temas de interés.

Para cada tópico se presenta una breve descripción, junto con una lista de referencias. El grado de descripción de cada tópico es tal que permite comprender la naturaleza generalmente aceptada del mismo y que el lector pueda encontrar material del tópico en las referencias señaladas. A su vez, cada KA cuenta con una matriz que relaciona los tópicos con las referencias citadas.

Uno de los objetivos del SWEBOK es aclarar el lugar y establecer los límites de la IS con respecto a otras disciplinas. Por este motivo el SWEBOK dedica un capítulo a las disciplinas relacionadas con la IS. La guía reconoce ocho disciplinas relacionadas: Ingeniería en Computación, Ciencias de la Computación, Administración, Matemática, Gestión de Proyectos, Gestión de la Calidad, Ergonomía del Software e Ingeniería en Sistemas. Los Ingenieros de Software deben tener conocimientos en estos campos, pero no forma parte del objetivo del SWEBOK caracterizar estas disciplinas. O sea, de forma intencional el SWEBOK no cubre aquel conocimiento que no es propio de la IS pero que todo ingeniero de software debe tener [8].

El SWEBOK establece para cada tópico un nivel de conocimiento esperado expresado mediante la Taxonomía de de Bloom [12]. La Taxonomía de Bloom es una clasificación de los diferentes objetivos y habilidades que los educadores pueden proponer a sus estudiantes. Es una taxonomía jerárquica donde el aprendizaje a niveles superiores (de la taxonomía) depende de la adquisición del conocimiento de los niveles inferiores. Asignar un nivel de Bloom a cada tópico sirve de base para el desarrollo de planes de estudio, certificaciones y licenciamientos.

El SWEBOK 2004 es una versión que tuvo una amplia repercusión y que se tomó como base para importantes proyectos que ayudaron a la evolución de la IS como una profesión. Por ejemplo, en lo que refiere a educación en IS, la IEEE-CS y la ACM proponen una guía curricular para carreras en IS a nivel de pregrado (SE2004) [2] y otra a nivel de posgrado (GSWE2009) [13], ambas propuestas utilizan al SWEBOK como una de las principales fuentes para la definición del conocimiento central. Otro ejemplo son las certificaciones CSDA[5] y CSDP[6] desarrolladas por la IEEE-CS, que utilizan el SWEBOK como base para armar los exámenes de certificación.

2.3 SWEBOK Versión 3

En la construcción de la versión 2004 del SWEBOK, el comité que estuvo a cargo se propuso como meta que dicha guía continúe evolucionando para satisfacer las necesidades de la comunidad de la ingeniería de software.

Actualmente se está trabajando en la construcción de una nueva versión de la guía llamada SWEBOK Versión 3 (V3)[7], que busca reflejar los avances de la disciplina. La actualización de la guía resulta muy importante por diversos motivos: la práctica ha cambiado, surgieron otros puntos de vista y el cuerpo de conocimiento ha crecido.

---

5 http://www.computer.org/portal/web/certification/csda

6 http://www.computer.org/portal/web/certification/csdp

7 http://www.computer.org/portal/web/swebok/v3guide

La práctica ha cambiado, ya que han surgido nuevas herramientas, métodos y tipos de software. Tópicos que antes eran experimentales ahora han alcanzado el estado de generalmente aceptado y por ende deben ser incluidos en la guía.

En este tiempo se realizaron otros esfuerzos importantes para la evolución de la IS como profesión, tales como el examen CSDP y la guía curricular SE2004. Éstos ofrecen puntos de vista ligeramente diferentes del cuerpo de conocimiento. Resulta entonces importante que se aproveche la oportunidad de la revisión para examinar estos productos para ver si la V3 se puede alinear mejor a los mismos. La comparación entre estos esfuerzos ha sugerido la necesidad de la creación de nuevas KA y la revisión de otras.

Otro factor importante para la actualización de la guía es que desde el 2004 el cuerpo de conocimiento ha crecido y se han actualizado varias de las citas referidas en dicha versión.

La versión en construcción del SWEBOK V3 cuenta con quince KA que están divididas en dos categorías: aquellas que caracterizan la práctica de la ingeniería de software y aquellas que caracterizan los requerimientos de educación en ingeniería de software[8]. En el Cuadro se presenta la lista de KA y se marca con un "(*)" aquellas KA que son nuevas con respecto a la versión 2004 de la guía. Como se puede apreciar, la mayor parte de las KA creadas se encuentran en la categoría referente a los requerimientos educacionales.

Áreas de Conocimiento que caracterizan la práctica de la Ingeniería de Software	
Requerimientos de Software	Gestión de la Configuración del Software
Diseño de Software	Gestión de la Ingeniería de Software
Construcción de Software	Procesos de la Ingeniería de Software
Testing de Software	Modelos y métodos de la Ingeniería de Software
Mantenimiento de Software	Calidad de Software
Práctica profesional de la Ingeniería de Software (*)	
Áreas de Conocimiento que caracterizan los requerimientos educacionales de la Ingeniería de Software	
Economía de la Ingeniería de Software(*)	Fundamentos de la Computación(*)
Fundamentos matemáticos(*)	Fundamentos de la Ingeniería(*)

Cuadro – Áreas de Conocimiento del SWEBOK V3

Recientemente el SWEBOK V3 ha culminado el proceso de revisión pública de todas las KA que lo conforman. A setiembre de 2013 se encuentra disponible en la página del proyecto la versión alfa de todas las KA.

Los cambios de esta versión de la guía incluyen:

8http://computer.centraldesktop.com/swebokv3review/

- La actualización de la descripción de los tópicos presentados en la versión 2004.
- La inclusión de nuevos tópicos que se han vuelto generalmente aceptados.
- La remoción de tópicos que ya no son relevantes.
- La adición de 5 nuevas KA que proveen una guía sobre los conocimientos básicos de la IS y permiten mejorar la integración con las disciplinas relacionadas.
- El renombre y redistribución de algunos materiales en diferentes KA.

3. Educación de Pregrado en IS

En esta sección se presenta la principal guía curricular construida para el desarrollo de programas de pregrado en ingeniería de software y los esfuerzos que se están realizando para mantenerla vigente.

En la sección 3.1 se describe la guía *Software Engineering 2004* (SE2004) propuesta por la IEEE-CS y la ACM. En la sección 3.2 se explica el proceso de revisión que se está realizando para actualizar dicha guía.

3.1 Software Engineering 2004 (SE2004)

En 1998 la IEEE-CS y la ACM establecieron un grupo de trabajo conjunto para crear una nueva versión de sus guías curriculares para programas de pregrado en computación. En la década de los '90 la computación avanzó enormemente, emergiendo un conjunto de familias de disciplinas relacionadas a la computación [14].

El proyecto *Computing Curricula 2001* (CC2001), llevó a la elaboración de cuatro volúmenes separados de guías curriculares para disciplinas relacionadas a la computación: ciencias de la computación, sistemas de información, ingeniería en computación e ingeniería de software. Luego de la publicación de CC2001, se ha agregado un volumen más que incorpora una guía curricular para la disciplina de tecnologías de la información [3].

Software Engineering 2004, SE2004 de aquí en más, es el volumen correspondiente a la guía curricular para programas de en IS desarrollado por la IEEE-CS y la ACM. Ésta tiene como objetivo servir de guía para las instituciones académicas y organismos de acreditación sobre lo que debería constituir una educación de pregrado en IS [2].

El trabajo llevado a cabo para la construcción de este volumen pasó por tres etapas. La primera etapa se enfocó en determinar el conjunto de resultados esperados del currículo y en describir qué es lo que un egresado de una carrera de pregrado en IS debería saber. La segunda etapa implicó la determinación y especificación del cuerpo de conocimiento central (SEEK) que debería ser incluido como mínimo en los programas de pregrado en IS. La tercera etapa produjo un conjunto de recomendaciones curriculares que describen cómo se podría estructurar, en diversos contextos, un currículo en IS que incorpore el SEEK.

SE2004 establece un conjunto de siete resultados que se espera que los estudiantes logren alcanzar al completar una carrera de pregrado en IS. Éstos son presentados en el Cuadro . Como se pude apreciar los mismos son lo suficientemente genéricos como para que puedan ser adaptados a una variedad de programas en IS [2].

1	Mostrar dominio de los conocimientos y habilidades de la IS así como de las cuestiones profesionales necesarias para comenzar la práctica como ingeniero de software.
2	Poder trabajar de forma individual y como equipo para desarrollar y entregar productos de 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 la IS que integren cuestiones éticas, sociales, legales y económicas.
5	Demostrar que se comprende y se puede aplicar teorías actuales, modelos y técnicas que proveen una base para: la identificación y 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 de desarrollo típico.
7	Ser capaz de aprender nuevos modelos, técnicas y tecnologías cuando estas emergen. Apreciar la necesidad del desarrollo profesional continuo.

Cuadro – Resultados esperados al egreso de carrera de pregrado en IS

Una parte central de la guía es la definición del *Software Engineering Education Knowledge* (SEEK), que es el cuerpo de conocimiento que debería ser incluido como mínimo en los programas de pregrado en ingeniería de software. El SEEK incluye conocimientos específicos en IS y a su vez conocimientos de disciplinas relacionadas tales como matemáticas, ciencias de la computación, ingeniería y economía [14].

Para la definición de este cuerpo de conocimiento se tomó como base principal el SWEBOK 2004, pero se realizaron varios cambios. Estos cambios se deben a que el SEEK define el conocimiento a incluir en los planes de estudios de programas de pregrado, mientras que el SWEBOK caracteriza el contenido de la disciplina de ingeniería de software utilizado por los ingenieros de software en la práctica. A su vez, a diferencia del SEEK, el SWEBOK 2004 incluye únicamente lo que está dentro de los límites de la IS y se omite intencionalmente el conocimiento de dominios y disciplinas relacionadas (situación que cambia en el SWEBOK V3 como ya se mencionó) [2].

Como se puede ver en la Error: No se encuentra la fuente de referencia, el SEEK está organizado de forma jerárquica en tres niveles: área de conocimiento (KA), unidad y tópico. Las KA representan el mayor nivel de la jerarquía y corresponden a sub-disciplinas de la IS. En el Cuadro se presentan las 10 KA que conforman el SEEK. Cada una de éstas se identifica por una abreviación, como por ejemplo PRF corresponde a 'prácticas profesionales'.

Figura – Estructura jerárquica del SEEK

Las KA se descomponen en módulos más chicos llamadas unidades. Para identificar las unidades se le agrega un sufijo de dos o tres letras al prefijo de la KA, por ejemplo PRF.com es la unidad de 'habilidades de comunicación' de la KA 'prácticas profesionales'. Cada unidad se subdivide a su vez en un conjunto de tópicos, que representan el menor nivel de la jerarquía.

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

Cuadro - Áreas de Conocimiento del SEEK

Dentro del SEEK se definió una parte como central (el núcleo). Éste contiene el material esencial que los profesionales de la enseñanza en IS están de acuerdo que es necesario obtener en una carrera de pregrado en esta área; por ende es el mínimo que debe impartirse en una carrera de pregrado en IS que busca seguir la guía SE2004. En la construcción de este cuerpo de conocimiento se buscó que sea lo más chico posible, para darle a las instituciones la mayor flexibilidad posible para adaptarse a los distintos contextos. De esta forma, cada programa podrá agregar unidades adicionales que pertenezcan o no al núcleo.

Para cada unidad del SEEK se indica el tiempo mínimo de dedicación expresado en horas. En este contexto, el tiempo corresponde al dedicado en clase para presentar un material en el formato de lectura tradicional (horas de contacto). Las horas especificadas no incluyen el tiempo dedicado fuera de clase. Como guía el SE2004 sugiere estimar al tiempo de dedicación fuera de clases, en promedio, como entre dos y tres veces las horas de contacto. Estas horas representan un nivel mínimo de cobertura por lo que siempre resulta apropiado poder dedicarle más tiempo a una unidad que lo expresado en el SEEK.

A su vez, 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 Bloom que son: conocimiento (ser capaz de recordar el conocimiento aprendido), comprensión (entender la información y el significado del material presentado) y aplicación (aplicar el conocimiento para resolver

problemas). Solamente se usan los tres primeros niveles ya que buscan representar capacidades que puedan ser aprendidos a nivel de una educación de pregrado.

De manera adicional para cada tópico se indica la relevancia del tópico en el cuerpo de conocimiento central. Un tópico marcado como esencial es parte del núcleo. Un tópico deseable no es parte del núcleo, pero debería ser incluido, de ser posible, en el núcleo de un programa en particular o en caso contrario formar parte de los materiales electivos. Por último, un tópico es opcional cuando sólo debe considerarse como electivo.

SE2004 presenta un conjunto de 18 guías que deberían ser consideradas tanto para el desarrollo de un plan de estudios de pregrado como para el dictado de cursos individuales en IS. Estas guías brindan consejos sobre: el desarrollo y la enseñanza del currículo, el diseño del currículo, estrategias para la pedagogía en ingeniería de software y las expectativas en cuanto al conocimiento y habilidades de los docentes [14]. Cada una de estas guías se expande para proporcionar asesoramiento sobre su aplicación y para advertir acerca de problemas que podrían ocurrir.

Esta guía curricular provee también un conjunto de secuencias de cursos de ejemplo que muestran cómo se puede enseñar el SEEK siguiendo las guías definidas. Se discuten distintos enfoques que pueden ser usados para los cursos introductorios, intermedios y adicionales en IS. Para cada uno de estos enfoques se definen cursos de ejemplo y se mapea los temas tratados con tópicos del SEEK.

También se presentan un conjunto de patrones de currículos, que muestran ejemplos de cómo se pueden agrupar un conjunto de cursos para definir un programa de pregrado en IS en diversos contextos. Los principales aspectos que diferencian a los patrones son: el contexto internacional, si se implementa bajo un contexto de facultad de ciencias de la computación o de ingeniería, si se enseña IS en los primeros dos años y si se tiene dos semestres por año académico o tres cuartos.

Los planes de estudio en IS deben poder adaptarse a distintos ambientes para poder ser utilizados por distintas instituciones. Debido a esto, en la guía se discute tanto la enseñanza en ambientes alternativos (educación a distancia) como en instituciones alternativas (coordinación entre currículos de distintas instituciones y cooperación).

Por último esta guía cuenta con un capítulo dedicado a la implementación del programa y a la evaluación. Aquí se definen cuáles son los elementos que son considerados críticos para lograr el éxito de un plan de estudios una vez que ya se estableció el currículo. Estos son: el cuerpo docente, el cuerpo estudiantil, la infraestructura y la participación de la industria. A su vez, se aclara que para mantener la calidad del currículo éste debe ser evaluado de regularmente.

A partir de la concepción del SE2004, distintas universidades del mundo han utilizado esta guía para crear nuevas carreras de pregrado en IS así como para adaptar y comparar con los planes de estudio existentes. A continuación se presentan de manera breve los artículos encontrados que presentan usos o adaptaciones del SE2004.

En el artículo "*An Assessment of the Software Engineering Curriculum in Turkish Universities: IEEE/ACM Guidelines Perspective*" se presenta una evaluación con respecto al SE2004 tomando en cuenta el plan de estudios de 11 programas de pregrado de distintas universidades de Turquía [15]. En particular para cada programa se comparan las horas dedicadas a cada KA con las propuestas en el SEEK.A partir de esta comparación se detectó que todos los programas

cubren la mayoría de los tópicos de las KA de Fundamentos en Computación y Fundamentos en Matemática & Ingeniería. En 9 de los 11 programas la KA de Evolución de Software falta o está cubierta en forma parcial. De manera similar se detecta una baja cobertura en la KA de Procesos de Software. Sin embargo, para las áreas de conocimiento de Gestión de Software y Calidad de Software se dedica más tiempo que el indicado como mínimo en el SEEK.

En la universidad de Gannon (Pennsylvania) se diseñó un programa de pregrado en ingeniería de software que buscó alinearse con SE2004 y cumplir con los criterios establecidos para poder ser acreditado por el *AccreditationBoard for Engineering and Technology* (ABET)[9]. En el diseño del plan de estudios se debieron respetar ciertas restricciones impuestas por la Universidad tales como contar con ciertos cursos centrales de la misma y se buscó reutilizar la máxima cantidad de cursos existentes posibles. Para cada uno de los cursos reutilizados se indica la cantidad de horas de contacto y se realiza un mapeo para determinar qué unidades del SEEK cubren. Las horas restantes se distribuyen en cursos específicos para ingeniería de software de modo de tratar las KA restantes del SEEK [16].

En el artículo "*Accreditation of Monash University Software Engineering (MUSE) Program*" se presenta el plan de estudios de un programa de pregrado en IS de la universidad de Monash de Australia y los esfuerzos realizados para acreditarlo [17]. En particular se presenta como el plan de estudios fue evolucionando en un período de diez años y cómo se buscó realizar evaluaciones comparativas con varios esfuerzos internacionales tales como el SWEBOK y SE2004.
La Universidad de Nanjing de China diseñó un plan de estudios denominado NJU-SEC2006 basándose en el SE2004 [18]. En el diseño de este plan de estudios se tuvo especial cuidado en tres aspectos. En primer lugar se buscó tener cursos en IS que tomen en cuenta las distintas escalas posibles del desarrollo de software: pequeña, mediana y larga. En segundo lugar las prácticas profesionales que son difíciles de enseñar, tales como trabajo en equipo y las habilidades de comunicación, se enseñan a través de todo el programa. En tercer lugar se modificaron los cursos tradicionales en ciencias de la computación de forma tal de adaptarlos para que estén más acordes a una educación en IS pero buscando mantener los conocimientos impartidos.
Si bien parecen pocas las universidades que adaptaron o usaron el SE2004, es de nuestro conocimiento que muchas más universidades lo están usando pero no han publicado su experiencia.[10]

3.2 Revisión del SE2004

La IEEE-CS y la ACM iniciaron a comienzos de 2011 un proyecto para revisar el SE2004, de forma de mantener las guías actualizadas [19].
El equipo de revisión del SE2004 definió 3 etapas a llevar a cabo:

i. La realización de un proceso de consulta para recolectar información y opiniones de los principales *stakeholders* (pertenecientes a la academia, industria y gobierno) sobre la necesidad de modificar el modelo curricular propuesto.
ii. El análisis y evaluación de los resultados del proceso de consulta para determinar el tipo y la magnitud de los cambios necesarios.
iii. La preparación de un reporte que describa el proceso de consulta realizado, el análisis y la evaluación para presentar a las juntas educacionales de la IEEE-CS y

9 http://www.abet.org/

10 Conversación por mail con Mark Ardis: integrante del comité para la creación de GSWE2009 e integrante de la comisión para elaboración del SE2013.

de la ACM. A su vez, se deben realizar recomendaciones referidas al tipo y magnitud de las revisiones necesarias y una estimación de esfuerzo necesario.

Para realizar la primera etapa de la revisión se armó una lista de *stakeholders* y se definió una encuesta con el objetivo de recoger información sobre el estado actual y el uso del SE2004. Se recibieron respuestas de 42 países distintos, sin embargo, la mayoría de las respuestas provinieron de Estados Unidos. La mayoría de los participantes de la encuesta estuvieron de acuerdo en que las Áreas de Conocimiento definidas aún son relevantes.

En base a los resultados del proceso de consulta se detectó que la estructura general del SE2004 es sólida y no debe ser cambiada. Con respecto al SEEK, se encontró que se necesitan realizar revisiones menores. Las áreas de revisión identificadas incluyen: la incorporación de métodos de desarrollo más modernos como los métodos ágiles, la necesidad de poner un mayor énfasis en la seguridad y en la computación orientada a servicios.

En el año 2012 la IEEE-CS y la ACM aprobaron un proyecto llamado SE2013 para realizar la revisión de las guías que sería llevado a cabo por 6 voluntarios. Se definió una duración estimada de un año y medio para este proyecto. Sin embargo, el proyecto no se ha logrado finalizar aún y el trabajo se continúa actualmente. Recientemente, en mayo de 2013 se presentaron los avances del proyecto en la conferencia "*Conference on Software Engineering Education and Training*" (CSEE&T 2013) [20].

Además del proyecto SE2013, en la actualidad se están realizando otros esfuerzos relacionados a la elaboración de guías curriculares que son: CS2013 (revisión de las guías para programas de pregrado en ciencias de la computación) [21] y SWEBOK V3. Se espera que los integrantes del proyecto SE2013 tengan una estrecha colaboración con los miembros de estos proyectos [19].

4. Educación de Posgrado en IS

A nivel de educación de posgrado la guía curricular *Graduate Software Engineering2009* (GSwE2009), es un plan de estudios de referencia para programas de Maestría Profesional en IS [13]. Éste puede ser utilizado como una guía para aquellas Facultades que están diseñando o mejorando sus programas de Maestrías Profesionales en IS [13]. Se debe entender por Maestría Profesional a las Maestrías que educan a nivel de posgrado a aquellas personas que buscan ejercer una carrera en la práctica (esto en contraposición a una Maestría Académica).

El GSwE2009 comenzó a desarrollarse en el año 2007 como parte del proyecto *Integrated Software and Systems Engineering Curriculum* (iSEEc) en el *Stevens Institute of Technology*. Este proyecto fue financiado por la oficina del departamento de defensa de Estados Unidos y participaron por más de dos años más de 40 autores pertenecientes a la academia y a la industria.

En otoño del año 2009 la IEEE-CS y la ACM adoptaron el GSwE2009 como parte de sus recomendaciones curriculares conjuntas en computación [22]. Esto significa que el GSwE2009 cumple con sus expectativas en cuanto a la calidad del proceso de desarrollo de las guías y del producto en sí. Futuras versiones del GSwE2009 serán mantenidas por estas dos sociedades profesionales.

El GSWE2009 se basa en un conjunto de recomendaciones para la creación de planes de estudio para programas de maestrías en IS desarrollado por en IS desarrollado por el SEI [23] en el SWEBOK 2004 y en la propuesta de Plan de estudios de enseñanza de pregrado SE2004 [2].

Esta guía incluye una descripción de: la arquitectura curricular, los conocimientos esperados al ingreso, el cuerpo de conocimiento central (CBOK[11]) y los resultados esperados al egreso. También proporciona orientación para la construcción, el mantenimiento y evolución de las recomendaciones. A su vez, cuenta con un anexo que describe las áreas de conocimiento definidas en el CBOK pero que no están contenidas en el SWEBOK 2004.

La arquitectura de la guía curricular está compuesta por: contenido preparatorio, contenido central, contenido específico de la Universidad, contenido electivo y una experiencia final. Esto se presenta en la Figura .

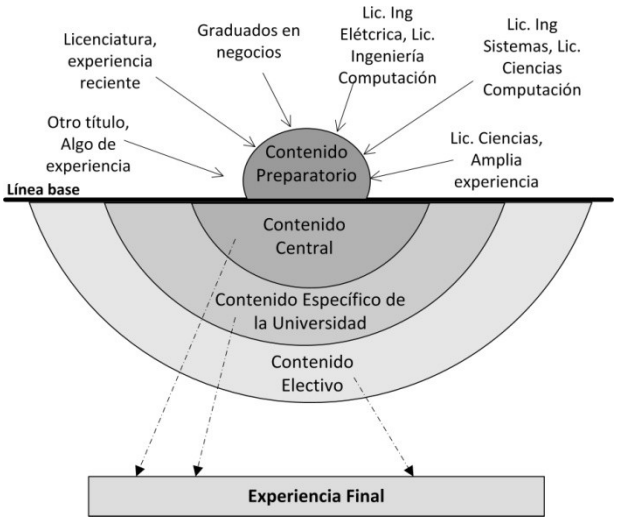

Figura – Arquitectura del GSwE2009

El contenido preparatorio es aquel que debe ser dominado por el estudiante antes de entrar al programa de maestría.

El GSwE2009 identifica habilidades y conocimientos fundamentales que todos los graduados de maestrías en ingeniería de software deberían tener. Estos definen el contenido central o cuerpo de conocimiento central (CBOK).

El contenido específico de la Universidad se corresponde con aquellos contenidos que la institución podría incluir de forma tal de adaptar su programa para cumplir con sus objetivos específicos.

El contenido electivo permite que los estudiantes se enfoquen en sus intereses particulares, dentro del enfoque establecido por el programa. El GSwE2009 recomienda que las maestrías cuenten con una experiencia final. Esta puede ser un proyecto, una práctica o una tesis. La carga horaria estimada de esta experiencia final es entre 3 a 6 créditos americanos.

El GSwE2009 asume que los estudiantes que ingresan a la maestría cumplen con las siguientes condiciones:

- Son egresados de una carrera de pregrado en informática, ingeniería o científica con algún estudio en computación.
- Han realizado algún curso introductorio de IS. Tienen al menos dos años de experiencia práctica en algún aspecto de la IS.

11 CBOK sigla en inglés correspondiente a **C**ore **Bo**dy of **K**nowledge

Los conocimientos esperados para el ingreso fueron definidos de forma tal que se pueda lograr cumplir con los 10 resultados esperados definidos para el egreso.

El GSwE2009 recomienda que los programas tengan entre 33 y 36 créditos americanos. Se espera que un estudiante de tiempo completo pueda terminar el programa en un tiempo de entre 18 a 24 meses. El número de créditos recomendados, combinado con los resultados esperados al egreso, determinan cuáles deberían ser los conocimientos previos de los estudiantes que ingresan a la maestría.

El conocimiento previo, al igual que el CBOK, se presenta dividido en KA. Para cada una de estas KA 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 con los resultados al egreso. Está organizado de forma jerárquica en 3 niveles: KA, unidades y tópicos. Este cuerpo de conocimiento contiene 11 KA. Para cada unidad de las KA se indica la profundidad del conocimiento que se espera que los estudiantes logren en aproximadamente 200 horas de contacto (menos del 50 % del total de los créditos), expresada en la Taxonomía de Bloom.

El CBOK fue desarrollado tomando como base principal el SWEBOK versión 2004 y realizando algunas modificaciones que se creyeron convenientes, tales como el adicionado de nuevas KA y modificación de algunas unidades y tópicos. Estos cambios se realizaron con el objetivo de dar soporte al logro de los resultados esperados establecidos el GSWE2009 y para adaptarse a las necesidades y opiniones de la academia, la industria y las sociedades profesionales. En el Cuadro se muestran las KA del CBOK y se las compara con las de SWEBOK 2004. Como se puede ver se añadieron las KA de Ética y Conducta Profesional e Ingeniería de Sistemas y se quitó la KA de Herramientas y Métodos del a Ingeniería de Software.

KA CBOK	**KA SWEBOK 2004**
Ética y Conducta Profesional	
Ingeniería de Sistemas	
Ingeniería de Requisitos	Requerimientos de 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

Cuadro –Diferencias entre las KA del CBOK y del SWEBOK 2004

En el currículo se establecen 10 resultados esperados al egreso. Es decir, se establecen resultados que se espera asimile el estudiante al culminar la maestría. Estos resultados cubren

diversos aspectos como por ejemplo: técnicos, éticos y de aprendizaje. En el Cuadro se presentan los 10 resultados, junto con una breve descripción de cada uno de ellos.

Resultado	Descripción
CBOK	Dominar el CBOK. El CBOK especifica niveles de Bloom que deberían ser cumplidos para cada KA.
Dominio	Dominar la IS 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 un comportamiento ético profesional
Ingeniería en 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.

Cuadro - Resultados esperados al egreso

Al igual que con el caso de SE2004, si bien se sabe que la guía GSWE2009 ha sido utilizada como base para la construcción/adaptación de maestrías profesionales en IS, no hay casi publicaciones en revistas o conferencias que presenten el uso de la guía.

En nuestra revisión bibliográfica encontramos un artículo que habla del uso de GSWE2009. El artículo "*Using GSwE2009 in the Creation and Modification of Graduate Software Engineering Programs and Related Curricula*" se presenta cómo cuatro Universidades de tres países distintos han usado esta guía para la construcción y adaptación de programas de Maestrías en IS [24].

5. Otros tipos de educación en IS

El futuro de nuestra profesión depende de la educación de la próxima generación a escala (a nivel mundial). Los jóvenes no deberían tener que esperar hasta la universidad para tener un primer acercamiento con la computación [25].

Las personas aprenden muchas de las cosas más importantes muy temprano en la vida. La computación es una nueva forma de alfabetización y forma parte de los conocimientos fundamentales del día de hoy [26].

En la actualidad diversos países están realizando distintos esfuerzos para incluir a la computación como parte importante de la educación temprana de las personas. En Estonia se está enseñando programación a los adolescentes de primer grado [25]. En Estados Unidos se han creado escuelas secundarias tales como la *Academy for Software Engineering* (AFSE)[12] diseñadas para preparar a los estudiantes para carreras tales como ciencias de la computación o ingeniería de software [27]. En el Uruguay, se ha creado la opción de Bachillerato Tecnológico

12 http://www.afsenyc.org/

en Informática que introduce a los estudiantes en temas relacionados con la computación tales como la programación, los sistemas operativos y las bases de datos[13].

No siempre la decisión de comenzar una carrera universitaria es una tarea sencilla para los estudiantes. Estos deben ser alentados para seguir sus intereses tomando en cuenta sus circunstancias particulares y ciertos factores tales como la capacidad, el potencial para el empleo y el costo al tomar esa decisión [26].

Las personas poseen una gran variedad de habilidades e intereses y para muchas de estas la obtención de un título universitario no es una opción viable ni necesaria. Para la obtención de un título universitario no alcanza solamente con tener una alta capacidad intelectual, sino que se necesita contar con tener ciertas características personales que influyen en la posibilidad de obtener un título. Dentro de estas características se incluye, por ejemplo el ser organizado, dedicado, contar con habilidades para el estudio y para tomar exámenes [26].

Para tomar la decisión sobre el camino a seguir en cuanto a la carrera se debe tener presente que la obtención de un título universitario no es la única opción de carrera [26]. Hoy en día existen distintas opciones tales como las tecnicaturas o realización de cursos cortos y especializados.

6. ¿Qué tan madura es la IS como profesión?

En esta sección se presentan distintos aspectos que hacen a la madurez de la ingeniería de software como profesión. En la sección 6.1 se mencionan los elementos que componen una profesión, en la sección 6.2 se presenta el estado actual de la profesión de ingeniería de software y en la sección 6.3 se presenta un resumen de los principales hitos asociados al establecimiento de la profesión.

El capítulo "Ruta de trabajo por la profesionalización de la ingeniería de software" de este mismo libro aborda este tema desde un punto de vista diferente al de esta sección. Dicho capítulo se extrae del recientemente desarrollado "Manifiesto por la profesionalización del desarrollo de software".

6.1 Elementos que componen a una profesión

La ingeniería de software se conoce, tanto en la industria como en la academia, como una profesión inmadura. En 1996, con el objetivo de entender y estudiar dicha afirmación, Gary Ford y Norman Gibbs publicaron un reporte titulado "*A Mature Profession of Software Engineering*" [28]. En dicho trabajo se estudiaron diversas profesiones bien establecidas tales como la medicina, el derecho, la arquitectura y la contabilidad. A partir de este estudio observaron que, a pesar de ser disciplinas muy distintas, hay componentes comunes a todas. Estas componentes logran que los profesionales sigan un camino de desarrollo profesional bastante similar.

El reporte presenta un modelo que permite caracterizar la madurez de una profesión en términos de 8 componentes:

1. Formación Profesional Inicial.

13 http://www.utu.edu.uy/webnew/modulos/utu/Areas%20de%20Cursos/Cursos%20y%20Programas/Informatica/Tipo%20de%20Cursos/BACHILLERATO/Informatica%20EMT%2005/Informatica%20EMT%202004%2005.htm

La formación profesional inicial refiere a los estudios que son completados de forma previa al comienzo de la práctica profesional. O sea, los profesionales generalmente comienzan sus carreras al completar un programa universitario en el campo elegido.

2. Acreditación.
   Los programas universitarios pueden ser acreditados por organismos de control que determinan si el programa ofrece una educación adecuada. Esto busca asegurar que los profesionales que egresen de programas acreditados puedan comenzar su vida profesional con los conocimientos necesarios.
3. Desarrollo de Habilidades (competencias).
   En la mayoría de las profesiones no alcanza solamente con la educación para poder desarrollar de forma completa las competencias necesarias. Los profesionales recién egresados necesitan práctica que les permita aplicar sus conocimientos antes de estar preparados para asumir la responsabilidad primaria en la realización de sus trabajos.
4. Certificación.
   La certificación es un proceso voluntario (administrado por una profesión) que permite determinar quién está plenamente capacitado para participar de una profesión. Una vez completada la formación profesional inicial y el desarrollo de habilidades, el profesional debe pasar uno o más exámenes que permiten asegurar que ha adquirido un mínimo nivel de conocimiento.
5. Concesión de Licencias.
   El licenciamiento es similar a la certificación excepto que este es obligatorio y es administrado por una autoridad gubernamental. La concesión de licencias tiene como objetivo proteger al público (proteger la vida, la salud, la propiedad y promover el bienestar general). El derecho y la medicina son dos profesiones en las cuales el licenciamiento es muy común. En estas profesiones es necesario contar con una licencia para poder ejercer como profesional. La ingeniería en Estados Unidos es una profesión que tiene licenciamiento. Sin embargo, no es obligatorio que todos los ingenieros estén licenciados. Las leyes de los distintos estados difieren en la definición de qué tipos de trabajos pueden ser sólo realizados por ingenieros licenciados.
6. Desarrollo Profesional.
   La educación 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, estas comienzan promoviendo el intercambio de conocimiento; luego evolucionan e incluyen funciones tales 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 de forma tal de asegurar que los profesionales se comporten de forma responsable.

Ford y Gibbs caracterizan a cada uno de estos componentes en alguna de las siguientes etapas evolutivas:

1. No existente: el componente no existe de ninguna forma.
2. Ad Hoc: existe una forma relacionada del componente, pero no se identifica con la profesión dada.
3. Específico: el componente existe y está claramente identificado con la profesión.
4. 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, Ford y Gibbs hacen una evaluación que presenta a la fecha de realizada la misma (1996) el estado para cada uno de los componentes de la profesión de ingeniería de software. Dicha evaluación determinó que todos los componentes estaban en la

etapa “1. Ad hoc” salvo el componente Desarrollo Profesional que estaba en la etapa “2. Específico”.

6.2 Estado actual de la IS como profesión

Desde la evaluación realizada por Ford y Gibbs en 1996 [28] a setiembre de 2013 la ingeniería de software ha tenido avances como profesión en lo que refiere a su madurez.

En el año 1999 McConnel y Tripp publican el artículo *“Professional Software Engineering: Fact or Fiction”* en el cual describen la evaluación realizada por Ford y Gibbs y el estado de los componentes de la profesión para la IS a esa fecha [29]. Luego en el 2004 se vuelve a publicar dicho artículo en el libro *Software Engineering, Volume I, 3rd Edition* [30].

En el 2013 se publica en el capítulo “11.a *Software Engineering as a Profession*” del libro “*Software Engineering Essentials Volume II: The Supporting Processes*” una actualización de lo presentado en el artículo publicado en 2004. En esta edición presentan el estado actual de los componentes de la profesión de IS [31]. Los autores actualizan la lista de referencias (que van hasta el año 2010) e incluyen acontecimientos recientes que influyen sobre el estado actual de los componentes. 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 McConnel y Tripp en el último libro y se incluyen nuevas referencias actualizadas relacionadas a dichos componentes.

6.2.1 Formación Profesional Inicial

Actualmente la mayoría de los practicantes tienen una carrera de pregrado en ciencias de la computación o en tecnologías de la información. La cantidad de egresados de carreras de pregrado en ingeniería de software es aún pequeña, pero este número ha venido creciendo. A modo de referencia, en el año 2010 en Estados Unidos existían 244 programas de pregrado, 70 programas online, 230 programas de maestría y 41 programas de doctorado en IS [31].

En 1987 el *Imperial College London* introdujo la primera carrera de pregrado en IS. Desde entonces, se han establecido carreras de pregrado en IS en muchas universidades del mundo. A través de un trabajo en conjunto, la IEEE-CS y la ACM desarrollaron la guía curricular para carreras de pregrado en IS, *Software Engineering* 2004(SE2004) [2]. A la fecha esta guía se encuentra en un proceso de revisión y actualización (ver sección 3).

La primera maestría en IS fue establecida en la universidad de Seattle en 1979 y el primer programa de doctorado se estableció en 1998 en la *US Naval Postgraduate School* (NPS). Al igual que con las carreras de pregrado, hoy en día existen una variedad de universidades a lo largo del mundo que ofrecen maestrías y doctorados en IS [31]. En lo que refiere a los planes de estudio para programas de maestrías profesionales en IS, se cuenta con la guía *Graduate Software Engineering 2009* (GSwE2009) [13]. Esta guía fue adoptada en el 2009 por la IEEE-CS y la ACM como parte de sus recomendaciones curriculares conjuntas en computación (ver sección 4).

6.2.2 Acreditación

La acreditación de los programas está comenzando a ser cada vez más importante en los Estados Unidos, ya que cada vez hay más programas en IS y por motivos de competitividad y reconocimiento las instituciones educativas buscan acreditar sus programas [31].

El criterio para acreditación de programas en IS en Estados Unidos fue desarrollado en 1998-1999 y los primeros programas acreditados fueron en 2003 [1]. La acreditación de los programas de IS es manejada por el *Engineering Accreditation Commission* (EAC) de ABET[14]. ABET es una organización dedicada a la acreditación de programas de educación universitaria o terciaria en disciplinas de ciencias aplicadas, ciencias de la computación, ingeniería y tecnología. A setiembre de 2013, ABET ha acreditado 27 programas de pregrado en IS[15].

Luego de que empezó el esfuerzo para acreditar programas en IS, ABET se fusionó en el 2001 con CSAB, el organismo acreditador para ciencias de la computación. A partir de esta fusión, se creó la *Computing Accreditation Commission* (CAC). Al igual que otras disciplinas de ingeniería, la IS es manejada por el EAC, mientras que el CAC maneja la acreditación de los programas de ciencias de la computación, sistemas de información y tecnologías de la información [32].

En Canadá el *Canadian Engineering Accreditation Board* (CEAB), que es el organismo acreditador para programas de ingeniería, ha acreditado la fecha un total de 12 programas de pregrado en IS[16] [33].

6.2.3 Desarrollo de Habilidades (competencias)

Dada la naturaleza dinámica de de la tecnología, la industria dedica recursos significativos para el entrenamiento de los ingenieros de software [29].

La IEEE-CS trabajó en un proyecto para determinar el conocimiento que es importante para un ingeniero de software profesional. Este proyecto llevó a la elaboración de la guía del cuerpo de conocimiento de la ingeniería de software, el SWEBOK 2004 [8]. Esta guía define los conocimientos necesarios para la práctica de la ingeniería de software y se ha utilizado para la construcción de guías curriculares para programas de pregrado y posgrado en IS así como para la definición de exámenes para certificación (ver sección 2.2). Actualmente se está trabajando en la construcción de una nueva versión de la guía llamada SWEBOK Versión 3 (ver sección 2.3).

6.2.4 Certificación

Las certificaciones profesionales son realizadas por los profesionales que desean demostrar su dominio en un cuerpo de conocimiento en particular reconocido por la comunidad de profesionales [34].

Existen 2 tipos distintos de certificaciones: las de base general y las específicas de productos. Las certificaciones de base general se basan en los cuerpos de conocimientos que cubren toda una disciplina profesional o una especialidad dentro de la disciplina. Por otro lado las certificaciones específicas de productos, tal como su nombre lo indica, se basan en un producto o en una línea de productos [34].

14 http://www.abet.org/

15 http://main.abet.org/aps/Accreditedprogramsearch.aspx

16 http://www.engineerscanada.ca/files/w_2012AccreditedEngineeringPrograms.pdf

Por lo general las certificaciones de base general son otorgadas por sociedades profesionales y se rigen por normas nacionales e internacionales. A modo de ejemplo, algunas certificaciones conocidas son el *Project Management Professional,* el *Certified Financial Planner* y para lo que es la ingeniería de software el *Certified Software Development Professional* (CSDP)[17].

Los candidatos que aplican a una certificación de base general deben cumplir con requisitos establecidos referentes a experiencia y formación. Los conocimientos con respecto al cuerpo de conocimiento son generalmente evaluados mediante un examen, aunque algunos programas de certificación usan revisión por pares para evaluar los conocimientos y/o la experiencia profesional. A su vez, la mayoría de las certificaciones requieren que las personas certificadas demuestren su actividad profesional y que realizan una formación continua para poder mantener la certificación [34].

La IEEE-CS ha desarrollado las certificaciones *Certified Software Development Professional* (CSDP) y *Certified Software Development Associate* (CSDA)[18]. Estas dos certificaciones son las primeras que cumplen con el estándar ISO/IEC 24773 que establece métodos para la certificación de profesionales en ingeniería de software a nivel mundial[19].

La certificación CSDP está pensada para profesionales de software o ingenieros con más de 4 años de experiencia que buscan confirmar su dominio en las prácticas estándares de desarrollo de software y avanzar en sus carreras. Esta certificación fue introducida por la IEEE-CS en el año 2002.

La certificación CSDP está basada en el SWEBOK y consiste en un examen de 180 preguntas a ser contestadas en 3,5 horas. Recientemente se actualizó esta certificación para incluir las 15 áreas de conocimiento que contiene la nueva versión de la guía SWEBOK[20]. Esta certificación es ofrecida por distintos centros en muchos países [34].

La certificación CSDA sirve para achicar la brecha entre la educación recibida y los requisitos de trabajo del mundo real. Esta certificación está prevista para recién graduados o personas que están terminando una carrera de pregrado en IS y también para profesionales que no han realizado una carrera de pregrado pero que cuentan con más de 2 años de experiencia en programación. La certificación CSDA es ofrecida desde el año 2007 y puede ser tomada como un primer paso para luego obtener la certificación CSDP. Al igual que el CSDP, esta certificación está basada en el SWEBOK y fue recientemente actualizada para alinearse con la nueva versión de la guía.

La IEEE-CS tiene alianzas con diversos *Registred Education Providers* (REP) que brindan cursos para las certificaciones CSDA, CSDP y para el programa *SWEBOK Certificate Program(*SCP)[21].

---

17 http://www.computer.org/portal/web/certification/csdp

18http://www.computer.org/portal/web/certification/csda

19http://www.computer.org/portal/web/certification/news_room#6-8-11

20 http://www.computer.org/portal/web/certification/news_room

21http://www.computer.org/portal/web/certification/training/reps

Los REP pueden ser empresas que brindan capacitaciones, escuelas técnicas que brindan cursos especializados, colegios, universidades o corporaciones. Éstos licencian el material para capacitación de la IEEE-CS y realizan cursos dirigidos por un instructor. Los cursos pueden tener fines de lucro o servir para uso interno[22].
A setiembre de 2013 hay un total de 19 REP que están agrupados en 4 zonas:

- América
- India &Sri Lanka
- Europa, Medio Oriente, África
- China

El programa SCP es un conjunto 4 cursos de 25 horas cada uno el cual busca introducir a los estudiantes en las 15 KA de la última versión del SWEBOK. Está previsto para estudiantes de computación que pertenecen a mercados emergentes, tales como India, y que necesitan contar con una base sólida en los fundamentos del ciclo de vida de la IS.

En 1998 la IEEE-CS diseñó una hoja de ruta de certificaciones para los profesionales en el desarrollo de software que incluye las certificaciones CSDA, CSDP y en un futuro la certificación CSDM[23]. La certificación CSDM tendría como público objetivo a los especialistas que deseen marcar la maestría en alguna sub-disciplina de la IS como por ejemplo en Arquitectura de Software. Esta certificación estaba prevista para finales de 2010 pero finalmente no se ha desarrollado y se ha quitado de la hoja de ruta de certificaciones. Se ha dejado de lado ya que la IEEE-CS considera que sería de valor sólo para una pequeña proporción de la comunidad de IS mientras que actualmente esta sociedad profesional se está centrando en la definición de certificados de baja gama y también en la certificación de temas emergentes tales como la computación en la nube[24].

Además de estas certificaciones existen otros organismos que brindan certificaciones en distintos países. Por ejemplo, en Alemania el *International Software Quality Institute* (iSQI) brinda certificaciones a ingenieros de software profesionales en distintas especialidades tales como arquitectura de software, gestión de proyectos y *testing* [34]. En el Reino Unido la *Institution of Electrical Engineers* lleva a cabo un programa de certificación de ingenieros de software y en Australia, la *Australian Computer Society* ofrece un programa de certificación en tecnologías de la información con una sub-especialización en IS [31].

También existe otra rama de certificaciones de base general que son aquellas que están asociadas a sub-disciplinas de la IS. En la actualidad hay una amplia variedad de certificaciones de este tipo, lo cual demuestra el avance que ha tenido la disciplina. A continuación se presentan, simplemente a modo de ejemplo, algunas de las certificaciones existentes de este tipo.

El *International Software Testing Qualifications Board* (ISTQB)[25] ha creado el esquema más exitoso para la certificación internacional de *testers* de software. Actualmente hay 2 niveles de

22 http://www.youtube.com/watch?v=VMuJCYYJFuE#t=170

23 http://www.computer.org/portal/web/cspresident/1/-/blogs/certification-builds-bridges-for-professional-development

24 Conversación por mail con Chris Ruoff, Sr. Manager Sales and Channel Development de la IEEE Computer Society

25 http://www.istqb.org/

certificación: una a nivel fundacional (de 1 módulo) y otra para nivel avanzado (de 3 módulos). Además se está desarrollando una certificación para nivel experto. A marzo de 2013 el ISTQB ha emitido más de 295.000 certificaciones en 70 países del mundo.

En lo que refiere a Ingeniería de Requisitos, el *International Requirements Engineering Board* (IREB) ha creado la certificación *Certified Professional for Requirements Engineering*[26]. Esta consiste de 3 niveles: fundacional, avanzado y experto (el nivel experto no se encuentra aún disponible) existiendo un programa definido para cada nivel. La certificación de nivel fundacional se encuentra disponible en diversos idiomas y ha sido otorgada a más de 13.000 personas a lo largo del mundo.

El *Software Engineering Institute* (SEI) ha elaborado 3 certificaciones relacionadas con la arquitectura de software[27]. La certificación *Software Architecture Professional* proporciona una sólida base teórica y práctica en arquitectura de software. La certificación SOA *Architect Professional* cubre los temas teóricos y prácticos necesarios que un profesional pueda desempeñarse de manera exitosa en la definición de arquitecturas orientadas a servicios. La certificación *ATAM Evaluator* capacita al participante para que este pueda participar como miembro de un equipo de evaluación de arquitectura siguiendo el método ATAM.

Las certificaciones relacionadas con un producto (o con líneas de productos) generalmente usan exámenes para evaluar la familiaridad del candidato con el producto y su uso [34]. Microsoft y Apple son ejemplos de empresas que brindan certificaciones para sus productos.

6.2.5 Concesión de Licencias

La concesión de licencias en el último tiempo ha sido (y sigue siendo) un tema controversial para la IS. Existen distintas opiniones sobre si el conocimiento actual de la IS es lo suficientemente maduro como para poder licenciar. Es decir, ¿un profesional con una licencia de ingeniero de software tendrá el conocimiento necesario como para proteger al público de riesgos de software, o bien, el cuerpo de conocimiento de la IS no es aún lo suficientemente maduro como para servir de apoyo para cumplir con esta responsabilidad [1]?

En los Estados Unidos, Texas es el único estado que desde 1998 otorga licencias a ingenieros de software [35]. A su vez, Texas y Florida regulan el uso de los títulos de "ingeniero en computación" e "ingeniero de software" [31].

En 1998 no existía un examen de IS reconocido a nivel nacional en Estados Unidos para licenciar ingenieros de software. Por tal motivo el *Texas Board of Professional Engineers* (TBPE) determina que solamente los ingenieros de software altamente experimentados pueden ser candidatos para obtener la licencia como ingenieros de software.

Desde 1998 hasta el 2006 para obtener la licencia de ingeniero de software en Texas se necesitaba cumplir con los siguientes requisitos [35]:

- Poseer un título de pregrado en ciencias de la computación, ingeniería u otras titulaciones de pregrado que la junta considere adecuada.
- Al menos 16 años de experiencia probada en la realización de trabajos de ingeniería (o al menos 12 años de dicha experiencia si se posee una carrera de pregrado aprobada por ABET).
- Referencias de al menos 9 personas, 5 de las cuales deben ser ingenieros licenciados.
- Credenciales educativas y de otro tipo que se consideren necesarias.

26 http://www.certified-re.de/en/

27 http://www.sei.cmu.edu/training/certificates/architecture/index.cfm

A partir del año 2006 el TBPE cambió las reglas para el licenciamiento de ingenieros de software y comenzó a exigir que todos los candidatos pasen el examen general de *Professional Engineer* (PE)[28].

Otros países han procedido con relativa facilidad al otorgamiento de licencias para ingenieros de software. Por ejemplo, en Canadá el término "ingeniero" es propiedad del *Engineers Canada* y los programas acreditados son ofrecidos por facultades de ingeniería. Las provincias de Ontario, Alberta y British Columbia licencian ingenieros de software. En el Reino Unido los graduados de programas acreditados por la *British Computer Society* (BCS) y que posean un nivel de experiencia apropiado pueden aplicar para obtener el estado de *Chartered Engineer* [34].

Las sociedades profesionales IEEE-CS y ACM han marcado distintas posturas en lo que refiere al licenciamiento de ingenieros de software. La IEEE-CS se ha manifestado a favor y la ACM en contra.

La ACM considera que no conviene licenciar a ingenieros de software. En 1999 el consejo de la ACM comunicó que se opone al licenciamiento de ingenieros de software porque considera que es prematuro y que no sería efectivo para hacer frente a los problemas de calidad y fiabilidad del software [10].

En el año 2000 la ACM realiza un análisis para determinar si conviene licenciar a ingenieros de software que realicen software de seguridad crítica. En base a este análisis la ACM concluye que no conviene licenciar por diversos motivos. En primer lugar, para obtener el licenciamiento hay que tomar el examen "*Fundamentals of Engineering*" (FE). Este examen tiene muchos temas tales como la termodinámica y la mecánica de los fluidos que no son relevantes para la IS. Lograr el dominio en estos temas puede llevar a que se le reste valor a estudiar otras áreas más relevantes para la disciplina. En segundo lugar la ACM consideró que el estado del conocimiento y la práctica es inmaduro como para otorgar licencias. La posición contraria al licenciamiento llevó a que en el año 2000 la ACM se retire del SWECC ya que el consejo de la sociedad profesional sintió que las actividades dicho comité se habían vuelto demasiado relacionadas a la promoción del licenciamiento de ingenieros de software [10].

Por otro lado, la IEEE-CS ha estado trabajando activamente en una iniciativa para licenciar ingenieros de software que pasen un examen de competencias. Desde 1999 trabaja para desarrollar el examen "*Principles and Practices of Software Engineering"* en conjunto con IEEE-USA y el *National Council of Examiners for Engineering and Surveying* (NCEES), bajo el patrocinio de la *National Society of Professional Engineers* y el *Texas Board of Professional Engineers*[29]. El examen "*Principles and Practices of Software Engineering*" comenzó a dictarse en Abril de 2013[30]. En esta primera edición del examen se presentaron 12 personas y 6 lo aprobaron[31].

Para poder licenciarse como ingeniero de software se debe pasar por dos exámenes. El primero es el *Fundamentals of Engineering* (FE) (como ya fue mencionado, examen que la ACM no lo

---

28 http://www.nspe.org/PEmagazine/pe_1207_Software_License.html

29http://www.computer.org/portal/web/pressroom/New-Software-Engineering-Exam-Approved-for-Licensure

30http://ncees.org/about-ncees/news/ncees-introduces-pe-exam-for-software-engineering/

31 Estos datos fueron recabados a partir de una conversación por mail con Phillip A Laplante CSDP, P.E., Ph.D., Chair of the Software Engineering Licensure Examination Development Committee

considera interesante en su totalidad para los ingenieros de software), que es un examen que cubre un espectro grande de temas y que es generalmente tomado luego de graduarse de programas acreditado por ABET. El segundo examen es el "*Principles and Practices of Sotware Engineering*" que es un examen específico de la disciplina de IS y que puede ser tomado sólo luego de contar con varios años de experiencia laboral en el tema[32].

No todos los estados americanos requieren que se tome este examen. Por ahora son 10 los que han indicado que lo van a ofrecer. A su vez otros países han manifestado interés en el proceso de licenciamiento. Se debe tener presente que aún en los estados americanos que licencien ingenieros de software, dicho licenciamiento solamente será requerido si se trabaja en sistemas de software críticos que afectan 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 pequeñas empresas. La mayoría de los estados no obligan a licenciar a las personas que trabajan como empleados de grandes empresas o agencias del gobierno ya que en esos casos es el empleador el que tiene la obligación y responsabilidad de proteger al público [36].

6.2.6 Desarrollo Profesional

La educación profesional continua permite mantener o mejorar el conocimiento y las habilidades de los profesionales luego de haber comenzado la práctica profesional.

Los requerimientos de educación continua ayudan a asegurar que una vez que se completa la formación profesional inicial y se desarrollan las habilidades necesarias para ejercer en la profesión, los profesionales mantengan un nivel mínimo de competencia a lo largo de su carrera [31].

Hoy en día existen muchas formas de desarrollo profesional para los ingenieros de software. Entre estas se encuentran la educación continua, los programas de maestría y la asistencia a conferencias.

Con respecto a la educación continua en el caso de la certificación CSDP, se requiere de una re-certificación cada 3 años. A su vez, establece que para mantener el certificado los profesionales deben realizar como mínimo 30 *Professional Developlemnt Units* (PDU) en el período de 3 años por el cual vale la misma[33].

Para el desarrollo profesional también es importante aprender los estándares de la práctica de la profesión. La IEEE ha participado activamente en el desarrollo de estándares para ingeniería de software por más de 35 años [31].

6.2.7 Sociedades Profesionales

La IEEE-CS y la ACM son las dos sociedades profesionales más importantes relacionadas con la ingeniería de software. Ambas disponen de conferencias, revistas, grupos de interés y bibliotecas digitales. Estas sociedades profesionales han trabajado activamente en el establecimiento de la ingeniería de software como una profesión [29].

Como se mencionó anteriormente, con el objetivo de fomentar la evolución de la IS como profesión estas sociedades forman en 1998 el SWECC. El SWECC lanzó proyectos muy importantes para el establecimiento de la profesión (ver sección 2.1).

32 http://theinstitute.ieee.org/career-and-education/career-guidance/licensing-software-engineers-is-in-the-works

33 http://www.computer.org/portal/web/certification/resources

En la actualidad la IEEE-CS es la sociedad profesional líder en lo que refiere a la certificación y licenciamiento de ingenieros de software [37]. A su vez, trabaja fuertemente en la definición de estándares.

6.2.8 Código de Ética

El SWECC fue el responsable por la creación del grupo de trabajo *Software Engineering Ethics and Professional Practice* (SEEP). Este desarrolló el Código de Ética y Práctica Profesional [38].

A finales de 1998 el Código de Ética y Práctica Profesional fue adoptado por la IEEE-CS y la ACM. Esto representó un hito en la computación ya que fue el primer código de conducta desarrollado por un equipo internacional (con representantes de todos los continentes) y que fue examinado a lo largo de su creación por practicantes de todo el mundo. A su vez, este código de ética fue avalado por muchos cuerpos profesionales y asociaciones a lo largo del mundo [39].

Este código sirve como apoyo para lograr una profesión madura ya que define en forma explícita las obligaciones éticas de un ingeniero de software [31].

El código de ética está disponible en 2 formas: una versión corta que resume las aspiraciones a un alto nivel de abstracción y una versión completa que incluye cláusulas adicionales[34].

6.3 Evolución de la Profesión de IS

En la Figura se presenta una línea del tiempo con los principales hitos y acontecimientos asociados a la profesión de IS desde el establecimiento de la misma (1968) hasta setiembre de 2013. Como puede apreciar en la misma, diversos acontecimientos extremadamente importantes han sucedido desde 1968 cuando se acuñó el término IS por primera vez. También se puede apreciar una aceleración importante a partir de 1996. Esto marca que la profesión y la educación en ingeniería de software continúan un camino de maduración claro y sostenido. Este camino no se puede transitar en unos pocos años sino que hay que dar espacios de tiempo de varias décadas para que la profesión y la educación en IS puedan catalogarse como maduras.

34 http://www.computer.org/portal/web/getcertified/resources/code_of_ethics

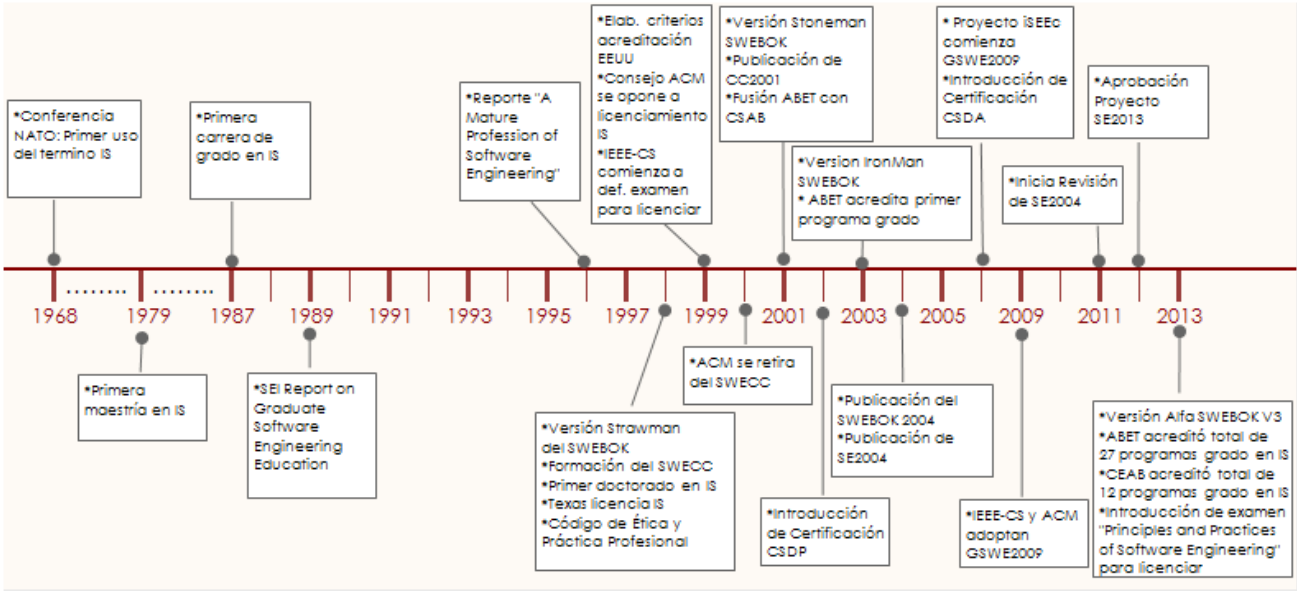

Figura – Evolución de la profesión de IS

7. Conclusiones

En este capítulo se presentó el estado actual de la educación en IS así como distintos aspectos relevantes para la educación en esta disciplina.

El término IS fue usado por primera vez en 1968. Desde entonces la educación y la profesión en IS han evolucionado y madurado de manera sostenida. Esta evolución acompaña las necesidades que surgen de la academia, la industria y las sociedades profesionales.

Varios hitos importantes sucedieron desde 1968 hasta el presente. Un hito fundamental fue el desarrollo de la guía para el cuerpo de conocimiento en IS: el SWEBOK. Contar con el cuerpo de conocimiento permite establecer qué es lo que tiene que conocer y cuáles habilidades debe tener un profesional de la disciplina. Además, contar con este cuerpo permite elaborar planes de estudios, certificaciones y licencias a partir del mismo. Actualmente el SWEBOK está en revisión y se espera que la nueva versión esté pronta para fines de 2013 o principios de 2014.

En los últimos años se han elaborado guías curriculares propuestas por la IEEE-CS y la ACM para educación de pregrado (SE2004) y de posgrado (GSwE009). Ambas guías se basan en el SWEBOK y cada una define un cuerpo de conocimiento central que debe ser incluido como mínimo los programas de pregrado y posgrado respectivamente. Estas guías han sido utilizadas por distintas universidades del mundo para crear nuevas carreras en IS así como para adaptar y comparar con las existentes. Actualmente la guía de pregrado se encuentra en revisión y se espera contar con una nueva versión a comienzos o mediados de 2014.

Las guías de educación en IS y las acreditaciones de los programas impulsarán (y de alguna forma obligarán) a las Universidades a tomar estas guías como base para sus planes de estudio. En esta segunda década del siglo es probable que una mayor cantidad de Universidades publique los resultados obtenidos sobre su adopción y las adaptaciones que debieron realizar debido a su contexto particular.

Ford y Gibbs definen un modelo que permite caracterizar la madurez de una profesión en términos de 8 componentes y realizan una evaluación a 1996 del estado de la IS con respecto a esos componentes. Luego McConnel y Tripp realizan un análisis más reciente sobre el estado actual de dichos componentes para Estados Unidos y parte de Europa. Estos resultados muestran que aún la disciplina es inmadura en varios aspectos. Todavía tenemos que esperar que

la industria, y muchas veces la academia, sigan "modas"[35] en lo que refiere al desarrollo de software y que sigamos todavía con prácticas de *code-and-fix* como presentan McConnell y Tripp. De todas formas, esto no debe sentirse como algo dramático, todos los involucrados en la IS debemos comprender que es una disciplina reciente y, como presentamos en este capítulo, los avances en la misma han sido vertiginosos. Estos avances son los que nos deben dejar orgullosos y son los que nos indican que el futuro nos depara una profesión madura; aunque todavía deban pasar varios años para que esto se concrete.

Podemos suponer que los avances en esta profesión serán semejantes a los avances que se han dado en otras profesiones, pero probablemente de forma más acelerada debido a la propia aceleración de la historia[36]. Entre estos avances se tendrá que definir aún más la educación en IS y los diversos profesionales en torno a la IS. Si se piensa, por ejemplo, en la medicina, se puede ver la existencia de diferentes profesiones y que no todos son médicos: enfermeros, técnicos en radiografía y varios tipos de técnicos más, administradores de salud, médicos, especialistas médicos (cirujanos, traumatólogos, etc.), entre otros. ¿Tendremos en el futuro especialistas en IS? Ejemplos de estos especialistas podrían ser Arquitectos de Software, Responsables de Calidad de Software, etc. ¿Tendremos nuestros técnicos? Ejemplos de estos técnicos podrían ser programadores, diseñadores de interfaz de usuario y probadores (*testers*) de software. No todos tendrían por qué realizar una carrera de grado en IS, no todos tendrían por qué tener un título universitario. No podemos asegurar que esto vaya a acontecer exactamente así. Sin embargo, lo que es seguro es que el futuro, e incluso el futuro cercano, nos depara muchos cambios. Es importante que estemos preparados y para eso no hay nada mejor que conocer nuestra propia historia.

35 Por moda nos referimos al uso de técnicas, métodos, procesos de software, etc. por parte de la industria, sin que estas hayan sido efectivamente probadas o se conozca realmente los efectos de aplicarlas. Más explícitamente nos referimos al uso de algo únicamente porque "es usado por otros y dicen que da buenos resultados". Esto es lo que sucede actualmente en la industria de la IS. La ingeniería de software empírica tiene y va a tener un rol central en lograr cambiar esta situación. La educación en IS también tendrá ese rol.

36 La aceleración de la historia es un concepto tratado por varios autores del materialismo dialéctico (entre otros). Por ejemplo, Juan Grompone presenta claramente el concepto de aceleración de la historia, con datos empíricos, en su libro La Danza de Shiva - Tomo V: La Construcción del Futuro [40].

Bibliografía

[1] Gregory W. Hislop, "Software Engineering Education:Past, Present, and Future," in *Software Engineering Effective Teaching and Learning Approaches and Practices*., ch. I.

[2] IEEE-CS/ACM, *Joint Task Force on Computing Curricula. Software Engineering 2004: Curriculum guidelines for undergraduate degree in programs in software engineering*., 2004.

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

[4] The Institute of Electrical and Eletronics Engineers, IEEE Standard Glossary of Software Engineering Terminology, 1990.

[5] David Parnas, "Software Engineering - Missing in Action: A Personal Perspective," *IEEE Computer Society*, vol. 44, no. 10, pp. 54-58, 2011.

[6] Navid Ahmadi, Mehdi Jazayeri, Francesco Lelli, and Sasa Nesic, "A Survey of Social Software Engineering," *ASE Workshops*, pp. 1-12, 2008.

[7] Cleidson De Souza, Yvonne Dittrich, Helen Sharp, and Janice Singer, "Cooperative and Human Aspects of Software Engineering (CHASE 2009)," *ICSE'09*, 2009.

[8] IEEE Computer Society, *SWEBOK, Guide to the Software Engineering Body of Knowledge*, P. Bourque and R. Dupuis, Eds., 2004.

[9] Pierre Bourque et al., *Guide to the Software Engineering Body of Knowledge, A Straw Man version*.: IEEE Computer Society;UQAM, 1998.

[10] John White and Barbara Simons, "ACM's Position on Licensing of Software Engineers," *Communications of the ACM*, vol. 45, no. 11, pp. 91-92, Noviembre 2002.

[11] Pierre Bourque, Robert Dupuis, James W. Moore, Leonard Tripp, and Alan Abran, "Annual Status Report Guide to the Software Engineering Body of Knowledge," 2000.

[12] B. S Bloom, M. B Engelhart, E. J Furst, W. H. Hill, and D. R. Krathwohl, *Taxonomy of educational objectives. The classification of educational goals. Handbook 1: Cognitive domain*, B. S Bloom, Ed. New York: Longmans Green, 1956.

[13] Project Integrated Software & Systems Engineering Curriculum (iSSEc), *Curriculum Guidelines for Graduate Degree Programs in Software Engineering*, A. Pyster, Ed.: Stevens Institute of Technology, 2009.

[14] "Software Engineering Education in the Modern Age," in *Software Engineering Education in the Modern Age, Software Education and Training Sessions at the International Conference on Software Engineering, ICSE 2005, St. Louis, MO, USA, May 15-21, 2005,*

Revised Lectures, vol. 4309, St Louis, 2005, pp. 11-27.

[15] Alok Mishra and Ali Yazici, "An Assessment of the Software Engineering Curriculum in Turkish Universities: IEEE/ACM Guidelines Perspective," *Croatian Journal of Education*, vol. 13, pp. 188 -219, 2011.

[16] Stephen T. Frezza, Mei-Huei Tang, and Barry J. Brinkman, "Creating an Accreditable Software Engineering Bachelor's Program," *IEEE Software*, vol. 23, no. 6, pp. 27-35, Noviembre-Diciembre 2006.

[17] Sita Ramakrishnan, "Accreditation of Monash University Software Engineering (MUSE) Program," *International Journal of Issues in Informing Science and Information Technology*, vol. 4, pp. 73-89, 2007, http://proceedings.informingscience.org/InSITE2007/IISITv4p073-089Rama272.pdf.

[18] Eryu Ding et al., "Research and practice on software engineering undergraduate curriculum NJU-SEC2006," in *IEEE-CS Conference on Software Engineering Education and Training*, Honolulu, HI, 2011, pp. 492-496.

[19] Mark Ardis, David Budgen, Greg Hislop, Renee McCauley, and Mark Sebern, "Revisions to Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," in *Annual Conference of the American Society for Engineering Education*, San Antonio, TX, 2012.

[20] Mark Ardis et al., "Workshop on Revisions to SE 2004," in *CSEE&T*, 2013, pp. 356-358.

[21] The Joint Task Force on Computing Curricula Association for Computing Machinery and , *Computer Science Curricula 2013 - Ironman Draft*., 2013.

[22] M. Ardis et al., "Advancing Software Engineering Professional Education," *Software, IEEE*, vol. 28, no. 4, pp. 58-63, 2011.

[23] Mark A. Ardis and Gary A. Ford, "SEI Report on Graduate Software Engineering Education," in *CSEE*, 1989, pp. 208-249.

[24] Mark Ardis, Shawnr Bohne, Lucia Camilloni, Sylvia Ilieva, and Diego Vallespir, "Using GSwE2009 in the Creation and Modification of Graduate Software Engineering Programs and Related Curricula," in *Conference on Software Engineering Education and Training (CSEET 2013)*, 2013.

[25] George K. Thiruvathukal. (2013, Febrero) ComputingNow. [Online]. http://www.computer.org/portal/web/computingnow/archive/february2013?lf1=405398114d974216108976e6124795

[26] Ann E.K. Sobel, "Should Everyone Go to College?," *Computer in Science and Engineering*, pp. 82-83, 2012.

[27] Mark A. Ardis and Peter B. Henderson, "Software Engineering Education (SEEd) - Interview about the Academy for Software Engineering," *ACM SIGSOFT Software Engineering Notes*,

vol. 37, no. 3, pp. 8-9, 2012.

[28] Gary Ford and Norman Gibbs, "A Mature Profession of Software Engineering," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, CMU/SEI-96-TR-004, 1996.

[29] Steve McConnell and Leonard Tripp, "Professional Software Engineering-Fact or Fiction?," *IEEE Software*, vol. 16, no. 6, pp. 13-18, Noviembre 1999.

[30] Richard H. Thayer and Mark J. Christensen, Eds., *Software Engineering, Volume 1, The Development Process, 3rd Edition*.: Wiley-IEEE Computer Society Press, 2013.

[31] Steven McConnel and Leonard L. Tripp, "Software Engineering Professional Practices," in *Software Engineering Essentials, Volume II: The Supporting Processes*, Richard Hall Thayer and Merlin Dorfman, Eds., 2013, ch. 11, pp. 159-164.

[32] James McDonald and Mark J. Sebern, "Software Engineering Accreditation in the United States," in *Software Engineering Effective Teaching and Learning Approaches and Practice*., ch. XIII.

[33] Stephen, Tang,Mei-Huei Frezza and Barry J. Brinkman, "How to Create a Credible Software Engineering Bachelor's Program: Navigating the Waters of Program Development," in *Software Engineering Effective Teaching and Learning Approaches and Practice*., ch. XVI.

[34] Seidman and Stephen B., "An International Perspective on Professional Software Engineering Credentials," in *Software Engineering Effective Teaching and Learning Approaches and Practices*., ch. XVIII.

[35] John R. Speed, "What Do You Mean I Can't Call Myself a Software Engineer?," *IEEE Software*, pp. 45-5-, 1999.

[36] Phillip A. Laplante, "An International Perspective on U.S. Licensure of Software Engineers," *Technology and Society*, vol. 32, no. 1, pp. 28-30, 2013.

[37] Donald J. Bagert, "Licensing Software Engineers: The Computer Society Must Continue to Lead," in *COMPSAC*, 1999.

[38] J. Barrie Thompson, "A Long and Winding Road (Progress on the Road to a Software Engineering," in *COMPSAC*, 2001.

[39] Simon Rogerson, "The Software Engineering Code of Ethics and Professional Practice:a case for being proactive," in *26 th Annual International Computer Software and Applications Conference*, 2002.

[40] Juan Grompone, *La Danza de Shiva - Tomo 5: La Construcción del Futuro*. Montevideo: La Flor del Itapebí, 2001.