Sugerencias para la Inclusión de Temas de Medición de Software en un Currículo de Ingeniería de Software para Estudiantes de Pregrado

Mónica Villavicencio
Facultad de Ingeniería en Electricidad y Computación
Escuela Superior Politécnica del Litoral (ESPOL)
Guayaquil, Ecuador
mvillavi@espol.edu.ec

Alain Abran
Deptartamento de Ingeniería de Software y TI
École de technologie supérieure
Montreal, Canadá
alain.abran@etsmtl.ca

***Resumen*—En la última década, se han realizado esfuerzos para desarrollar una versión actualizada del Cuerpo de Conocimiento de la Ingeniería de Software (SWEBOK por sus siglas en inglés). Aunque esta actualización representa una gran contribución para este campo, aún existe la necesidad de actualizar las guías curriculares de ingeniería de software. Tomando como base investigaciones previas, el presente estudio aborda la necesidad de proporcionar una serie de sugerencias respecto a temas de mediciones de software que deben enfatizarse en programas universitarios. Para ello, se efectuó una revisión de los cuerpos de conocimiento relacionados a mediciones de software, así como de las guías curriculares de ingeniería de software a nivel de pregrado. Todo esto acompañado de un análisis de opiniones de profesionales de ingeniería de software en ejercicio de sus funciones, así como de profesores universitarios que imparten dicha cátedra. Los resultados indican que cinco temas de medición de software son esenciales en un currículo de ingeniería de software.**

***Palabras claves*— Guías curriculares; ingeniería de software; educación superior; mediciones de software.**

I. Introduccion

El SWEBOK (Software Engineering Body of Knowledge) es una guía que enmarca los contenidos de la disciplina de la Ingeniería de Software. Uno de los objetivos de SWEBOK es proporcionar las bases para el desarrollo de currículos de estudio [1]. La versión original de la guía SWEBOK (año 2004) fue la piedra angular para generar el Conocimiento Educativo de la Ingeniería de Software (Software Engineering Education Knowledge - SEEK), el mismo que fuera usado para desarrollar las guías curriculares para programas universitarios de ingeniería de software (SE2004) [2]. Estas guías proporcionan sugerencias a los educadores sobre el contenido de los programas, la forma de enseñar los contenidos en contextos diferentes, y las habilidades (técnicas y de gestión) que necesitan los estudiantes de pregrado para afrontar los procesos de desarrollo de software. De acuerdo a estos lineamientos curriculares, una de las siete características deseables en el trabajo de los ingenieros de software es: "Los ingenieros miden cosas, y cuando es apropiado, trabajan cuantitativamente; calibrando y validando sus mediciones; y usando aproximaciones basadas en la experiencia y datos empíricos" [2]. Esta característica conecta la definición de ingeniería de software al perfil esperado de un ingeniero en software. La ingeniería de software se define como "la aplicación de un enfoque sistemático, disciplinado, cuantificable al desarrollo, operación, y mantenimiento de software; esto es, la aplicación de ingeniería al software" [1]. La característica antes mencionada es también evidente dentro de las directrices curriculares SE2004 (desarrollados en conjunto entre ACM & IEEE), en donde existe una presencia extensa de temas de medición de software. SE2004 permanece -hasta la presente fecha- como un referente para el desarrollo curricular.

En la última década, se han realizado esfuerzos para desarrollar una versión actualizada de la guía SWEBOK, publicada en el año 2014 como SWEBOK v3.0. Con respecto a la versión inicial SWEBOK 2004, la versión 2014 extiende de 10 a 15 las áreas de conocimiento (KA – Knowledge Areas); áreas que contienen temas relacionados a la medición del software. Es importante anotar que en el año 2010, un área de conocimiento completa de mediciones de software fue desarrollada por Abran et al [3], obteniendo como resultado un Cuerpo de Conocimiento de Mediciones de Software (Software Measurement Body of Knowledge). Este nuevo cuerpo de conocimiento – incluido en la enciclopedia de ingeniería de software – integra y expande el conocimiento relacionado a mediciones de software contenido a lo largo de SWEBOK 2004.

Se espera que la reciente versión del SWEBOK (v3.0) contribuya a la actualización de las directrices curriculares de ingeniería de software. Dentro de este contexto, este artículo presenta sugerencias para incluir temas esenciales de medición de software en programas universitarios a nivel de pregrado.

Este artículo está organizado de la siguiente manera: la sección II presenta un mapeo de temas relacionados a la medición de software entre las directrices curriculares (SE2004) y los Cuerpos de Conocimiento afines. La sección III resume las opiniones de profesores universitarios y profesionales de ingeniería de software en ejercicio acerca de los temas de mediciones de software que deben enseñarse en programas universitarios. La sección IV presenta sugerencias para actualizar las directrices curriculares de ingeniería de software respecto a los temas de medición de software. Finalmente, la sección V concluye el artículo.

II. Mapeo de temas de medición de software

Para el mapeo de los temas de medición de software, usamos tres documentos:

1. Las directrices curriculares para programas universitarios en ingeniería de software SE2004 [2];
2. SWEBOK versión 3.0 [1]; y

3. El cuerpo de conocimiento de mediciones de software [3].

El capítulo 4 del SE2004 presenta el SEEK (Software Engineering Education Knolwedge), el mismo que propone el conocimiento de ingeniería de software que los estudiantes de pregrado deberían adquirir. El SEEK está organizado en tres niveles jerárquicos: áreas de conocimiento; unidades de conocimiento; y temas (ver un ejemplo en la Tabla I). SEEK considera diez áreas de conocimiento; 1) Fundamentos de computación; 2) Fundamentos de Matemáticas e Ingeniería; 3) Práctica Profesional; 4) Modelamiento y Análisis de Software; 5) Diseño de Software ; 6) Verificación y Validación de Software; 7) Evolución de Software; 8) Procesos de Software; 9) Calidad de Software; y 10) Gestión del Software.

Para cada tema, el SEEK sugiere:

- Niveles de aprendizaje (**N**) de acuerdo a la versión original de la Taxonomía de Bloom [4], y
- Relevancia (**R**) de los temas (esencial **E**, deseable **D**, y opcional **O**)

La versión original de la Taxonomía de Bloom tiene seis niveles de aprendizaje, denominados como: conocimiento, comprensión, aplicación, análisis, síntesis y evaluación. De éstos, los tres primeros niveles fueron usados en SEEK (conocimiento **K**, comprensión **C**, y aplicación **A**). Un ejemplo de la jerarquía de SEEK se muestra en la Tabla I donde los temas Mediciones y Métricas tienen el nivel de aprendizaje **K** (conocimiento) y relevancia **E** (esencial).

TABLA I. EJEMPLO DE JERARQUIA EN EL SEEK

Areas de conocimiento SEEK	Unidades de conocimiento	Tópicos	N	R
Fundamentos de matemáticas e ingeniería	Fundamentos de Ingeniería para el Software	Mediciones y métricas	K	E
		Teoría de mediciones	C	E

Después de revisar las directrices curriculares, se identificó que seis áreas de conocimiento del SEEK incluyen temas de mediciones de software (ver las dos primeras columnas de la Tabla II), todas estas áreas son consideradas esenciales (**E**) en el currículo 2004.

Habiendo identificado estas seis áreas, buscamos información complementaria en otros Cuerpos de Conocimiento (SWEBOK y el cuerpo de conocimiento de mediciones de software). En el caso de SWEBOK 2004, éste incluye 10 áreas de conocimiento en las que se incluyen temas relacionados a medición de software presentados en forma dispersa a lo largo del contenido de ellas. Esta dispersión motivó la creación de un nuevo cuerpo de conocimiento de Mediciones de Software en el 2010, con el objetivo de agrupar todos los tópicos correspondientes [3]. El cuerpo de conocimiento de Mediciones de Software divide el conocimiento en seis grandes tópicos: 1) Conceptos Básicos; 2) Procesos de Medición; 3) Mediciones por fase del ciclo de vida del software (SLC); 4) Técnicas y herramientas; 5) Datos cuantitativos; y 6) Estándares de Mediciones. Cada tópico es dividido en sub-tópicos. Por ejemplo, el tópico Conceptos Básicos está dividido en tres sub-tópicos: fundamentos; conceptos y definiciones; y modelos de medición de software.

La versión actual de SWEBOK (v3.0) considera 15 áreas de conocimiento (KAs) divididas en tópicos y sub-tópicos: 11 de las 15 KAs contienen temas de medición de software. Por ejemplo, el área de conocimiento de Gestión de Ingeniería de Software tiene el tópico Mediciones en Ingeniería de Software.

La Tabla II muestran el mapeo entre SEEK (incluido en SE2004), el cuerpo de conocimiento de Mediciones de Software, y SWEBOK v3.0. El mapeo fue efectuado en base a las directrices curriculares para ver si estas están alineadas con dichos cuerpos de conocimiento. Por lo tanto, buscamos los temas de medición de software contenidos en SEEK dentro de los otros cuerpos de conocimiento. Al hacer esto, observamos que el tema Métricas de Diseño incluidos en el SEEK no está considerado ni en SWEBOK v3.0 ni en el cuerpo de conocimiento de Mediciones de Software; Además, observamos que el tema Procesos de Software en equipo está incluido marginalmente en SWEBOK v3.0 y no está incluido en el cuerpo de conocimiento de Mediciones de Software (Ver Tabla II- adaptadas de [5]).

El mapeo en la dirección opuesta, es decir, desde el SWEBOK v3.0 hacia las directrices curriculares, se muestra en la Sección IV donde se presentan sugerencias para un nuevo currículo- considerando sólo el análisis de los temas de medición de software (ver Tabla V).

III. PRIORIDADES: OPINIONES DE ACADÉMICOS Y PROFESIONALES EN EJERCICIO

Las opiniones de profesores y profesionales acerca de los temas de mediciones de software que deberían ser priorizados en las universidades se recopilaron en dos fases: 1) una encuesta vía web a los profesionales en ejercicio y 2) un estudio Delphi con dos paneles de expertos (profesionales en ejercicio y profesores universitarios).

A. Encuesta vía web

La encuesta se realizó a profesionales en ejercicio – específicamente profesionales de organizaciones privadas o públicas que trabajan en mejoramiento de procesos de software o como especialistas en mediciones de software. La encuesta buscaba identificar: 1) el nivel de importancia - percibido por los profesionales- respecto a las mediciones de software; 2) la apreciación de las organizaciones sobre el conocimiento de mediciones de software que poseen los estudiantes recién graduados cuando se convierten en sus empleados; y 3) los tópicos específicos de medición de software que deberían enfatizarse en los programas de ingeniería de software desde el punto de vista del profesional en ejercicio. Para este propósito, se diseñó un cuestionario con una lista de 12 tópicos de medición de software, los cuales fueron tomados del Cuerpo de Conocimiento de Mediciones de software [3] y la guía SWEBOK capítulo 12 (Borrador de Febrero 4 de 2008) [6]. En ese momento, el capítulo 12 era el Área de Conocimiento de Mediciones de Software en SWEBOK. Cincuenta y dos profesionales de 18 países contestaron el cuestionario durante la primavera de 2011. Detalles de la encuesta se encuentran disponibles en [7].

De esta encuesta, identificamos los tópicos que necesitaban ser enfatizados en los cursos universitarios.

Para el análisis de resultados, clasificamos a los encuestados en profesionales en ejercicio de instituciones certificadas y no certificadas. La tabla III muestra 11 temas de medición de software de acuerdo al orden de preferencia seleccionado por los participantes.

TABLA II. MAPEO ENTRE CUERPOS DEL CONOCIMIENTO

Software Engineering Education Knowledge (SEEK) y SE2004		**Cuerpo de Conocimiento de Mediciones de Software**		**SWEBOK v3.0**	
Areas de Conocimiento	**Tópicos**	**Tópicos**	**Sub Tópicos**	**Areas de Conocimiento**	**Tópicos**
Fundamentos de Matemáticas e Ingeniería	Métricas y mediciones	Conceptos Básicos	Fundamentos	Fundamentos de Ingeniería para el Software	Mediciones
	Teoría de mediciones		Definiciones y conceptos		
Diseño de Software	Mediciones de atributos del diseño (ej. Acoplamiento, cohesión,etc.)	Mediciones en el ciclo de vida (SLC)	Diseño de Software	Diseño de Software	Análisis y evaluación de la calidad del diseño
	Métricas de diseño (ej. Factores arquitectónicos, interpretación, etc.)	*No encontrado*		*No encontrado*	
Verificación y Validación del Software	Métricas y mediciones (ej. Confiabilidad, usabilidad, etc.)	Mediciones en el ciclo de vida (SLC)	Pruebas de Software	Pruebas de Software	Mediciones relacionadas a las pruebas
	Análisis de reporte de fallas				
	Análisis de defectos				
Procesos de software	Análisis y control de la calidad (ej. Prevención de defectos, métricas de calidad, análisis de causa raíz, etc.)	Técnicas y herramientas	Técnicas de medición	Proceso de ingeniería de software	Mediciones de software
	Proceso de software individual (modelo, definición, mediciones, análisis, mejora)		Técnicas de medición		
	Procesos de software en equipo (modelo, definición, organización, mediciones, análisis, mejora)	*No encontrado*	*No encontrado*		
Calidad del Software	Modelos and métricas de calidad del Software	Mediciones en el ciclo de vida (SLC)	Calidad del Software	Calidad del Software	Consideraciones prácticas
	Métricas y mediciones de la calidad del producto				
Administración del Software	Estimación del esfuerzo	Mediciones en el ciclo de vida (SLC)	Mediciones de software para constucción y pruebas	Gestión de la Ingeniería de Software	Planificación de proyectos de software
		Técnicas y herramientas	Herramientas para mediciones		
	Medición y análisis de resultados	El proceso de medición	Realizar y evaluar el proceso de medición		Mediciones en ingeniería de software

Uno de los tópicos, “Medidas de mantenimiento de software”, no aparece en la tabla III ya que no fue seleccionado por ninguno de los participantes.

Al observar las cinco primeras posiciones de la tabla, podemos notar similitudes y diferencias. Por ejemplo, los profesionales de ambos tipos de organizaciones concordaron en

seleccionar "Conceptos Básicos" como el tema más importante a cubrirse en los cursos universitarios. Adicionalmente, otros dos temas están listados dentro de los cinco primeros: "Técnicas y herramientas" y "El proceso de medición." Respecto a las diferencias, puede observarse que los profesionales que trabajan en organizaciones no certificadas ubicaron el tema "Mediciones para la Gestión de Ingeniería de Software" en segundo lugar de importancia, mientras que los de organizaciones certificadas no lo consideraron dentro de los cinco primeros. Igualmente, los profesionales provenientes de organizaciones certificadas dieron una importancia considerable a los temas "Estándares de Medición" y "Mediciones para la fase de requerimientos" mientras que aquellos de organizaciones no certificadas no lo hicieron.

TABLA III. TEMAS QUE NECESITAN MAYOR ÉNFASIS DE ACUERDO A LA OPINIÓN DE PROFESIONALES

	Temas seleccionados por profesionales en ejercicio	
#	**Organizaciones No-Certificadas**	**Organizaciones Certificadas**
1	Conceptos Básicos	Conceptos Básicos
2	Medición en la Gestión de Ingeniería de software	El proceso de medición
3	Técnicas y herramientas	Estándares de Medición
4	Mediciones relacionadas a la calidad	Mediciones para fase de requerimientos
5	El proceso de medición	Técnicas y herramientas
6	Estándares de Medición	Mediciones relacionadas a la calidad
7	Mediciones para fase de requerimientos	Mediciones para pruebas
8	Repositorios para datos cuantitativos	Medición en la Gestión de Ingeniería de software
9	Mediciones para la fase de construcción	Mediciones para la fase de diseño
10	Mediciones para la fase de diseño	Repositorios para datos cuantitativos
11	Mediciones para pruebas	Mediciones para la fase de construcción

B. Estudio Delphi

Los estudios Delphi son comúnmente usados en investigación educativa para diseño y perfeccionamiento curricular [8-12]. Estos estudios siguen un proceso de comunicación estructurado en varias iteraciones (rondas) para consolidar las opiniones de expertos acerca de un sujeto de estudio que está siendo investigado [8-17].

Nuestro Delphi fue diseñado para llegar a un consenso entre profesores universitarios y profesionales en ejercicio – con probada experiencia en el área de ingeniería de software - respecto a temas de mediciones de software que deben enseñarse a estudiantes de grado. En el estudio también se preguntó a los expertos acerca de los niveles de aprendizaje que los estudiantes deberían alcanzar de acuerdo a la taxonomía de Bloom.

El estudio Delphi tuvo dos paneles de expertos en mediciones de software: profesores universitarios y profesionales. Los criterios usados para invitar a los participantes de los paneles fue el siguiente:

Profesionales

- Demostrar cinco o más años de experiencia profesional en mediciones de software, trabajando en programas de mejora de procesos de software y/o en programas de mediciones de software como un miembro del equipo o especialista.
- Participar como miembro de un comité o asociación de mediciones de software en su país de origen o fuera de él (no obligatorio, pero preferible).
- Haber publicado artículos sobre mediciones de software relacionados con su experiencia profesional.
- Demostrar educación post-secundaria.

Profesores universitarios

- Demostrar cinco o más años de experiencia como docente en cursos especializados de mediciones de software, o cursos relacionados de ingeniería de software en los que se enseñen temas de medición de software.
- Haber publicado artículos sobre mediciones de software relacionados a sus áreas de interés de investigación.
- Participar como miembro de un comité o asociación de mediciones de software en su país de origen o fuera de él (no obligatorio, pero preferible).

Tomando en consideración estos criterios, se elaboró una lista de posibles participantes a través de una búsqueda de artículos relacionados con mediciones de software en bibliotecas digitales como IEEE Xplore, Engineering Village, entre otras. Adicionalmente, se realizó una búsqueda de participantes en grupos especializados en mediciones de software existentes en LinkedIn, por ejemplo: Foro de Análisis y Medición; Mediciones de Software y Puntos de Función; Grupo de Usuarios del método COSMIC; entre otros.

La metodología seguida para llevar a cabo este estudio Delphi tomó como referencia los trabajos realizados por Okoli and Pawlowski (2004) [15] y Skulmoski *et al*. (2007) [17]. La Figura 1 muestra las mejoras realizadas a dicha metodología, entre estas: la separación de las actividades en tres grandes fases (preparación, ejecución y verificación); la inclusión de los procedimientos de aprobación por parte del comité de ética de la institución a cargo de llevar a cabo la investigación o entes reguladores de gobierno; y la inclusión de actividades de verificación de resultados. Como se puede apreciar en la figura, este estudio Delphi fue ejecutado en tres rondas. En cada ronda, una aplicación web estuvo disponible para recoger información de los participantes.

En la primera ronda, los expertos seleccionaron -de una lista de 13 tópicos de mediciones de software- cinco tópicos que (según su mejor criterio) deberían enseñarse en programas universitarios. La lista tomó en consideración 13 tópicos de medición incluidos en el cuerpo de conocimiento de mediciones de software [3] (los mismos tópicos que aparecen en la Tabla III más "Mediciones para el mantenimiento de software" y "Mediciones para la gestión de configuración de software"). Vale la pena mencionar que los tópicos fueron presentados en un orden aleatorio en cada una de las rondas.

Por cada tópico, los expertos debían identificar los niveles de aprendizaje que los estudiantes de grado debían alcanzar en la universidad, escogiendo de entre una lista de enunciados. Por ejemplo, el tópico "Conceptos básicos de mediciones de software" contenía cinco enunciados de resultados de aprendizaje: Recordar terminología y conceptos; Dar ejemplos de conceptos básicos, métodos y procedimientos de medición; Explicar conceptos básicos; Usar terminología y conceptos en

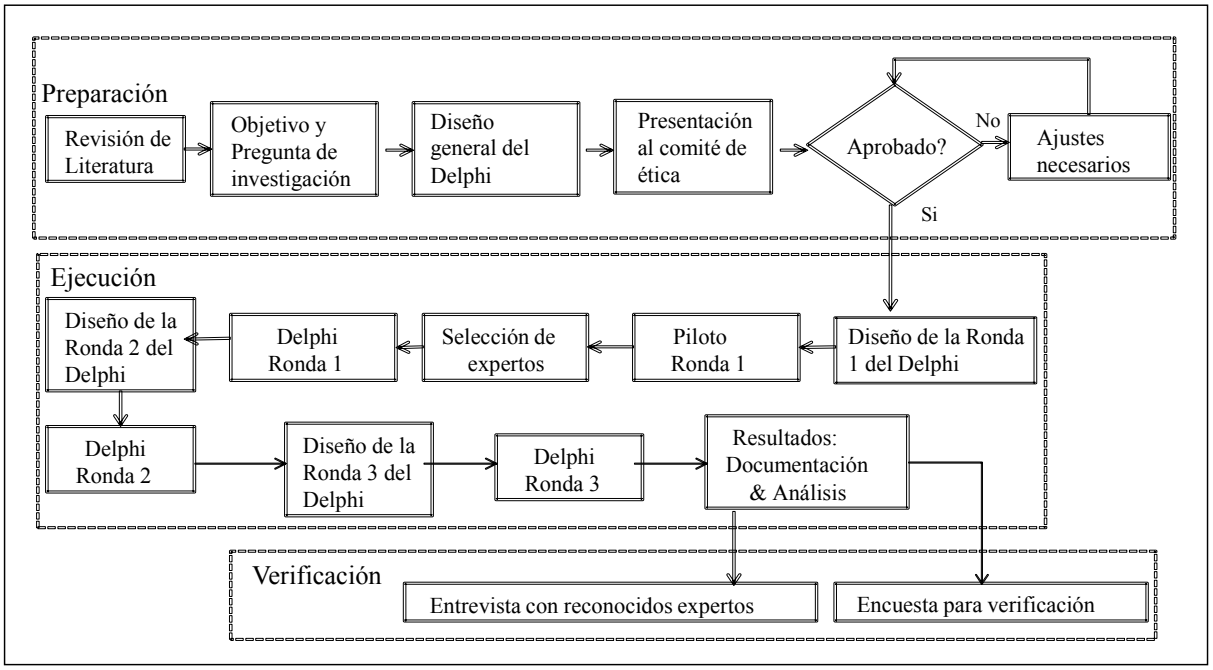

Fig. 1. Vista General del estudio Delphi – adaptado de [15] y [17].

Un ejercicio o proyecto propuesto; y Diseñar/modificar procedimientos de medición. Para evitar sesgos, los participantes del estudio no conocían la relación entre los enunciados mostrados en la aplicación web y los seis niveles de aprendizaje de la taxonomía de Bloom –versión revisada por Anderson *et al* 2001. Los seis niveles de aprendizaje de esta versión de la taxonomía son: Recordar (**R**), Comprender (**C**), Aplicar (**A**), Analizar (**Z**), Evaluar (**V**) y Crear (**E**) [18].

Con los datos recopilados de cada panel (profesionales y profesores universitarios), se utilizaron los siguientes criterios para seleccionar los cinco tópicos de mediciones de software más importantes (que deberían ser incluidos en el currículo de programas de ingeniería de software):

- Más del 50% de los participantes (profesores universitarios y profesionales) en ambos paneles eligió el tópico.
- Más del 50% de los participantes (profesores universitarios o profesionales) de un panel escogió el tópico.
- El tópico no alcanzó el nivel de aceptación del 50%, pero aun así fue calificado como uno de los 5 tópicos más importantes de ambos paneles.

Los participantes de los dos paneles estuvieron de acuerdo en seleccionar cuatro tópicos como relevantes. Sin embargo, el quinto tópico seleccionado en cada panel fue diferente, dando un total de seis tópicos que cumplían alguno de los criterios expuestos anteriores. Dichos tópicos son:

1. Conceptos básicos de mediciones de software (ambos paneles)
2. El proceso de medición (ambos paneles)
3. Técnicas y herramientas para mediciones de software (ambos paneles)
4. Mediciones para la gestión de ingeniería de software (ambos paneles)
5. Mediciones para la fase de requerimientos (panel de profesionales)
6. Mediciones para la fase de diseño (panel de profesores)

Adicionalmente, en la primera ronda, los participantes seleccionaron los niveles de aprendizaje que - según ellos - los estudiantes de pregrado deberían de alcanzar para los cinco tópicos de medición de software considerados más importantes. Una aplicación web se desarrolló de tal manera que una vez que los participantes habían seleccionado los cinco tópicos más importantes, sus correspondientes niveles de aprendizaje aparecieran en la pantalla. Cada tópico tenía entre 4 y 6 enunciados que representaban los niveles de aprendizaje basados en la taxonomía de Bloom. Esto significa que cada nivel esperado de aprendizaje por tópico estaba asociado a un nivel de la taxonomía de Bloom. Los participantes no sabían la relación entre los niveles de aprendizaje que se mostraban en la pantalla con la taxonomía de Bloom. En el análisis de datos, se hizo uso de la relación para conocer qué niveles de aprendizaje de acuerdo a la taxonomía de Bloom fueron seleccionados por los participantes.

Una vez que los participantes seleccionaron los niveles de aprendizaje por tópico (los cinco más importantes), utilizamos los siguientes criterios para identificar la preferencia de los mismos:

- Más del 50% de los participantes en ambos paneles eligió el nivel de aprendizaje.
- Más del 50% de los participantes de un panel eligió el nivel de aprendizaje.
- El nivel de aprendizaje no alcanzó más del 50% de aceptación, sin embargo, fue considerado como el más importante en ambos paneles.

Debido a la extensión de las tablas de niveles de aprendizaje, sólo se muestran en este artículo los resultados definitivos (ver Tabla IV – Niveles de aprendizaje por tópico).

En la segunda ronda, se pidió a los participantes ordenar los seis tópicos resultantes de la ronda 1 (los cuales fueron presentados de forma aleatoria en la aplicación web) de acuerdo a su importancia y que, además, dieran una explicación para justificar su ranking.

En la ronda 3 se presentaron los resultados del ranking obtenido y se les preguntó si estaban de acuerdo o no con la

lista de tópicos de mediciones de software considerados como prioritarios, posición que debían de argumentar obligatoriamente. Si no estaban de acuerdo, los participantes podían indicar su nuevo ranking.

Los criterios utilizados para determinar el ranking de los tópicos en las rondas 2 y 3 fue el siguiente:

- La moda (la posición – del 1 al 6 - más comúnmente seleccionada por los participantes para un tópico específico)
- El número de votos obtenidos por cada tópico.

Con estos dos criterios, definimos un grado de consenso (GC) entre los participantes, como se explica a continuación:

- Menos del 10% de los votos (0-9,99%): Muy débil grado de consenso (MD)
- Menos del 30% de los votos (10 a 29,99%): Débil grado de consenso (D)
- Menos del 50% de los votos (30 a 49,99%): Moderado grado de consenso (M)
- Menos del 70% de los votos (50 a 69,99%): Fuerte grado de consenso (F)
- Menos del 90% de los votos (70 a 89,99%): Muy fuerte grado de consenso (MF)
- Menor o igual al 100% de los votos (90-100%): Extremadamente fuerte grado de consenso (EF)
- Con una moda indeterminada: No hay consenso

En la ronda 2 del panel de los profesores, logramos identificar claramente el orden de las cuatro primeras posiciones del ranking de tópicos de mediciones de software. No obstante, las posiciones 5 y 6 no estaban claras. Por tanto, se requirió una tercera ronda para confirmar las cuatro primeras posiciones e identificar las posiciones 5 y 6.

Por otro lado, en la ronda 2 realizada con el panel de profesionales, identificamos las posiciones 1 y del 4 al 6. Es decir, que las posiciones 2 y 3 tuvieron un bajo nivel de consenso. La tercera ronda nos permitió identificar estas posiciones.

Con respecto a los niveles de aprendizaje por tópico, en la segunda ronda se evidenció un claro consenso entre los participantes, alcanzo grados de consenso considerados como fuerte, muy fuerte, extremadamente fuerte. Por tal razón, los niveles de aprendizaje por tópico no fueron incluidos en la ronda 3.

Los criterios utilizados para determinar las preferencias de los participantes con respecto a los niveles de aprendizaje fueron:

- Menos del 10% de los votos (0-9,99%): preferencia muy débil (MD)
- Menos del 30% de los votos (10 a 29,99%): preferencia débil (D)
- Menos del 50% de los votos (30 a 49,99%): preferencia moderada (M)
- Menos del 70% de los votos (50 a 69,99%): preferencia fuerte (F)
- Menos del 90% de los votos (70 a 89,99%): preferencia muy fuerte (MF)
- Menor o igual a 100% de los votos (90-100%): preferencia extremadamente fuerte (EF)

La tabla IV muestra los niveles de aprendizaje por tópico que fueron seleccionados por los participantes. Como se puede notar, los tres primeros niveles correspondientes a la Taxonomía de Bloom son los más preferidos (R, C y A) - ver tercera columna de la tabla IV.

TABLA IV. TEMAS SUGERIDOS Y NIVELES DE APRENDIZAJE DE ACUERDO AL ESTUDIO DELPHI

Tópicos	Niveles de Aprendizaje por Tema	Nivel de Bloom
#1 Conceptos básicos de mediciones de software	Recordar terminología y conceptos	R
	Dar ejemplos de conceptos básicos, métodos y procedimientos de medición	C
	Explicar conceptos básicos	C
	Usar terminología y conceptos en un ejercicio o proyecto propuesto	A
#2 El proceso de medición	Recordar el proceso de medición	R
	Usar el proceso de mediciones en un proyecto o situación dado.	A
#3 Técnicas y herramientas para mediciones de software	Recordar las técnicas y herramientas de medición de software existentes	R
	Explicar las técnicas y herramientas de medición de software	C
	Usar una técnica en un ejercicio o proyecto	A
#4 Mediciones para la gestión de la ingeniería de software	Recordar conceptos relacionados a estimación de esfuerzo y medidas para planeación y control de proyectos	R
	Explicar cómo estimar el esfuerzo de un proyecto	C
	Medir el tiempo y esfuerzo en un proyecto	A
#5 Mediciones para la fase de requerimientos	Recordar los métodos más comunes de medición de tamaño funcional	R
	Interpretar cómo trabajan los métodos de medición de tamaño funcional	C
	Obtener el tamaño funcional de software en un ejercicio o proyecto siguiendo un método de medición	A

Por propósitos de generalización, efectuamos una verificación de resultados por medio de:

- Una encuesta realizada a profesores universitarios y profesionales de la rama que asistieron a conferencias de ingeniería de software llevadas a cabo en Septiembre, Octubre y Noviembre del 2012 [5]; y
- Entrevistas con escritores de libros de medición de software [5].

Los asistentes a las conferencias, que voluntariamente contestaron el cuestionario de verificación de resultados del Delphi, debían indicar su grado de acuerdo con los resultados obtenidos mediante el uso de una escala de Likert de cinco puntos (Muy en desacuerdo; en desacuerdo; Neutral; de acuerdo; y Muy de acuerdo). Es importante anotar que dos de los seis tópicos (*Técnicas y Herramientas* y *Mediciones para la Fase de Requerimientos*) obtuvieron porcentajes menores de acuerdo. Es decir, que las respuestas obtenidas fueron *De acuerdo* y *Muy de acuerdo*; a diferencia de los otros tópicos que obtuvieron *Muy de acuerdo.* Las razones dadas por los profesionales para justificar su selección han sido resumidas a continuación:

- La fase de requerimientos es la que impacta más fuertemente en el éxito del proyecto. Todos los problemas comienzan con requisitos pobres. Los estudiantes de carreras ligadas a las tecnologías de la información deben estar convencidos de que estudiar este tema es más importante que aprender sobre técnicas y herramientas.

- Las mediciones de la fase de requerimientos son usadas para las mediciones de la gestión, por tanto, las de requerimientos deben de aprenderse antes.
- Los estudiantes deben saber primero cuáles son las mediciones específicas antes de seleccionar herramientas. Las mediciones te indican "qué" medir y las herramientas "cómo" hacerlo.
- Las mediciones para el diseño son muy específicos; por tanto, son adecuados para estudiantes de maestría.
- Las mediciones de puntos de función -usadas en la fase de requerimientos- requieren tiempo para ser aprendidas, sería más conveniente que se enseñen a estudiantes de maestría y no de pregrado.

Con relación a los niveles de aprendizaje seleccionados por los participantes de la encuesta, se detectaron discrepancias; especialmente con respecto al tópico de *Mediciones para la Fase de Diseño*. Se evidenció que este tema es generalmente preferido por los profesores universitarios, pero no por los profesionales. Los resultados indican que aparentemente los profesores esperan que sus estudiantes apliquen la teoría mediante la puesta en prácticas de las mediciones de diseño (Ej: cohesión y acoplamiento); probablemente porque los estudiantes de pregrado están más familiarizados con tareas de diseño y programación.

Como se mencionó en la página anterior, por fines de verificación se entrevistaron escritores de libros de mediciones de software, cuatro autores en total. Durante la entrevista - 40 minutos en promedio, se les preguntó acerca de sus opiniones con respecto a los resultados obtenidos en el estudio Delphi. Ellos tenían la libertad de hacer comentarios y argumentar su acuerdo o desacuerdo con los resultados del Delphi, ya que la entrevista fue de tipo semiestructurada. Este tipo de entrevistas, se caracterizan por [19, 20]:

- Tener una guía de la entrevista (lista de preguntas que deben ser contestadas);
- Ser flexibles (cambiar el orden o añadir preguntas según el curso de la entrevista);
- Dar explicaciones con relación a las preguntas y pedir aclaraciones si la respuesta no es clara;
- Usar palabras apropiadas de acuerdo a la situación;
- Usar un estilo conversacional, pero manteniendo el enfoque en la guía.

En general, las opiniones recopiladas en las entrevistas, indican que los autores de los libros están de acuerdo con los resultados del Delphi, ya que el orden de los tópicos resultó ser el mismo; aunque, ellos manifestaron algunos desacuerdos que se presentan a continuación.

- Uno de los autores mencionó que las mediciones de gestión de software se les debe enseñar después de las mediciones específicas (Ej: de requerimientos y de diseño). Comentó que el ranking de los tópicos resultantes del Delphi está lógicamente ordenado desde el punto de vista de los profesionales que trabajan en organizaciones que ya tienen datos históricos para realizar sus estimaciones. Dijo que éste no es el caso de los estudiantes de pregrado, porque están en el proceso de aprendizaje de la medición del software, por lo que deberían aprender primero en la fase de requerimientos.
- Otro autor indicó que el ubicaría al tópico de mediciones de gestión de software en segundo lugar ya que considera que los estudiantes necesitan saber primero por qué necesitamos hacer mediciones específicas para obtener mediciones de gestión.
- Otro autor dijo que estaba de acuerdo con los tópicos seleccionados como prioritarios, a excepción de las mediciones para la fase de diseño, ya que piensa que este tópico no debería ser una prioridad. Según él, sólo los primeros cinco tópicos en el ranking deben ser considerados como obligatorios para los estudiantes universitarios.
- Dos autores sugirieron que la importancia de la medición del software debe ser fuertemente enfatizada. Uno de ellos también recomendó que se vincule la importancia de las mediciones con tener objetivos claros para medir el software; así como darles a conocer las consecuencias de no realizar mediciones. Por último sugirió que la academia debe buscar maneras para motivar a los estudiantes a medir.

Las opiniones de los entrevistados en relación a los niveles de aprendizaje que los estudiantes deben alcanzar por cada tópico priorizado fueron de alguna manera similares a los obtenidos en el estudio Delphi y en las encuestas realizadas en la fase de verificación. Esto significa que, los expertos -en su mayoría- eligieron los niveles de aprendizaje que corresponden a los primeros niveles de la taxonomía de Bloom (Recordar, Comprender y Aplicar). Sin embargo, se observaron las siguientes diferencias:

- En el caso de las mediciones para la fase de diseño, todos los entrevistados coinciden en que los estudiantes universitarios sólo deberían alcanzar los dos primeros niveles de aprendizaje (recordar y entender).
- Un entrevistado considera que los estudiantes de pregrado no deben alcanzar el nivel de aprendizaje *Aplicar* para los dos tópicos siguientes: Proceso de Medición, y Técnicas y Herramientas. Es decir, los estudiantes no deberían utilizar un proceso de medición en un proyecto, ni usar técnicas de medición en un ejercicio o proyecto. Mientras que otro entrevistado manifestó su desacuerdo en que los estudiantes realicen mediciones del tamaño de los requerimientos funcionales del software. Según estos expertos (ambos con experiencia en docencia universitaria), los maestros pueden enfrentar restricciones de tiempo al tratar de hacer frente a los niveles más altos de aprendizaje (desde el nivel *aplicar* hasta el nivel *crear*). Además, uno de los expertos dijo que los tópicos *Proceso de Medición*, y *Técnicas y Herramientas* son dependientes del contexto. Por lo tanto, llegar al nivel de *comprensión* de estos temas puede ser suficiente para los estudiantes de pregrado.

Para resumir esta sección, podemos decir que los resultados del estudio Delphi sugieren indicar que cinco tópicos de mediciones de software son altamente recomendados para ser enseñados a los estudiantes de pregrado. La primera columna de la Tabla IV muestra estos temas en el orden de prioridad que obtuvieron en el estudio.

De las Tablas III y IV, se puede observar que el orden de estos tópicos concuerda de alguna manera con la preferencia de tópicos identificada en la encuesta vía web administrada a profesionales (ver literal A de la sección III). En otras palabras, cuatro tópicos definidos como prioritarios en el estudio Delphi aparecen entre los cinco primeros tópicos seleccionados por los profesionales que participaron en la encuesta web.

La segunda columna de la tabla IV muestra los niveles de aprendizaje -por tema- que alcanzaron un alto porcentaje de preferencia entre los expertos (más del 70%). Esto es, los niveles de aprendizaje que los estudiantes de pregrado

necesitan alcanzar. En la Tabla IV, el mapeo entre los enunciados del cuestionario y la taxonomía de Bloom aparecen a la derecha (R: Recordar; C: Comprender; A: Aplicar). Cabe mencionar que el cuestionario usado en la primera fase de este estudio incluyó enunciados que cubrían los seis niveles de aprendizaje de la taxonomía de Bloom; no obstante, solo tres de éstos fueron seleccionados por los participantes (R, C y A) y pasaron a otras fases.

IV. Sugerencias para un currículo de ingenieria de software

La información recogida de las secciones 2 y 3 fue usada como dato de partida para identificar mejoras a tomarse en cuenta para la revisión y actualización de los lineamientos del currículo de ingeniería de software a nivel de pregrado. Las mejoras sugeridas son las siguientes:

A. Estandarización de las áreas de conocimiento

En el mapeo realizado entre las directrices curriculares (SE2004), el Cuerpo de Conocimiento de mediciones de software y la guía SWEBOK, observamos que las áreas de conocimiento (denominados como tópicos en el Cuerpo de Conocimiento de Mediciones de software) tienen un contenido similar pero utilizan nombres diferentes. Por lo tanto, nuestra primera recomendación es la de usar el mismo nombre de un área de conocimiento para todos los cuerpos de conocimiento. Sugerimos, además, que el nombre de cada área sea el que utiliza el SWEBOK v3, ya que es el documento a partir del cual se desarrollaron los otros cuerpos de conocimiento y guías curriculares. Por ejemplo:

- Se debería usar como nombre *Requerimientos de Software* (SWEBOK v3.0) en lugar de *Modelamiento y Análisis de Software* (SEEK –SE2004)
- Se debería usar Pruebas de Software en vez de Verificación y Validación de Software.

Consideramos que esta estandarización de nombres facilitará el trabajo de voluntarios y otras personas que hacen uso de los Cuerpos de Conocimiento para diseño de currículos y otros propósitos.

B. Versiones de la taxonomía de Bloom

En nuestra revisión de las directrices curriculares SE2004, observamos que los niveles de aprendizaje incluidos en dicho documento están basados en la versión original de la taxonomía de Bloom [4]. A este respecto, recomendamos usar la versión actualizada de esta taxonomía publicada por Anderson et al en 2001 [18]. En esta versión actualizada, los nombres de los niveles de aprendizaje fueron cambiados de sustantivos a verbos. Por ejemplo: de Conocimiento a Recordar; de Comprensión a Comprender; de Aplicación a Aplicar; y así sucesivamente. Todos los estudios resumidos en este artículo han usado la versión actualizada.

C. Centrarse en prioridades

En las directrices curriculares SE2004, todos los temas relacionados a medición de software incluidos en SEEK son considerados esenciales (E) – Ver tabla V, columna R de las Guías curriculares vigentes. Sin embargo, el trabajo de investigación -resumido en la sección III de este artículo- revela que existen prioridades en este campo; las mismas que deberían tomarse en cuenta al momento de preparar cursos universitarios relacionados a mediciones de software.

Basándonos en los hallazgos de los estudios realizados, especialmente aquellos del estudio Delphi, consideramos que los cinco tópicos prioritarios son aquellos que deberían considerarse como esenciales (E) en las nuevas directrices curriculares para programas de pregrado de ingeniería de software. Estos tópicos, al ser menos numerosos, pueden ser profundizados con los estudiantes para alcanzar los niveles de aprendizaje esperados. Este planteamiento involucra la re-conceptualización de las directrices curriculares, la misma que debería enfocarse en la profundización de los tópicos prioritarios en lugar de cubrir contenidos extensos (muchos tópicos). Consideramos, además, que los tópicos propuestos como deseables (D) para estudiantes de pregrado, bien podrían considerarse como esenciales para estudiantes de posgrado.

Nuestra sugerencia para formular una nueva versión de las directrices curriculares para programas de pregrado de Ingeniería de Software –en lo que respecta únicamente a tópicos relacionados a mediciones de software- se muestra en la tabla V. En dicha tabla, se presentan: las áreas de conocimiento (tomadas de SWEBOK v3.0); las unidades de conocimiento sugeridas; los tópicos de mediciones de software, los niveles de aprendizaje (N) y la relevancia de cada tópico (R). Es importante anotar, que en dicha tabla aparece un total de siete tópicos: cinco considerados como prioritarios de acuerdo a los resultados expuestos en la sección III y dos (Diseño de Software y Calidad del Software) que captaron una atención considerable de los profesionales encuestados.

A los cinco tópicos prioritarios se les asignó una relevancia E (Esencial) y a los dos restantes D (Deseable). Asimismo, es importante indicar que los niveles de aprendizaje mostrados en la Tabla V sólo corresponden a los niveles más altos seleccionados por los participantes del estudio Delphi. Por ejemplo, para el tema #1 – Conceptos Básicos de Mediciones de Software, los participantes escogieron tres niveles de aprendizaje (Recordar, Comprender y Aplicar); por lo tanto, la Tabla V sólo presenta el nivel más alto (Aplicar).

De acuerdo a la Tabla V, tres consideraciones deben tomarse en cuenta para formular nuevas directrices curriculares:

1) *La inclusión del tema de Medición del Tamaño Funcional en el área de conocimiento Requerimientos de Software.* Las guías curriculares vigentes (SE2004) mencionan este tema como parte del curso SE323 Gestión de Proyectos de Software sugerido en los lineamientos [2]. Sin embargo, este tema no aparece en la tabla asociada al SEEK donde se sugieren los temas, niveles de aprendizaje y relevancia para estudiantes de grado.
2) *La revisión del tema Métricas de Diseño*. Este tipo de mediciones (por ejemplo, factores arquitectónicos) no esté presente en el área de conocimiento de Diseño de Software del SWEBOK v3.0 ni tampoco en el Cuerpo de conocimiento de Mediciones de Software. Por consiguiente, sugerimos que se haga una revisión de dicha área de conocimiento para determinar el contenido a ser incluido en los cuerpos de conocimiento; y así mismo determinar si éste es apropiado para los estudiantes de pregrado o posgrado. Nuestro estudio no incluyó este tema en la lista de ejemplos de tópicos de medición que los participantes debían escoger, ya que tomamos como referencia el cuerpo de conocimiento de mediciones de software y SWEBOK v3.0. Entre los ejemplos que presentamos a los participantes, sólo incluimos el tema Mediciones de Atributos de Diseño (por ejemplo: acoplamiento, cohesión, etc.) que si se encuentra en los cuerpos de conocimientos antes mencionados.

TABLA V. SUGERENCIAS PARA ELABORAR NUEVAS DIRECTRICES CURRICULARES

Areas de Conocimiento (tomadas de SWEBOK v3.0)	**Unidades de conocimiento (sugeridas)**	**Guías curriculares vigentes (SEEK del SE2004)**			**SUGERENCIAS para actualizar las directrices curriculares de ingeniería de software basadas en SWEBOK v3.0**		
		Tópicos	**N**	**R**	**Tópicos**	**N**	**R**
Fundamentos de Matemáticas e Ingeniería	Fundamentos de Ingeniería para el Software	Métricas y mediciones	K	E	Conceptos básicos de mediciones de softare (definición, unidades, escalas, etc)	A	E
		Teoría de mediciones	C	E			
Requerimientos de Software	Mediciones de requerimientos de software	*No encontrado en SEEK*			Medición de talla funcional (qué es?, métodos)	A	E
Diseño de Software	Análisis y evaluación de la calidad del diseño	Mediciones de atributos del diseño (ej. Acoplamiento, cohesión,etc.)	K	E	Mediciones de diseño (orientadas a objetos y a funciones)	U	D
		Métricas de diseño (ej. Factores arquitectónicos, interpretación, etc.)	A	E	*No encontrado en SWEBOK v3.0 (se sugiere una revisión)*		
Proceso de Ingeniería de Software	Técnicas de medición de procesos de software	Medición y análisis de procesos de software	C	E	Introducción a técnicas y modelos de medición de procesos de software (análisis de causa-raíz, mejora y evaluación de procesos, ejemplos de modelos y métodos para procesos de software)	A	E
		Análisis y control de la calidad (ej. Prevención de defectos, métricas de calidad, análisis de causa raíz, etc.)	C	E			
		Proceso de software individual (modelo, definición, mediciones, análisis, mejora)	C	E			
		Procesos de software en equipos (modelo, definición, organización, mediciones, análisis, mejora)	C	E			
Calidad del Software	Mediciones de la calidad del software	Modelos y métricas de calidad del Software	C	E	Medición de la calidad del software (el costo de la calidad, estándares y modelos relacionados que incluyen mediciones -Ej. ISO/IEC/IEEE12207 and CMMI)	R	D
		Métricas y mediciones de la calidad del producto	C	E			
Gestión de la Ingeniería de Software	Planificación de proyectos de software	Estimación de esfuerzo	A	E	Estimación de esfuerzo	A	E
	Mediciones en ingeniería de software	Medición y análisis de resultados	A	E	Medición y análisis de resultados	U	D
		No encontrado en SEEK			El proceso de medición	A	E

3) La inclusión del tema Proceso de Medición dentro del área de conocimiento Gestión de Ingeniería la de Software. Este tema no fue considerado en ninguna de las áreas de conocimiento del SE2004.

V. CONCLUSIONES

Este artículo presenta recomendaciones a ser consideradas para actualizar las directrices del currículo de ingeniería de software para programas de pregrado basadas en el análisis de los cuerpos de conocimiento, los lineamientos curriculares existentes y las aportaciones de profesores universitarios y profesionales de la ingeniería de software en ejercicio.

Primero, el análisis de los cuerpos de conocimiento revela una presencia significativa de temas de medición de software dentro de sus áreas de conocimiento (KAs). De hecho, la última versión de SWEBOK v3.0 contiene temas de medición en 11 de sus KAs.

Segundo, los resultados de la encuesta realizada vía web y del estudio Delphi permitieron formular recomendaciones para el desarrollo de nuevas directrices curriculares para programas de pregrado en ingeniería de software. Las recomendaciones abarcan la inclusión de temas esenciales de medición de software que deben estar presentes en los nuevos lineamientos

curriculares, así como sus correspondientes niveles de aprendizaje y relevancia.

Los autores de este artículo sostenemos que estas recomendaciones son útiles para ayudar a los profesores universitarios en la selección de temas de medición de software que deben priorizarse en los cursos de pregrado que ellos imparten. También sostenemos que la metodología aplicada en este trabajo de investigación puede ser valiosa para diseñadores de currículos e investigadores que se enfocan en temas educacionales.

Finalmente, una implicación práctica es que la metodología utilizada en este estudio puede aplicarse no sólo en investigaciones relacionadas a la enseñanza de mediciones de software, sino también en cualquier ciencia o campo de la ingeniería. Esto es factible ya que la metodología proporciona un enfoque bien estructurado para identificar prioridades en cualquier ámbito del conocimiento.

REFERENCIAS

[1] P. Bourque and R. Fairley, SWEBOK : Guide to the Software Engineering Body of Knowledge: IEEE Computer Science, 2014.

[2] IEEE ACM. (2004, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. SE2004 Volume – 8/23/2004, 135. Available: http://sites.computer.org/ccse/SE2004Volume.pdf

[3] A. Abran, A. April, and L. Buglione. (2010, 26 January 2011). Software Measurement Body of Knowledge. Encyclopedia of Software Engineering 1:1(1), 1157 — 1168.

[4] B. Bloom, Taxonomy of educational objectives the classification of educational goals, First edition ed. New York Green Longmans, 1956.

[5] M. Villavicencio, "Development of a framework for the education of software measurement in software engineering undergraduate programs," Ph. D., Software Enginnering and Information Technology, École de technologie supérieure, Montreal, 2014.

[6] P. Bourque, A. Abran, J. Garbajosa, G. Keeni, and B. Shen. (2008, October 15, 2010). SWEBOK Version 3. 18. Available: http://www2.computer.org/cms/Computer.org/SWEBOK/MeasurementKA-Draft-Feb2008.pdf

[7] M. Villavicencio and A. Abran, "The Necessary Software Measurement Knowledge in Software Engineering Education from the Practitioners' Point of View," in 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) Montreal, 2012.

[8] T. Amos and N. Pearse, "The delphi technique and educating entrepreneurs for the future," in 7th European Conference on Research Methodology for Business and Management Studies, ed: Academic Publishing Limited, Reading, UK 2008 2008, pp. 17-24.

[9] D. W. Gatchell, R. A. Linsenmeier, and T. R. Harris, "Determination of the core undergraduate BME curriculum - the 1st step in a Delphi study," in Conference Proceedings. 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 1-5 Sept. 2004, Piscataway, NJ, USA, 2004, pp. 5200-1.

[10] P. Howze and C. Dalrymple, "Consensus without all the meetings: using the Delphi method to determine course content for library instruction," Reference Services Review, vol. 32, pp. 174-84, 2004.

[11] H.-L. Hung, J. W. Altschuld, and Y.-F. Lee, "Methodological and conceptual issues confronting a cross-country Delphi study of educational program evaluation," Evaluation and Program Planning, vol. 31, pp. 191-198, 2008.

[12] J. Lambeth, R. Joerger, and J. Elliot, "Implications for Focusing Research in Career and Technical Education and Workforce Development," Career and Technical Education Research, vol. 34, pp. 137-153, 2009.

[13] P. Bourque, R. Dupuis, A. Abran, J. W. Moore, L. Tripp, and S. Wolff, "Fundamental principles of software engineering - a journey," Journal of Systems and Software, vol. 62, pp. 59-70, 2002.

[14] E. Hall, "The Delphi Primer: Doing Real-World or Academic Research Using a Mixed-Method Approach," in The Refractive Thinker®: Vol II Research Methodology vol. 2, ed: The Lentz Leadership Institute, 2009, pp. Kindle Locations 103-104.

[15] C. Okoli and S. D. Pawlowski, "The Delphi method as a research tool: an example, design considerations and applications," Information & Management, vol. 42, pp. 15-29, 2004.

[16] R. C. Schmidt, "Managing Delphi Surveys Using Nonparametric Statistical Techniques*," Decision Sciences, vol. 28, pp. 763-774, 1997.

[17] G. J. Skulmoski, F. T. Hartman, and J. Krahn, "The Delphi Method for Graduate Research," Journal of Information Technology Education, vol. 6, pp. 1-21, 2007.

[18] L. Anderson, D. Krathwohl, P. Airasian, K. Cruikshank, R. Mayer, P. Pintrich, et al., A taxonomy for learning, teaching and assessing. A revision of Bloom's taxonomy of Educational Objectives. New York: Addison Wesley Longman, Inc, 2001.

[19] A. Bhamani Kajornboon. (2005, Using interviews as research instruments. 10. Available: www.culi.chula.ac.th/e-journal/bod/annabel.pdf

[20] P. Leedy and J. E. Ormrod, Practical research: planning and design, Ninth edition ed.: Pearson, 2010.

Mónica Villavicencio. Obtuvo su doctorado en Ingeniería de Software (2014) en la École de Technologie Supérieure ETS -Université du Québec (Montreal, Canadá). Ella es profesora titular en la Escuela Superior Politécnica del Litoral (ESPOL) en Guayaquil, Ecuador donde imparte cursos de ingeniería de software para los programas de pregrado y postgrado. Sus intereses de investigación son: mejora de procesos de software, mediciones de software, gestión de proyectos de software y calidad de software.

Alain Abran. Tiene un doctorado en Ingeniería Eléctrica y Computación (1994) de la École Polytechnique de Montréal (Canadá). El es profesor en la École de Technologie Supérieure (ETS) - Universidad de Quebec (Montreal, Canadá). Tiene más de 20 años de experiencia en la industria de desarrollo de sistemas de información e ingeniería de software. Sus intereses de investigación son: Modelos de estimación, fundamentos de la ingeniería de software, medición del tamaño funcional del software, gestión de riesgos y mantenimiento de software.