Uso de la Taxonomía Curricular ACM para Mejorar la Carrera de Computación

Adolfo Di Mare
Universidad de Costa Rica
adolfo.dimare@ecci.ucr.ac.cr

RESUMEN

Se usa como base la taxonomía curricular de la Association for Computing Machinery (ACM) para llegar al consenso académico que permite mejorar la carrera de computación. También se discuten las ventajas de codificar los contenidos de cursos usando esa taxonomía.

Palabras claves: Curriculum, ingeniería, computación, taxonomía curricular.

ABSTRACT

The Association for Computing Machinery (ACM) curricular taxonomy is used as a base of to reach the academic consensus to improve a career in computing. The advantages of encoding the content of courses using this taxonomy is also addressed.

Keywords: Curriculum, engineering, computing, curricular taxonomy.

1. INTRODUCCIÓN

Cuando una academia es joven cada docente siente que puede impartir cualquier curso, y lo hace con un interés exploratorio porque desea aprender y profundizar su conocimiento. Cuando alcanza la madurez su actitud cambia pues, además de que ya conoce bien sus limitaciones, también comprende que debe focalizar sus intereses pues quien mucho abarca poco aprieta. Por eso, la madurez académica a veces acarrea también algunos de los problemas de la vejez, pues conforme cada docente se consolida en una área del saber tiene la tendencia de evitar su desarrollo en otras áreas.

En muchas universidades es usual que los profesores más viejos tengan propiedad ("tenure", en inglés), lo que les da una posición de negociación mucho más fuerte, pues cada catedrático tiene protección especial que se ha ganado con sus logros académicos. Por eso, quien imparte "Programación" a duras penas cambiará su curso por "Redes" o por "Bases de datos". Además, su sólida posición académica le permite continuar impartiendo un curso similar al que ha preparado por una década, pues la "libertad académica" aunada a la "calidad académica" justifica que sea aquella persona con más alto rango académico quien define qué debe incluirse en el temario de un curso y también quién debe impartirlo. De esta manera, cada catedrático se convierte en un agente en contra del cambio o la renovación: a menos que el sistema académico sea revolucionado el estancamiento es el resultado de la madurez de la academia.

En algunas disciplinas el saber acumulado requiere pocos cambios docentes. Por ejemplo, en las ciencias jurídicas los cambios son menores, como por ejemplo incorporar las firmas digitales en contratos, pero la mayor parte del conocimiento ya está bien elaborado. En computación ya la tecnología se ha establecido y los cambios ya no son tan radicales como lo fueron en los primeros 50 años de la disciplina, pero siempre es necesario mejorar cada carrera para adaptarla a las nuevas necesidades que el desarrollo tecnológico crea. Como la madurez académica muchas veces impide que estas mejoras puedan llevarse a cabo, porque cada docente defiende su feudo, a fin de cuentas la única forma de lograr introducir mejoras es aumentar la cantidad de cursos en el plan de estudios. Esta política académica encarece el costo de la carrera, tanto en tiempo como en dinero, pues es necesario impartir más

cursos y además cada graduado debe permanecer más tiempo en el claustro universitario, lo que también disminuye la cantidad de graduados, pues algunos estudiantes no logran graduarse porque la probabilidad de que fallen en algún curso aumenta.

Hay varias maneras de lograr hacer los cambios o mejoras que una carrera necesita. Una forma es nombrar el equivalente a un tirano académico, encargándole la misión de desgarrar el programa actual a su antojo para luego decretar el nuevo, dejando de lado cualquier oposición que pueda surgir de cualquier otro académico; este método es inaceptable pues violenta la estructura misma de cualquier academia. La siguiente forma de proceder es crear dos o más versiones de la carrera a partir de la original, cuidando de mantener cursos de intersección entre cada una de las carreras. La otra forma de lograr mejoras curriculares es llegar a un consenso académico, el que requiere que una buena mayoría de los docentes accedan a hacer los cambios.

En conjunto el Institute of Electrical and Electronics Engineers Computer Society (IEEE) y la Association for Computing Machinery (ACM) avalaron el documento llamado (CS2001) en el que se definen cuatro profesiones en computación: Ciencias de la Computación, Ingeniería de Computadores, Ingeniería de Software y Sistemas de Información ([CS] Computer Science, [CE] Computer Engineering, [SE] Software Engineering, [IS] Information Systems) [ACM-2001]. En la versión 2008 de este currículo, llamada apropiadamente (CS2008) se mencionan una nueva carrera además de las cuatro anteriores: Tecnologías de Información, ([IT] Information Technology) [ACM-2008]. La taxonomía del conocimiento en que se basa este artículo es la definida en el documento (CS2008) disponible aquí [ACM-2012b]:

`http://acm.org/education/curricula-recommendations`

Propongo una forma de lograr avances para alcanzar el consenso académico necesario para mejorar una carrera de computación. La metodología es relativamente simple, y consiste en usar la taxonomía del conocimiento en computación preparada por la ACM para reformular el contenido de los cursos, incorporando también las Competencias Generales y Profesionales similares a las propuestas por [Arias-2006], con el fin de reacomodar los cursos para sintetizar un plan de la carrera que permita no solo minimizar la cantidad de cursos sino también incorporar las habilidades transversales que son necesarias en todo graduado universitario.

2. LA TAXONOMÍA ACM DEL CONOCIMIENTO EN COMPUTACIÓN

Cada una de las áreas del conocimiento en computación ha sido identificada y claramente definida en la taxonomía curricular contenida en (CS2008), que es la versión 2008 del Currículo en Computación de ACM [ACM-2008]. La revisión de esta taxonomía estará disponible en 2013, pero ya existe una primera versión de este documento [ACM-2012a]:

	AL	Algorithms and Complexity	Algoritmos y Complejidad
	AR	Architecture and Organization	Arquitectura y Organización
	CN	Computational Science	Ciencia Computacional
	DS	Discrete Structures	Estructuras Discretas
	GV	Graphics and Visual Computing	Graficación y Computación Visual
	HC	Human-Computer Interaction	Interacción Hombre Máquina
*	IAS	Security and Information Assurance	Validación de Información y Seguridad
	IM	Information Management	Administración de Información
	IS	Intelligent Systems	Sistemas Inteligentes
+	NC	Networking and Communication	Redes y Comunicación
	OS	Operating Systems	Sistemas Operativos
*	PBD	Platform-based Development	Desarrollo Basado en Plataformas
*	PD	Parallel and Distributed Computing	Computación Paralela y Distribuida
	PL	Programming Languages	Lenguajes de Programación
*	SDF	Software Development Fundamentals	Fundamentos para Desarrollo de Software

	SE	Software Engineering	Ingeniería de Software
+	SF	Systems Fundamentals	Fundamentos de Sistemas
	SP	Social and Professional Issues	Asuntos Profesionales y Sociales

Figura 1

Fuente: "Appendix A: Overview of the Body of Knowledge"
http://acm.org/education/curricula/ComputerScience2008.pdf
http://ai.stanford.edu/users/sahami/CS2013/strawman-draft/cs2013-strawman.pdf

Las 14 áreas definidas en el reporte (CS2008) fueron transformadas en 18 áreas para la nueva versión de la taxonomía [ACM-2012a]. Aquí se muestran marcadas con "*" las 4 nuevas áreas; también están marcadas con "+" las que fueron sustancialmente modificadas. Cada una de éstas 18 áreas del conocimiento está subdividida en varias unidades de conocimiento que contienen los temas de relevantes al área; en total (CS2008) contiene la definición de 147 unidades de conocimiento. Por ejemplo, para la área de conocimiento "Fundamentos de Programación" (PF) el Currículo ACM define las siguientes unidades:

(SDF) Software Development Fundamentals	
SDF/AlgorithmsDesign	Algoritmos y Diseño
SDF/FundamentalProgrammingConcepts	Conceptos Fundamentales de Programación
SDF/FundamentalDataStructures	Estructuras de Datos Fundamentales
SDF/DevelopmentMethods	Métodos de Desarrollo

Figura 2

A cada unidad de conocimiento se le identifica unívocamente con su código mnemónico que contiene 2 partes separadas por la barra "/": el código de 2 o 3 letras del área y el nombre del la unidad. Por ejemplo, el código "SDF/FundamentalDataStructures" identifica la unidad "Estructuras de Datos Fundamentales" del área "Fundamentos para Desarrollo de Software" (SDF).

(SDF) Software Development Fundamentals (9)
→ Vectores
→ Registros o estructuras (agregados heterogéneos)
→ Hileras y procesamiento de cadenas
→ Pilas, colas, colas de prioridad, conjuntos y diccionarios
→ Referencias y alias
→ Estructuras enlazadas simnples
→ Estrategias para la elecciÃ³n de la estructura de datos apropiada

Figura 3

De lo ya expuesto se deduce que la jerarquía taxonómica (CS2008) está organizada en los siguientes 3 niveles: 1) área, 2) unidad o tema y 3) tópico o subtema. De forma natural se puede usar esta taxonomía para definir el contenido de un curso al agrupar varios temas. Por ejemplo, un curso de "Programación Avanzada" podría formularse agrupando los siguientes temas (listados aquí en orden alfabético, no en orden cronológico de exposición):

Programación Avanzada
AL/BasicAnalysis
AL/DistributedAlgorithms
AL/FundamentalAlgorithms
HC/BuildingGUIInterfaces
HC/Foundations
PL/ObjectOrientedProgramming

Figura 4

Usar la taxonomía (CS2008) para definir cursos es una forma directa de aprovechar el trabajo realizado por la ACM, lo que también facilita estandarizar el conocimiento impartido en versiones diferentes de las carreras de computación.

El reporte (CS2008) incluye una estimación en horas de la cantidad mínima de tiempo requerida para cubrir cada tema, con el fin de definir un cuerpo básico de temas ("core") que no supere 280 horas. Por eso, ACM recomienda una cantidad mínima de horas que deben usarse para impartir cada unidad de cocimiento. Por ejemplo, el área de conocimiento "Algoritmos y Complejidad" (AL) tiene asignado un total de 31 horas para cubrir cinco temas del cuerpo básico de formación: AL/BasicAnalysis, AL/AlgorithmicStrategies, AL/FundamentalAlgorithms, AL/DistributedAlgorithms, AL/BasicComputability. Esto quiere decir que, de acuerdo a la recomendación (CS2008), cualquier carrera en computación ([CS] [CE] [SE] [IS] [IT]) debe invertir alrededor del 10% (31/280) del tiempo de instrucción dedicado a la formación básica en los temas de "Algoritmos y Complejidad".

En el sistema estatal universitario de Costa Rica la medida curricular que se usa no es de "horas" sino de "créditos", definidos de la siguiente manera: "Crédito es una unidad valorativa del trabajo del estudiante, que equivale a tres horas reloj semanales de trabajo del mismo, durante 15 semanas, aplicadas a una actividad que ha sido supervisada, evaluada y aprobada por el profesor" [CONARE-1976]. En la práctica se supone que un estudiante debe invertir 2 horas de estudio individual, en casa o en el laboratorio, por cada hora de instrucción presencial. Por ejemplo, un curso que requiere la asistencia a 4 horas de lección por semana tiene 4 créditos y, en total, requiere 12 horas de estudio por semana durante 15 semanas para un total de 180==3*4*15 horas de estudio de las que 120 son estudio individual.

Esta definición de crédito permite calcular la cantidad de créditos sumando la cantidad de horas recomendadas por ACM para cada tema, para obtener el total de horas, y luego dividiendo entre 15 (la duración en semanas de cualquier curso). Por ejemplo, si al definir el contenido de un curso con base en el temario ACM se escogen 6 temas, cada uno con una dedicación de (2 9 6 10 4 8) horas, en total se requerirán 39==(2+9+6+10+4+8) horas para impartir esos temas, lo que significa que un curso de 4 créditos que tenga 4 horas presenciales de lección, para un total de 60==4*15 horas, permitirá impartir todos los 6 temas dejándole al docente una holgura de 21==(60-39) horas, que es un poco más del 33% del tiempo disponible para el curso. Si en lugar de 4 se le asignaran 3 créditos a ese mismo curso, el programa del curso quedaría muy "apretado" lo que podría tener efectos negativos en la calidad académica. Para concretar si llamamos CR el total de créditos de un curso y sum(ACM) el total de horas recomendado para los temas en (CS2008), la forma de calcular CR a partir de sum(ACM) es usar esta fórmula: CR=ceil(sum(ACM)/15).

3. TAXONOMÍA DE BLOOM

Junto al temario específico de cada área del conocimiento, el Currículo ACM también especifica cuáles capacidades de desempeño debe adquirir cada estudiante, definiendo cuáles son los objetivos de aprendizaje para cada tema. Por ejemplo, para " Estructuras de Datos Fundamentales" fueron definidos estos objetivos de aprendizaje:

- Discutir el uso apropiado de estructuras de datos incluidas en el lenguaje. [Conocimiento]
- Describir aplicaciones comunes para cada estructura de datos en la lista de tópicos. [Conocimiento]
- Comparar implementaciones alternativas de estructuras usando como criterio el rendimiento. [Evaluación]
- Escribir programas que usan cada una de las siguientes estructuras de datos: vectores, hileras, listas enlazadas, pilas, colas, conjuntos y diccionarios. [Aplicación]
- Comparar y contrastar los costos y beneficios de estructuras de datos dinámicas y estáticas. [Evaluación]
- Escoger la estructura de datos apropiada para modela un problema dado. [Evaluación]

Además de cada objetivo de aprendizaje también está definido el nivel de destreza que debe adquirir el estudiante. El significado de este nivel es el siguiente [ACM-2012a]:

Conocimiento

El estudiante entiende qué es un concepto o qué significa. Este nivel de destreza provée un nivel básico de información sobre el concepto en lugar de la habilidad requerida para aplicar el conocimiento.

Aplicación

El estudiante es capaz de aplicar el concepto de una forma concreta. Aplicar el concepto puede incluir, por ejemplo, la habilidad de implementar un concepto de programación, usar una técnica particular de prueba o realizar un tipo particular de análisis.

Evaluación

El estudiante es capaz de considerar el concepto desde mútiples puntos de vistas o de justificar la elección de una estrategia particular para resolver un problema. Este nivel de destreza implica más que la aplicación de un concepto: incluye la habilidad de seleccionar un enfoque apropiado tomando en cuenta alternativas bien entendidas.

Estos 3 niveles de aprendizaje son una simplificación práctica adecuada de los niveles de conocimiento definidos en la Taxonomía del Conocimiento de Bloom, que fué propuesta a mediados del Siglo XX por los autores Bloom y Krathwohl [BK-1956] en un espectro que va desde lo concreto a lo abstracto: conocer, comprender, aplicar, analizar, sintetizar y evaluar. Esta jerarquía de 6 niveles de conocimiento fue revisada posteriormente a principios del Siglo XXI en un libro editado por Anderson y Krathwohl [AK-2001] u fué posteriormente usada para redactar el informe (CS2008).

En [Churches-2009] se refina un poco más la definición aportada por [AK-2001] con el fin de adaptarla a la era digital, incluyéndole otros verbos relevantes a cada nivel de conocimiento. En la siguiente lista aparecen los verbos usados en (CS2008) y, entre paréntesis, los nuevos aportados por [Churches-2009]:

Recordar [nivel 1]:

reconocer, recordar, describir, declarar (listar, identificar, recuperar, denominar, localizar, encontrar)

Entender [nivel 2]:

interpretar, ejemplificar, clasificar, inferir, comparar, explicar, parafrasear, sumarizar (resumir)

Aplicar [nivel 3]:

ejecutar (i.e. llevar a cabo), implementar (i.e. usar), computar, manipular, resolver (desempeñar)

Analizar [nivel 4]:

diferenciar, organizar, atribuir, discriminar, distinguir, subdividir (comparar, deconstruir, delinear, encontrar, estructurar, integrar)

Evaluar [nivel 5]:

verificar, criticar, valorar, comparar, contrastar (revisar, formular hipótesis, experimentar, juzgar, probar, detectar, monitorear)

Crear [nivel 6]:

generar, planear, producir, innovar, idear, diseñar, organizar (construír, trazar, elaborar)

Usando esta lista de verbos es posible definir el nivel de cada objetivo de aprendizaje. Por ejemplo, “Analizar y explicar ...” tiene un nivel 4 [Analizar] en la jerarquía, mientras que “Elegir la construcción ...” tiene un nivel 2 [Entender] (el artículo [Ferrer-2003] incluye una lista más completa de verbos).

Si se usa (CS2008) o [ACM-2012a] para definir el temario de los cursos, se obtiene también la definición de las capacidad de desempeño junto con el nivel de Bloom para cada objetivo de aprendizaje. Además, a pesar de que el temario es tan detallado, todavía queda una gran amplitud para cualquier docente pueda introducir cambios que ayuden a mejorar el curso por estas dos razones. Primero, no queda escrita en piedra cuál es la secuencia en que

debe ser abordado cada tema y segundo, cada tópico puede ser impartido de muchas formas diferentes, lo que contrasta positivamente con la mala costumbre de definir, semana por semana o lección por lección, cuál es el tema impartido.

Parece paradójico, pero en muchos casos al usar el temario ACM se obtiene una definición mucho más concreta de qué es cada curso sin ponerle límites innecesarios al docente. Por eso, al especificar el curso en términos del temario ACM se obtiene un mejor resultado que si se hace de la forma tradicional, la que podríamos calificar como "artesanal", pues se evita la rigidez que es usual en muchos programas de estudios.

4. COMPETENCIAS, HABILIDADES, DESTREZAS Y ACTITUDES

De acuerdo a [Schmidt-2006], "... la mayor parte de autores incluyen en el concepto de competencia la adquisición de conocimientos, la ejecución de destrezas y el desarrollo de talentos que se expresan en el saber, el saber hacer y el saber ser, es decir, al conjunto de conocimientos, procedimientos, ejecuciones, actitudes y valores coordinados, combinados e integrados en el ejercicio profesional". La formación del futuro profesional debe incluir no solo cada uno de los temas mencionados en el plan de estudios, sino también las competencias que requiere para lograr un desempeño adecuado y, por eso, es saludable incluir en el plan de estudios la definición de las competencias profesionales que cada graduado debe asimilar. Para eso es necesario definir estas competencias.

A diferencia de los temas definidos en la taxonomía (CS2008), a las competencias se les define como ejes transversales, pues deben ser impartidas en muchos cursos para que, por repetición, sean absorbidas paulatinamente por todos los estudiantes.

En la Escuela de Ciencias de la Computación e Informática [ECCI] de la Universidad de Costa Rica se ha dado un largo proceso para mejorar el plan de estudios del Bachillerato en Computación e Informática, cuya última modificación sustancial ocurrió en el año 2000 (cuando todavía la mayor parte del profesorado no había alcanzado el rango de catedrático). Más de una década se ha utilizado para llegar a un consenso que permita modificar el plan de estudios, pero lo que se ha logrado se reduce a aumentar la cantidad de cursos hasta llegar al máximo legal de 144 créditos (el mínimo es 128) [CONARE-2004]. Sin embargo, en el proceso de discusión académica se logró crear una lista de Competencias Generales en Computación, la que luego usó el profesor Arias para sintetizar la lista que aquí reproduzco [Arias-2006]:

Competencias Generales en Computación

XI Intelectual

- Capacidad de abstracción y síntesis
- Capacidad de diseño y modelado
- Capacidad de investigación
- Capacidad verbal
- Dominio de idiomas (inglés)

XG Sicológica

- Capacidad crítica
- Tenacidad y tolerancia
- Dominio táctico y estratégico
- Imaginación y creatividad
- Iniciativa
- Capacidad grupal
- Liderazgo

XM Moral

- Probidad moral
- Capacidad ética
- Conocimiento jurídico y legal

XS Social

Conciencia social
Conciencia ambiental
Capacidad económica

En [Arias-2006] cada una de estas competencias viene acompañada por su descripción, que permite definirla con precisión. Por ejemplo, definición general de las competencias intelectuales es la siguiente:

XI Competencias intelectuales

Son aquellas que se refieren a capacidades intelectuales de orden general. Es decir, capacidades que se requieren para desempeñar cualquier aspecto de la profesión, y no alguno en particular. Dependen, esencialmente, de la formación recibida por la persona desde sus primeros años de vida.

Para cada una de las competencias específicas se incluye también su definición concreta:

XI Competencias intelectuales → Capacidad de abstracción y síntesis

Se refiere a lo que comúnmente se denomina "capacidad de análisis". Incluye la capacidad del individuo de estudiar problemas, logrando niveles de síntesis útiles y sólidos, por medio de mecanismos de abstracción

Cualquier carrera está compuesta de una serie de cursos entrelazados porque algunos deben ser impartidos antes que otros. Incluir los ejes transversales en los cursos de la carrera no solo es conveniente, sino que ayuda a definir mejor el plan de estudios. Por eso, al usar la taxonomía (CS2008) conviene combinarla con [Arias-2006].

5. PROGRAMACIÓN I

Un ejercicio relativamente sencillo de comprender, o de realizar, es obtener la definición de un curso basado el temario ACM tomando como punto de inicio el temario actual del curso. En este ejemplo se toma el curso CI-1101 Programación I [DiMare-2010b]:

`http://www.di-mare.com/adolfo/cursos/2012-1/ci-1101.htm`

Objetivo

Proveer al estudiante la formación básica en programación para su adecuado desempeño en los cursos subsiguientes del área de programación, fomentando en el estudiante habilidades generales para la resolución de problemas de programación.

Contenidos

- Nociones básicas de sistema operativo, arquitectura de un computador, lenguaje de programación, algoritmo.
- Conceptualización y definición de clases, atributos de clases, instancias de clases, tipos y variables.
- Entrada y salida de datos.
- Estructuras básicas de control: secuenciación, bifurcación, iteración.
- Conceptualización e implementación de métodos por medio de funciones (sin parámetros, con parámetros de valor, con parámetros de referencia).
- Distintos tipos de módulos: procedimiento, función, clase, programa, unidad (es decir, un conjunto de procedimientos o funciones y estructuras de datos).
- Estructuras de datos basadas en arreglos.
- Estructura de una clase: parte pública, parte privada, constructores y destructores.
- Funciones recursivas tales como: factorial, Fibbonacci, multiplicación de enteros, potencia de dos números, máximo común divisor de dos números, Torres de Hanoi; además, recorrido, inserción y borrado sobre árboles binarios,
- Esquemas genéricos de algoritmos de ordenamiento básicos, tales como: burbuja, selección e inserción. Idealmente, la implementación de estos algoritmos debería estudiarse en el contexto de arreglos básicos, así como de la clase lista.
- Algoritmos de búsqueda secuencial y búsqueda binaria.

- Diferencia entre memoria estática y memoria dinámica.
- Funcionalidad de clases contenedoras básicas: arreglos (unidimensionales y multidimensionales), lista, pila, cola, conjunto, árbol binario ordenado.
- Implantación de clases contenedoras básicas: arreglo, lista, pila, cola, conjunto y árbol binario ordenado.
- Nociones preliminares de herencia, polimorfismo, clases abstractas, funciones virtuales, abstracción, encapsulamiento y ocultamiento de información.

Al examinar este temario se pueden identificar los siguientes tópicos mencionados en (CS2008):

Programación I

OS/OverviewOfOperatingSystems	2
SDF/FundamentalProgrammingConcepts	10
SDF/FundamentalDataStructures	12
PL/ObjectOrientedProgramming	4
Redacción de especificaciones	1
Créditos totales: 2 = ceil(29/15)	29

Figura 5

y procedimientos de A veces es necesario incluir temas que no que no están mencionadas en (CS2008) como ocurre aquí con “Redacción de especificaciones”, que es un tema importante si se usa la metodología de enseñanza propuesta en [DiMare-2010a] para impartir el primer curso de programación. En este caso, basta agregar ese tema al temario general de contenidos.

El número que acompaña a cada unidad de conocimiento es la cantidad mínima de horas recomendada por ACM para impartir el tema, lo que en total suma 29 horas. Al aplicar la fórmula para obtener la cantidad de créditos hay que evaluar la expresión ceil(29/15) que resulta en un total de 2 créditos, pero si el curso tiene 4 quedan 31==60-29 horas de holgura para impartir el curso. También es importante definir los ejes transversales para este curso:

Programación I

XI/Capacidad de abstracción y síntesis	33%
XI/Capacidad de diseño y modelado	25%
XI/Capacidad de investigación	15%
XI/Capacidad verbal	10%
XI/Dominio de idiomas (inglés)	10%
XI/Capacidad crítica	25%
XI/Tenacidad y tolerancia	25%
XI/Imaginación y creatividad	25%
XI/Capacidad grupal	25%

Figura 6

Al incluir 10% para la capacidad “XI/Dominio de idiomas (inglés)” queda definido que en el curso CI-1101 cada estudiante debe adquirir hasta por lo menos el 10% del dominio del idioma que necesita para mostrar un desempeño decoroso como profesional.

El curso completo CI-1101 , que incluye el temario, los objetivos de aprendizaje, los ejes transversales, los temas adicionales y la bibliografía, aparece completo en el apéndice, el que no se reproduce aquí directamente por razones de espacio, pero que está disponible en este sitio [DiMare-2012]:

`http://www.di-mare.com/adolfo/p/acmcompu/`

seran producidos directamente de los manuscritos listos para camara tal como se recibe de los autores. Por lo tanto, los autores deben tratar de reproducir sus escritos tan parecido como se pueda a este modelo.

6. REFINAMIENTO DEL PROGRAMA

Agrupar las unidades de conocimiento ACM para obtener el contenido de los curso puede dejar muy cargados algunos cursos. Por ejemplo, en Programación I no hace falta impartir todos los tópicos del tema OS/OverviewOfOperatingSystems, en que se incluyen nociones sobre sobre “Asuntos de diseño (eficiencia, robustez, flexibilidad, portabilidad, seguridad, compatibilidad)” y sobre “Incluencia de la seguridad, redes, multimedios y ventanas”, que son temas importantes al estudiar sistemas operativos, pero que sobran en el primer curso de computación. La manera de lidiar con este pequeño escollo es trasladar a otros cursos los tópicos y los objetivos de aprendizaje que son muy avanzados. Por eso, en algunas ocasiones es necesario aumentar la granularidad con que se trabaja.

CS2008_UNIT

KNOWLEDGE_UNIT	HOURS
SDF/FundamentalDataStructures	12
PL/ObjectOrientedProgramming	4

Figura 7

Si se usa el identificador de cada unidad de conocimiento como una llave, se puede almacenar en una base de datos SQL toda la jerazquía del conocimiento ACM.

CS2008_DETAIL

KNOWLEDGE_UNIT	SC	T	LG	DESCRIPTION
SDF/FundamentalDataStructures	01	T	en	Arrays
SDF/FundamentalDataStructures	03	T	en	Strings and string processing
SDF/FundamentalDataStructures	06	T	en	Simple linked structures
OS/Concurrency	01	T	en	States and state diagrams
OS/Concurrency	04	T	en	The role of interrupts
OS/Concurrency	09	O	en	Explain conditions that lead to deadlock

Figura 8

Debido a que la taxonomía está escrita en inglés, es necesario traducirla a otros lenguajes, lo que obliga a almacenar el texto de los temas, sus tópicos y sus objetivos de aprendizaje, en una tabla relacional adicional. En la columna T se indica si el tuple corresponde a un tópico 'T' o a un objetivo de aprendizaje 'O'. El campo SC permite mantener la secuencia en que los tópicos y objetivos de aprendizaje aparecen en (CS2008), y el campo LG indice el lenguaje en que está escritala descripción: { en, es, it, etc. }. Por ejemplo, al traducir al español el primer renglón se obtendría el siguiente tuple:

< SDF/FundamentalDataStructures, 03, 'T', en, Cadenas y procesamiento de hileras >

La traducción al italiano resultaría en:

< SDF/FundamentalDataStructures, 01, 'T', it, Vettores >

Aunque esta base de datos es muy simple, es suficiente para definir los cursos usando la tabla CURSO en la que aparezca el código de la unidad de conocimiento junto con el valor SC que indica cuál es el tópico o el objetivo de aprendizaje incluir en curso. Si se quiere evitar que las modificaciones futuras a (CS2008) rompan la secuencia definida por el campo SC, se puede usar un identificador adicional de manera que la llave usada en la tabla CURSO no sea [KNOWLEDGE_UNIT+SC].

7. MATRIZ DE EJES TRANSVERSALES

Un ejercicio saludable es hacer una matriz en la que se contrasten cada una de las Competencias con los cursos, de manera que para cada curso CURSO(i) se quede definido el porcentaje COMPETENCIA(j) que le corresponde al curso. Al analizar una fila de la matriz se obitiene cuáles son las habilidades transversales a desarrollar en cada

curso, y al examinar cada columna se puede verificar si la competencia se cubre apropiadamente a lo largo de la carrera.

	Abst	Disñ	Grup
Pg1	33	25	25
Pg2	50	45	75
OS	100		100

Figura 9

En esta versión reducida de la matriz de competencias se muestra que en Pg1 cada estudiante debe obtener el 33% de su "Capacidad de abstracción y síntesis", en Pg2 debe haber adquirido el 60% y cuando termina OS ya debe contar con el 100%. Para la "Capacidad de diseño y modelado" se presume que al finalizar OS solo ha adquirido el 45% del desempeño que necesita, lo que muestra que hay una carencia curricular en cuanto a este eje transversal. Una forma de mostrar los porcentajes es usar el valor acumulado como se muestra aquí, aunque en algunas ocasiones puede ser más útil mostrarlos por separado.

8. METODOLOGÍA PARA MEJORAR LA CARRERA

Cuando se firman tratados entre países el mecanismo de aprobación que se usa es crear un pequeño equipo de trabajo encargado de redactar y negociar el documento a aprobar para luego someter a votación el documento previamente confeccionado. Por ejemplo, los tratados de libre comercio son redactados por especialistas enviados por cada gobierno, quienes se enfrascan en una profunda negociación. A fin de cuentas, ese pequeño grupo de trabajo produce un documento de consenso que es luego presentado al Poder Legislativo de cada país signatario, con una condición muy especial: cada asamblea legislativa puede aceptar o rechazar el documento, pero no puede modificarlo. La negociación no se da en cada órgano legislativo sino en el seno de la comisión redactora, pues los tratados son complicados y por eso modificarlos requiere de un gran esfuerzo intelectual, del que es incapaz cualquier cámara legislativa.

Cuando el plan de estudios de una carrera debe ser aprobado por un cuerpo colegiado es necesario abrir oportunidades de participación que le permitan a todos en la academia expresar su parecer. Como ese no es un asunto tan complicado como un tratado internacional, no hace falta usar el procedimiento de los tratados, pues conviene más usar un procedimiento menos excluyente. Pero, definitivamente, es necesario contar con un documento base que sirva para llegar al consenso académico que permita modificar una carrera.

El reporte (CS2008) se puede usar para construir el documento base que luego será visto y aprobado en el plenario académico. Debido a que (CS2008) es una taxonomía muy exhaustiva, permite lograr construir el documento base rápidamente, pues se puede tomar "147 temas" junto con "20 capacidades" y asignarlos a "25 cajitas" que representan cada uno de los cursos que conforman el plan de estudios. Este procedimiento puede ser automatizado, como lo ha demostrado el profesor Ernesto Cuadros de la Universidad Católica San Pablo, en Arequipa Perú, quien cuenta con un programa que recibe como insumo "temas y cajitas" y produce un Plan de Estudios completo, con base en las recomendaciones ACM [VC-2006].

Los docentes encargados de impartir cada curso son los llamados a proponer la primera versión del curso revisado, pues con una reunión pueden identificar el temario de su curso. Una vez que cada pequeño grupo de profesores ha preparado su propuesta de cursos, es posible hacer una o más reuniones plenarias para definir el plan de estudios completo. En resumen, la forma de proceder puede delinearse de esta manera:

- Reunión plenaria para definir el perfil del profesional que se quiere producir
- Designación de un coordinador general
- Designación del comité de secretariado y edición de los documentos
- Seminario taller para analizar los documentos ACM/IEEE disponibles en [ACM-2012b]:
  `http://acm.org/education/curricula-recommendations`
- Creación de un repositorio digital para almacenar la definición de cada curso
- Definición individual del programa de cada curso, labor realizada por cada docente con base en (CS2008)

- Trabajo grupal por todos los docentes de cada curso para refinar y unificar el temario
- Identificación de temas, tópicos y objetivos de aprendizaje faltantes y duplicados
- Reacomodo temático general de los cursos para colocar correctamente faltantes y duplicados
- Reuniones plenarias para revisar las propuestas de modificación de cada curso

La razón por la que es necesario definir el perfil profesional antes de embarcarse en la construcción o modificación de un plan de estudios es muy simple: hay que delimitar el rango profesional del futuro graduado. Si no se hace esta definición antes de todo lo demás, el resultado puede ser desastroso. Por ejemplo, puede ocurrir que se necesitan 2 carreras en lugar de solo 1, por lo que al juntar los requisitos de ambas resulta en un programa de estudios demasiado abultado. Esta primera definición debe incluir algunas otras consideraciones, en especial de calendarización, para evitar que el proceso se demore años o décadas.

Es importante definir una persona, o un grupo muy pequeño de personas, quienes se encarguen de hacer el trabajo de secretariado. También es posible utilizar herramientas colaborativas como [WIKI-2012] para esto, pero siempre quien coordina se encarga de agregarle inteligencia al repositorio digital de la propuesta.

Aunque es posible que varios docentes se tomen a pecho el encargo de estudiar las recomendaciones curriculares ACM, en la práctica es más rápido que quien coordine las exponga en un pequeño seminario de inducción, en la que también se puede mostrar cómo funciona el repositorio digital que funciona durante todo el proceso.

Los últimos pasos de esta metodología incluyen el trabajo en pequeños grupos para definir el contenido de cada curso, y unas pocas reuniones plenarias para aprobar el documento final.

9. EXPERIENCIA EN EL USO DE LA METODOLOGÍA

En la Escuela de Ciencias de la Computación e Informática [ECCI] de la Universidad de Costa Rica se ha aplicado esta metodología, pues al comenzar el ciclo lectivo se trató de lograr que los profesores presentaran el programa de su curso usando la taxonomía ACM, en formato digital. Desafortunadamente poco avance se logró con esta directriz.

En todo proceso de cambio importante siempre es fundamental que la dirección lo apoye como prioridad. Sin el compromiso de la administración superior es muy difícil concretar resultados que es lo que a fin de cuentas ha ocurrido en la ECCI, en donde la prioridad la tiene el proceso de acreditación del plan de estudios, por lo que muchos profesores han preferido postergar la discusión de los mecanismos para actualizar el plan de estudios de la carrera.

Debido a la forma en que se toman las decisiones académica en la ECCI, en donde se usa el consenso como mecanismo de decisión, es difícil concretar cambios pues cada profesor defiende su curso en la carrera, se opone a cambiarlo, y solo está dispuesto a aceptar cambios después de un proceso de duro convencimiento. En un ambiente tan politizado, cuesta formar coaliciones suficientemente fuertes para lograr que una mayoría acepta una nueva propuesta.

Otra barrera que tiene el uso de la taxonomía ACM es que para la mayoría es desconocida, por lo que no es hasta que se discute una propuesta específica que cada integrante de la asamblea de profesores conoce el detalle de una propuesta. Esto obliga a discutir el tema de cada curso en varias sesiones lo que atrasa mucho el proceso. También ocurre, como en toda organización humana, que algunos se oponen silenciosamente al cambio para evitar el trabajo que significa hacer de una forma distinta lo que se viene haciendo de una manera durante años.

El usar la taxonomía ACM puede ser una forma de aliviar la presión que cada profesor, en forma individual, ejerce sobre el proceso de mejora académica, pues al usar un temario que es bastante general pero muy completo, se aumenta la posibilidad de convencer a suficientes profesores para que acepten nuevas propuestas.

De todas formas, el conocer esta forma distinta de proceder sí ha ayudado pues, poco a poco, se van preparando los insumos que permitan preparar una versión inicial de la propuesta nueva para el plan de estudios, el que luego

será distribuido para su revisión por el plenario de la asamblea de profesores escuela. El proceso también es muy provechoso porque ayuda a definir la bibliografía de cada curso.

10. CONCLUSIONES

De la discusión anterior se deduce que usar una taxonomía como la ACM tiene las siguientes ventajas:

- Los cursos son más completos y quedan mejor especificados
- Queda más campo para innovar porque los programas no quedan demasiado detallados
- La definición ACM es más general pero al mismo tiempo cubre mejor todo el ámbito del conocimiento en computación
- Se puede hacer sugerencias para mejorar CS2008 en el proceso de usar el temario
- La construcción de cursos se reduce a reordenar el temario
- Aplicar esta metodología puede ayudar a romper el estancamiento en que cae una academia cuando las propuestas de planes de estudios deben ser aprobados por consenso. Copia de algunos de los documentos mencionados en este trabajo están disponibles aquí [DiMare-2012]:
  http://www.di-mare.com/adolfo/p/acmcompu/

11. AGRADECIMIENTOS

Alejandro Di Mare hizo muchas sugerencias que ayudaron a mejorar las primeras versiones de este trabajo. Las ideas expuestas en este trabajo nacen al escuchar al pionero latinoamericano en este campo, profesor Ernesto Cuadros Vargas, quien mostró su sistema automático para generar el plan de estudios de una carrera durante el XVIII Congreso Iberoamericano de Educación Superior en Computación [CIESC 2010] realizado en la Universidad Nacional de Asunción, Asunción, Paraguay, octubre 2010.
http://socios.spc.org.pe/ecuadros/

REFERENCIAS

[ACM-2001] Association for Computing Machinery: "CC 2001: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science", 2001.
http://acm.org/education/education/education/curric_vols/cc2001.pdf

[ACM-2008] Association for Computing Machinery: "CS2008 Curriculum Update: The Computing Curricula Computer Science Volume is complete and approved", 2008.
http://acm.org//education/curricula/ComputerScience2008.pdf

[ACM-2012a] Association for Computing Machinery: "Computer Science Curricula 2013 Strawman Draft (February 2012)", 2012.
http://ai.stanford.edu/users/sahami/CS2013/strawman-draft/cs2013-strawman.pdf

[ACM-2012b] Association for Computing Machinery: "Curricula Recommendations)", 2012.
http://www.acm.org/education/curricula-recommendations

[AK-2001] Anderson, L.W. (Ed.) & Krathwohl, D.R. (Ed.) & Airasian, P.W. & Cruikshank, K.A. & Mayer, R.E. & Pintrich, P.R. & Raths, J., & Wittrock, M.C.: "A taxonomy for learning, teaching, and assessing: A revision of Bloom's Taxonomy of Educational Objectives (Complete edition)", New York: Longman, 2001.

[BK-1956] Bloom, B.S. (Ed.) & Engelhart, M.D. & Furst, E.J. & Hill, W.H. & Krathwohl, D.R.: "Taxonomy of educational objectives: The classification of educational goals. Handbook 1: Cognitive domain", New York: David McKay, 1956.

[Arias-2006] Arias, Rodolfo: "Competencias generales, Clasificación Propuesta por Rodolfo Arias", Escuela de Ciencias de la Computación e Informática, Universidad de Costa Rica, 2006.
http://www.di-mare.com/adolfo/p/acmcompu/

[Churches-2009] Churches, Andrew: "Taxonomía de Bloom para la Era Digital", Eduteka 2009.
http://www.eduteka.org/TaxonomiaBloomDigital.php

[CONARE-1976] Consejo Nacional de Rectores, Costa Rica: "Convenio para Unificar la Definición de Crédito en la Educación Superior de Costa Rica", 1976.
http://cu.ucr.ac.cr/normativ/definicion_credito.pdf

[CONARE-2004] Consejo Nacional de Rectores, Costa Rica: "Convenio Para Crear una Nomenclatura de Grados y Títulos de la Educación Superior Universitaria Estatal", 2004.
http://cu.ucr.ac.cr/normativ/nomenclatura_grados_titulos.pdf
http://www.conare.ac.cr/ [Servicios] → [Leyes, Convenios y Decretos]

[DiMare-2010a] Di Mare, Adolfo: "Aprendizaje Java acelerado por casos de prueba JUnit", Artículo #22 del XVIII Congreso Iberoamericano de Educación Superior en Computación [CIESC 2010] realizado en la Universidad Nacional de Asunción, Asunción, Paraguay, octubre 2010.
http://www.di-mare.com/adolfo/p/JUnit6d.htm

[DiMare-2010b] Di Mare, Adolfo: "CI-1101 Programación I", II Semestre 2010, Escuela de Ciencias de la Computación e Informática Universidad de Costa Rica.
http://www.di-mare.com/adolfo/cursos/2010-2/ci-1101.htm

[DiMare-2012] Di Mare, Adolfo: "Recopilación de Materiales de la Taxonomía ACM", 2012
http://www.di-mare.com/adolfo/p/acmcompu/

[Ferrer-2003] Ferrer Torres, Ramón A.: "Lista de verbos que se pueden utilizar para expresar objetivos de tipo cognoscitivo", Universidad Estatal de Venezuela, Facultad de Humanidades y Educación, 2003.
http://www.scribd.com/doc/29679486/

[Schmidt-2006] Schmidt M., Sandra: "Competencias, Habilidades Cognitivas, Destrezas Prácticas y Actitudes - Definiciones y Desarrollo", Universidad Tecnológica de Chile, Instituto Profesional, Centro de Formación Técnica, 2006.
http://www.scribd.com/doc/51265621/

[VC-2006] Vidal, Elizabeth & Cuadros Vargas, Ernesto: "Computer Science Curricula design for peruvian universities: San Pablo Catholic University case study", e Proceedings of the 1st IFIP International Conference on Education for the 21st century -- Impact of ICT and Digital Resources, 2006.
http://socios.spc.org.pe/ecuadros/papers/WCC-TC3.pdf

[WIKI-2012] WikiPedia: "WIKI", 2012.
http://es.wikipedia.org/wiki/Wiki

Autorización y Renuncia

Los autores autorizan a LACCEI para publicar el escrito en las memorias de la conferencia. LACCEI o los editores no son responsables ni por el contenido ni por las implicaciones de lo que esta expresado en el escrito