RACCIS, 2(2), 7-11, 2012.

Revista Antioqueña de las
Ciencias Computacionales y la Ingeniería de Software

ISSN: 2248-7441

www.fundacioniai.org/raccis
raccis@fundacioniai.org

A study at Masters Level Training in Software Engineering

Un Estudio al Nivel de Formación en Posgrados de Ingeniería de Software

George Clinton
Universidad de Alaska. *gclinton(AT)alaska.edu*

INFORMACIÓN DEL ARTÍCULO

Tipo de artículo: Investigación.

Historia del artículo
Recibido: 17-06-2012
Correcciones: 19-09-2012
Aceptado: 01-10-2012

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *Computer science education*.

General Terms
Software Engineering, Computer Science.

Keywords
Software Engineering, Computer Science, education, curriculum, level of training.

Palabras clave
Ingeniería de Software, plan de estudios, Ciencias Computacionales, nivel de formación.

ABSTRACT
Sponsored by the Department of Defense in the United States was formed an alliance among professionals from the academy, industry and government to design and structure a new model curriculum for Masters Programs in Software Engineering. Before starting this work was conducted at study to 28 of existing programs to determine the level of training in these masters in the country and the world. This article presents the results of that study.

RESUMEN
Con el patrocinio del Departamento de Defensa en los Estados Unidos se conformó una alianza entre profesionales de la academia, la industria y el Gobierno para diseñar y estructurar un nuevo modelo de plan de estudios para los programas de Maestría en Ingeniería de Software. Antes de empezar esta labor se realizó un estudio a 28 de los programas existentes para determinar el nivel de formación en estas maestrías en el país y en mundo. En este artículo se presentan los resultados de ese estudio.

1. INTRODUCCIÓN

A nivel mundial el software proporciona la mayor parte del valor en los nuevos productos y es la tecnología subyacente que hace posible los avances y el desarrollo de nuevas capacidades en la mayoría de las herramientas de la vida contemporánea. Los dispositivos médicos, los automóviles, las aeronaves, los sistemas de generación de energía y del medio ambiente, los teléfonos celulares y los elementos para el entretenimiento tienen una total dependencia de la funcionalidad basada en software, porque dirige gran parte de la complejidad de estos productos y sistemas. Debido a esa complejidad y a las dificultades inherentes al desarrollo de software la mayoría de las fallas y errores, que se detectan al integrar los sistemas y luego de la implementación del producto, se originan en la Ingeniería de Software para desarrollar estos productos.

La capacidad de una empresa o agencia gubernamental para gestionar y organizar sus proyectos depende en gran medida de sofisticados sistemas software, que apoyan los procesos y técnicas de negocio y que van desde sistemas de logística y de manufactura hasta de gestión de relaciones. Sin embargo, los reportes de la General Accountability Office [1], del Standish Group [2] y de otros organismos concluyen que crear y evaluar a tiempo el software a gran escala, dentro del presupuesto y con la funcionalidad esperada es poco común en esta industria.

La Ingeniería de Software es una disciplina reconocida por proponer y sustentar métodos y metodología para desarrollar software sin importar su escala, grado de confiabilidad y complejidad. Muchas universidades en el mundo ofrecen pregrados en esta ingeniería y la mayoría de ellas hacen parte del equipo que diseña el modelo de plan de estudios que publicaron ACM e IEEE en 2004 [3]. Otras tantas ofrecen maestrías en Ingeniería de Software. Sin embargo, sólo el Software Engineering Institute de Carnegie Mellon propuso e implementó en 1991 un plan de estudios modelo para la formación en este posgrado

[4]. Luego de estos años se requiere una nueva mirada y un replanteamiento a ese programa de estudios, el cual debe considerar la dependencia de la actual Sociedad de la Información y el Conocimiento de estos desarrollos tecnológicos.

El proyecto de currículo para integrar las ingenierías de Software y de Sistemas tiene como finalidad crear un modelo que refleje las exigencias actuales para construir software, la dependencia de la Ingeniería de Software de la de Sistemas y la influencia de los dominios de aplicación individuales, como las telecomunicaciones y los sistemas de defensa. Se espera que el plan de estudios resultante sea adecuado para una formación universitaria que conduzca a un grado de Maestría. Las actividades iniciales de este proyecto incluyeron la conformación inicial de un Early Start Team —EST—, limitado a entre 15 y 20 especialistas de la industria, el Gobierno, la academia y las asociaciones profesionales, y posteriormente el establecimiento de un Core Team más amplio y cubierto por los mismos cuatro sectores.

Antes de reunir el equipo y de sumergirse en el diseño del modelo de plan de estudios, se consideró necesario explorar el contexto sobre el cual se llevaría a cabo el proceso. Por lo tanto, el primer paso en este proyecto fue comprender la estructura y el contenido de los programas de maestría que actualmente están implementados. Más de 50 universidades en los Estados Unidos y muchas otras en el mundo ofrecen grados de maestría a nivel de Ingeniería de Software. De esa amplia población se recopiló información de 28 programas, que fue validada y analizada por profesores eruditos para lograr una descripción razonable del estado del arte.

2. METODOLOGÍA

El primer paso fue identificar las fuentes para la investigación, para lo cual se construyó una lista de instituciones y programas candidatos. El procedimiento incluyó búsqueda en la web, contactos directos y recomendaciones de los miembros del EST y se incluyó a universidades privadas y públicas y programas ofrecidos vía web. Casi todos los programas y universidades identificados y contactados accedieron a participar en la encuesta y además quedaron comprometidos en apoyar el valor del estudio para sus programas y para la comunidad en general. La muestra incluida en el estudio se detalla en la Tabla 1.

Luego se hizo evidente la necesidad de diseñar una taxonomía para estructurar el análisis de competencias cubiertas por el currículo de los programas. En lugar de crear un nuevo modelo de competencias de Ingeniería de Software, el equipo decidió utilizar el SWEBOK [5], debido a que es una taxonomía conocida, difundida, desarrollada de forma colaborativa y examinada ampliamente. Para simplificar la recopilación de datos sólo se utilizaron los tres primeros niveles de la misma, aunque para algunos programas se recogieron datos hasta el nivel cuatro. También se colectaron algunas materias impartidas que no eran visibles en el nivel tres.

Tabla 1 Programas incluidos en el estudio

Air Force Institute of Technology
Brandeis University
California State University, Fullerton
California State University, Sacramento
Carnegie Mellon University
Carnegie Mellon University West
DePaul University
Drexel University
Dublin City University, Ireland
Embry-Riddle Aeronautical University
George Mason University
James Madison University
Mercer University
Monmouth University
Naval Postgraduate School
Penn State University, Great Valley
Quebec University, Canada
Rochester Institute of Technology
Seattle University
Southern Methodist University
Stevens Institute of Technology
Texas Tech University
University of Alabama, Huntsville
University of Maryland University College
University of Michigan, Dearborn
University of Southern California
University of York, UK
Villanova University

Para la encuesta se desarrolló un instrumento con el objetivo de recoger y organizar los datos de los programas seleccionados, como datos del programa y los cursos y las competencias impartidas en cada uno de ellos. En la Tabla 2 se describe el instrumento aplicado.

Tabla 2 Datos solicitados

Información del programa
Universidad
Nivel de grado
Enfoque del programa
Requisitos para tesis y créditos
Cursos obligatorios
Cursos semi-obligatorios
(El estudiante puede tomar 50% o más de ellos)
Cursos electivos
Otros grados relacionados
Requisitos de Admisión
Información del curso
Número del curso
Título
Requisitos previos
Descripción
Objetivos y resultados del curso
Competencias
(Comparación con el nivel 3 de SWEBOK)
P: Enfoque primario del curso
C: Cubiertos en clase
< >: No Cubiertos

La información del programa y del curso se obtuvo de fuentes públicas —principalmente la web—. Con la información recogida el equipo realizó un mapeo inicial de los temas de los cursos. Una vez que se introdujeron

los datos se estableció contacto con los profesores eruditos para enviarle la información, y el análisis se realizó en conferencia con los miembros del equipo. Se complementaron los datos faltantes, se corrigieron los errores y se hicieron los cambios al instrumento solicitados por los profesores.

Aunque se intentó estandarizar la forma como se proporcionaron los datos, permanecieron algunas diferencias en el nivel de detalle y en la interpretación de las instrucciones por parte del personal del programa. Cuando la persona que responde la encuesta ha desarrollado o imparte un curso específico, la información entregada fue cubierta con un mayor nivel de granularidad, mientras que si la persona no tiene conocimiento de tal experiencia, la información tiende a ser más genérica. Esto originó ajustes a la forma como se realizó el análisis.

Para organizar los diferentes niveles de granularidad representados por los diferentes niveles de datos de SWEBOK se decidió analizarlos en el nivel 3 y reportarlos hasta el 1. Para ello fue necesario que el equipo agregara heurísticamente la granularidad desde los niveles más bajos a los más altos, donde los datos fueron proporcionados con granularidad más fina. La Tabla 3 muestra un ejemplo de cómo fueron agregados los niveles cuando era necesario.

Tabla 3. Nivel de agregación

P: Enfoque primario del curso; **c:** Cubierto en clase		**Curso 1**	**Curso 2**	**Curso 3**
Competencias en Software de SWEBOK		Obligatoria	Obligatoria	Semi-Obl.
1	**REQUISITOS DEL SOFTWARE**	c	P	
1.1	**Fundamentos de requisitos del software**			
1.1.1	Definición de requisitos del software	c	c	c
1.1.2	Requisitos del producto y del proceso		c	
1.1.3	Requisitos funcionales y no-funcionales	c	c	
1.1.4	Propiedades emergentes		c	
1.1.5	Requisitos cuantificables		c	
1.1.6	Requisitos del sistemas y del software	c		
1.2	**Procesos de los requisitos**	c		
1.2.1	Modelos de procesos		c	
1.2.2	Actores del proceso			
1.2.3	Soporte y gestión de procesos			
1.2.4	Calidad y mejoramiento de procesos			
1.3	**Elicitación de requisitos**	c		
1.3.1	Fuentes de los requisitos		c	
1.3.2	Técnicas de elicitación		c	
1.4	**Análisis de requisitos**	c		c
1.4.1	Clasificación de requisitos		c	
1.4.2	Modelado conceptual			
1.4.3	Diseño de arquitectura y distribución de requisitos			
1.4.4	Negociación de requisitos		c	
1.5	**Especificación de requisitos**			
1.5.1	Documento de definición del sistema			
1.5.2	Especificación de requisitos del sistema		c	
1.5.3	Especificación de requisitos del software		c	
1.6	**Validación de requisitos**			
1.6.1	Revisión de requisitos		c	
1.6.2	Prototipado			
1.6.3	Validación del modelo		c	
1.6.4	Pruebas de aceptación			
1.7	**Consideraciones prácticas**			
1.7.1	Naturaleza iterativa de los procesos de requisitos		c	
1.7.2	Gestión del cambio			
1.7.3	Atributos de los requisitos		c	
1.7.4	Trazabilidad de los requisitos		c	
1.7.5	Métricas de los requisitos			

3. HALLAZGOS

Los hallazgos del estudio se pueden clasificar en dos categorías generales: (1) características del programa y (2) características del currículo.

3.1 Características del Programa

El espectro de los programas investigados condujo a una serie de conclusiones acerca de cómo se manejaron, de los recursos de la facultad, del tamaño, de la longevidad y de las personalidades individuales. Algunos de los hallazgos más interesantes son:

- La Ingeniería de Software se concibe como una especialidad de las Ciencias Computacionales, del mismo modo como la Ingeniería de Sistemas a menudo se ha visto como una especialización de la

Ingeniería Industrial o de la Investigación de Operaciones. Los datos muestran que el 26% de los programas se encuentra en los departamentos de Ingeniería de Software, el 44% en los de Ciencias Computacionales y el resto en otras organizaciones académicas. Por ejemplo, en el Stevens Institute of Technology se encuentra adscrito a la School of Systems and Enterprises.

- El tamaño de las facultades generalmente es pequeño y con pocos profesores dedicados a la Ingeniería de Software. Los resultados muestran que el 48% de los programas tienen cinco o menos miembros dedicados a tiempo completo y adjuntos a la facultad. En los programas con muchos profesores, a menudo existe una fuerte dependencia de complementos para la formación. Se tiene la impresión que contar con profesores dedicados tan poco tiempo a la facultad hace que los programas sean más frágiles. Un pequeño cambio en la composición del cuerpo docente podría tener gran impacto en el contenido curricular. Un ejemplo de esta fragilidad se observó en uno de los pocos programas candidatos que no pudo participar en el estudio, en el que la falta de estudiantes lo había llevado a recortes de personal y a menos tiempo para actividades externas.

- Las matrículas estudiantiles generalmente son pocas en comparación con la Informática y otras disciplinas de la Ingeniería. Los datos muestran que un 29% de los programas tienen 25 estudiantes o menos y el 71% tienen 100 o menos.

- Muchos programas se centran en mercados específicos, a menudo impulsados por las empresas locales. Por ejemplo, Monmouth University abastece en gran medida la mano de obra local empleada por el Department of Defense, mientras que Carnegie Mellon University-West abastece la mano de obra local de los empresarios del Silicon Valley. Otros se orientan a los sistemas de defensa, sistemas embebidos en tiempo real, empresas emprendedoras de tecnología, Ingeniería de Software cuantitativa, economía del software, sistemas críticos para la seguridad, Ingeniería de Software seguro y a los sistemas software altamente confiables.

- Los requisitos de admisión varían ampliamente. Algunos aceptan personas con cualquier pregrado y un promedio regular, mientras que otros requieren un grado en Ciencias Computacionales y al menos dos años de experiencia. Los cursos de preparación o nivelación son considerablemente utilizados para apoyar a los estudiantes que no puedan cumplir con todos los requisitos. Muchos programas realizan exenciones de requisitos académicos para los estudiantes con amplia experiencia industrial.

- Los objetivos de egreso del programa son muy diversos. Los programas están configurados para producir graduados de acuerdo con las necesidades percibidas y los deseos de salida de la población estudiantil. Algunos se centran en potencializar habilidades de software y otros en las habilidades y conocimientos necesarios para gestionar proyectos complejos. En algunos casos los graduados están preparados para ser ingenieros líderes y ejecutivos de software.

- Los programas continúan funcionando a pesar de la preocupación generalizada porque en la última década ha disminuido el interés de los graduados por estudiar Ciencias Computacionales. De los 28 programas del estudio ocho se crearon a partir de 2005.

- Las ofertas On-line son populares y existen muchos programas que llegan a los estudiantes lejos de sus sedes físicas y algunos que citan un alcance global.

3.2 Características del Currículo

La estructura y contenido de los cursos de los programas existentes y su relación con estándares como SWEBOK puede proporcionar información valiosa para el desarrollo del currículo modelo. Aunque se recogieron datos sobre todos los cursos de los programas en Ingeniería de Software, el análisis inicial se limitó a cursos obligatorios o semi-obligatorios; éste último es el que un estudiante tiene por lo menos un 50% de probabilidades de tomar. Algunos de los hallazgos son:

- Menos del 40% de los programas requiere de un curso de introducción a la Ingeniería de Software. Varios profesores comentaron que su programa estaba dirigido a estudiantes con experiencia laboral en desarrollo de software. Los estudiantes que necesitan un curso introductorio o bien no se aceptan en el programa o se espera que tomen cursos de grado para prepararlos para la Maestría.

- En los cursos obligatorios y semi-obligatorios existe una amplia variación en la profundidad y amplitud de cobertura de SWEBOK. Las áreas mejor cubiertas son cursos en diseño y gestión de requisitos, mientras que las menos cubiertas son cursos en mantenimiento, en gestión de configuración, en calidad y en herramientas y métodos.

- Es evidente que SWEBOK no representa completamente los cursos obligatorios de muchos programas. Mientras se esperaba encontrar muchos cursos electivos en áreas que esta propuesta sólo las tiene de pasada, o incluso temas no incluidos en ella, sorprendió el hecho de que estos cursos son obligatorios en muchos programas. Algunos se centran en lenguajes específicos de programación, como C++, Java y C#, en la economía de software, en los factores humanos del diseño de interfaz y en las cuestiones legales y éticas de desarrollo de software.

- Una de las metas de este proyecto es examinar y abordar el grado en que la Ingeniería de Sistemas se debe integrar en un plan de estudios de posgrado. No es sorprendente que pocos de los programas estudiados aborden explícitamente la Ingeniería de Sistemas en sus cursos obligatorios y semi-obligatorios.

- El paradigma Orientado por Objetos es el estándar de desarrollo. Muy pocos trabajan con métodos estructurados en Ingeniería de Software sólo por interés histórico. Esto tiene consecuencias interesantes porque la mayoría de los programas de Ingeniería de Sistemas todavía trabajan estos métodos. Esta diferencia puede dar lugar a un modelo de enfrentamientos, por ejemplo, cuando se trata de integrar una arquitectura de sistemas desarrollada utilizando métodos estructurados con una arquitectura de software desarrollada utilizando métodos Orientados por Objetos.

- La flexibilidad de los cursos es muy variada. Por ejemplo, una universidad ofrece no-electivas como cursos obligatorios. En promedio, los estudiantes toman 11.6 cursos para el grado, 8.3 de los cuales son obligatorios o semi-obligatorios.

- Frecuentemente se requiere realizar prácticas profesionales o proyectos. Aunque la mayoría de los programas ofrece una opción de tesis, generalmente los estudiantes prefieren una de estas opciones.

4. CONCLUSIONES Y TRABAJO FUTURO

El análisis de los datos recogidos en este estudio continúa. El trabajo inicial permitió elaborar un perfil razonable de los programas de maestría en Ingeniería de Software que se ofrecen actualmente. Hay, sin embargo, muchos aspectos por completar. Como siempre, al analizar los datos emerge el valor de la información no recopilada. Algunas de las áreas identificadas para futuros estudios son:

- Analizar las asignaturas electivas.
- Diferenciar entre los diversos tipos de universidades: privadas vs públicas, grandes vs pequeñas, especializadas vs generales.
- Recolectar datos adicionales de las universidades extranjeras.
- Cuestiones de desarrollo de programas:
  - ¿Por qué tienen Maestrías en Ingeniería de Sistemas vs Maestría en Ciencias Computacionales con especialización en Ingeniería de Software?
  - ¿Cómo se decide el número de cursos obligatorios?
  - ¿Cómo diseñan su plan de estudios?
- Cuestiones de evolución de los programas
  - ¿Qué tanto ha cambiado el contenido desde el inicio del programa?
  - ¿Cómo han sido los cambios demográficos de los estudiantes en el tiempo?

El equipo de trabajo tiene la intención de hacer próximamente un estudio de seguimiento en el que se abordarán estas cuestiones y se establecerá un punto de referencia contra el cual se pueda medir el impacto del plan de estudios modelo. Se piensa llevar a cabo la medición a través de re-encuestas a la comunidad en un ciclo de dos años en los próximos 6 a 10 años.

5. REFERENCIAS

[1] GAO. 2006. Defense Acquisitions: Assessment of Selected Major Weapons Programs. United States Government Accountability Office, Report to Congressional Committees. GAO-06-391, April.

[2] The Standish Group International. 2003. The CHAOS Chronicles.

[3] LeBlanc, R. et al. 2004. Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. ACM and IEEE Computer Society.

[4] Ford, G. 1991. SEI Report on Graduate Software Engineering Education. Technical Report, CMU/SEI-91-TR-002, ESD-TR-91-002. Software Engineering Institute.

[5] Abran, A. et al. 2004. Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Computer Society.