Cuaderno

ACTIVA

Tecnológico de Antioquia
Institución Universitaria

Educación sin Fronteras

CUADERNO ACTIVA
Facultad de Informática

ISSN: 2027-8101

Edición:
Tecnológico de Antioquia - Institución Universitaria
Calle 78B No. 72A-220
www.tdea.edu.co

Corrección de estilo:
Alexander Arbey Sánchez Upegui

Diseño e impresión:
Litografía Dinámica
Medellín - Colombia
Marzo de 2011

MISIÓN

En el Tecnológico de Antioquia – Institución Universitaria – formamos personas comprometidas con el desarrollo del departamento y del país, en los ciclos de formación técnica profesional, tecnológica, profesional universitario y de formación avanzada; desde un Proyecto Educativo Institucional que potencializa la instrucción de conocimiento, fomenta el espíritu humanista, crítico e investigativo, la responsabilidad social y el desarrollo sostenible.

VISIÓN

El Tecnológico de Antioquia en el 2015 será identificado como una Institución Universitaria, líder en el orden departamental, competitiva en el ámbito nacional, con proyección internacional, reconocida por la excelencia académica e investigativa y la calidad humana de sus integrantes, para responder con eficiencia, eficacia, pertinencia y compromiso social a los requerimientos y necesidades de la sociedad en el marco de la interrelación empresa, institución y comunidad.

VALORES

AUTONOMÍA:
Actitud de valerse por sí mismo, usando responsablemente la propia libertad, sin aislarse o separarse del contexto social.

LIDERAZGO:
Capacidad de asumir la responsabilidad de conducir a otros al efectivo logro de sus fines personales y/o colectivos, influyendo en ellos, compartiendo valores, convicciones, ejemplaridad, creatividad, espíritu de iniciativa y de servicio, comunicación afectiva y efectiva, trabajo en equipo y valores éticos.

TOLERANCIA:
Comprensión y respeto a los demás, a sus ideas, culturas, credos, prácticas y sentimientos, capacidad de escuchar y aceptar a los demás, comprendiendo el valor de las distintas formas de entender la vida.

RESPETO:
Actitud de comprensión del ser de los demás, que nos permite entender su actuación y portarnos con cordura y tolerancia frente a ellos, reconociendo que algo o alguien tiene valor, sustentado en la moral y la ética.

HONESTIDAD:
Conducta humana que consiste en comportarse y expresarse con coherencia y sinceridad, y de acuerdo con los valores de la verdad y la justicia.

RESPONSABILIDAD:
Conducta para cumplir las obligaciones adquiridas, dando respuestas adecuadas a lo que se espera de una persona, empresa, institución, grupo o sociedad.

COMPROMISO SOCIAL:
Actitud de dar respuesta, desde el conocimiento, a las necesidades del entorno social coadyuvando al mejoramiento de la calidad de vida.

CONTENIDO

EDITORIAL

El Cuaderno ACTIVA de la Facultad de Informática del Tecnológico de Antioquia Institución Universitaria nació con el objetivo de brindar un espacio académico, investigativo y científico en los campos de la Informática, la Electrónica y las áreas referentes a las tecnologías de la información y comunicación. En el Cuaderno ACTIVA se busca la contribución de todos los miembros de la comunidad académica del Tecnológico de Antioquia, así como la participación de otras instituciones educativas y del sector productivo.

La edición actual cuenta con artículos referentes a los campos de la ingeniería de software, la calidad del software, los procesos investigativos, así como acciones adelantadas por la Facultad en procura de fortalecer el vinculo Estado-Empresa-Universidad. En esta edición participan investigadores del Tecnológico de Antioquia del grupo de investigación GIISTA e investigadores de la Universidad Nacional con su grupo de Investigación en Ingeniería de Software.

También, el Cuaderno ACTIVA referencia las memorias del "I Seminario Internacional de Ingeniería de Software Tecnológico de Antioquia", a través de algunos de los artículos más importantes que hicieron parte de dicho evento.

Con estos grandes activos académicos e investigativos, el cuaderno ACTIVA traza una nueva etapa de consolidación y de apertura de nuevos públicos y referentes intelectuales.

Fabio Alberto Vargas Agudelo
Decano

ARTÍCULO I

EXPERIENCIA INVESTIGATIVA CON LOS PROYECTOS MIPSOO Y SISMOO

Ricardo de Jesús Botero Tabares
Profesor de Tiempo Completo, Facultad de Informática
Tecnológico de Antioquia – Institución Universitaria
rbotero@tdea.edu.co

RESUMEN

Este artículo relata la experiencia investigativa con dos proyectos de investigación desarrollados por el grupo GIISTA de la Facultad de Informática. El primero, *Método Integrado de Programación Secuencial y Programación Orientada a Objetos para el análisis, diseño y elaboración de algoritmos -MIPSOO*, propone un método de aprendizaje de la programación orientada a objetos que comprende cuatro pasos: definición de la tabla de requerimientos, diseño del diagrama de clases, definición de las responsabilidades de las clases y escritura del seudocódigo. El segundo proyecto,*Sistema para el Modelamiento por Objetos-SISMOO,* implementa un entorno integrado de desarrollo que permite editar, compilar y ejecutar seudocódigo orientado a objetos.

Palabras clave: Aprendizaje basado en problemas, Aprendizajede la programación, Programación orientada a objetos.

INTRODUCCIÓN

Los procesos de aprendizaje en programación de computadoras deben conllevar la aplicación de paradigmas y modelos acordes con los recientes avances tecnológicos en ingeniería de software, en respuesta a las necesidades del sector productivo.

Las nuevas versiones de entornos integrados y lenguajes para el desarrollo de software se apoyan en paradigmas de programación que garantizan un producto final confiable, eficiente y fácil de utilizar[1]. Estas versiones llegan a las universidades y empresas desarrolladoras de software, donde se deben adaptar y comprender de la mejor manera posible para lograr los altos niveles de competitividad que exige una sociedad globalizada e interconectada. Por esto, las instituciones de educación superior que ofrecen programas de tecnología en sistemas, ingeniería en software u otros afines, deben idear mecanismos que propicien, desde los primeros niveles de formación, espacios para fortalecerconceptos esenciales en la formación de ingenieros de software con una visión de futuro cimentada en la sociedad del conocimiento, con una alta capacidad de apropiación e innovación de la tecnología informática.

El aprendizaje de la lógica de programación para resolución de problemas, ligado al proceso pedagógico paralelo implícito, ha sido influenciado por pa-

radigmas que iniciaron con el modelo de la programación libre (con su perogrullado *go to*), pasaron por la diagramación estructurada (programación imperativa con diseño *top-down*) y han desembocado en la programación orientada a objetos (desarrollo basado en componentes con una interfaz pública que posibilita su reutilización). Se observan ahora marcadas tendencias hacia la programación orientada a aspectos y a servicios.

1. ANTECEDENTES

Desde el año 2004 algunos profesores de la Facultad de Informática del Tecnológico de Antioquia que impartían las asignaturas de Algoritmos, Estructuras de Datos, Lenguajes de Programación I e Ingeniería de Software, manifestaron su preocupación porque los contenidos de estas asignaturas giraban en torno a la programación estructurada, cuando las versiones de los productos (compiladores, intérpretes y herramientas CASE) soportaban el paradigma orientado a objetos. Esta contingencia llevó a un grupo de docentes a proponer cambios curriculares en las asignaturas citadas y a idear un método que unificara criterios para la enseñanza y el aprendizaje de los fundamentos de programación con orientación a objetos, mediante la aplicación de un seudolenguaje y estrategias pedagógicas propias del aprendizaje significativo y basado en problemas. Dado que el cambio curricular se hacía lento por las discusiones académicas que la propuesta suscitaba, se presentó al Comité para el Desarrollo de la Investigación del Tecnológico de Antioquia-CODEI,el proyecto de investigación MIPSOO, propuesta realizada por los profesores Eucario Parra, Carlos Castro y Ricardo Botero. Finalizado este proyecto, se unió al grupo el profesor Gabriel Taborda, para fortalecer el grupo de trabajo que consolidó otro proyecto: SISMOO. En ambos proyectos fue fundamental el apoyo de los egresados de la Facultad de Informática Miguel Valencia y Juan David Maya.

2. EL PROYECTO MIPSOO

El método para el aprendizaje y la enseñanza de la programación orientada a objetos del proyecto MIPSOO incluye elementos didácticos del aprendizaje basado en problemas y principios pedagógicos constructivistas; propone un seudolenguaje que toma características asociadas a lenguajes de programación de la familia C/ C++/ Java y al Lenguaje de Modelado Unificado (UML); contiene elementos sintácticos que posibilitan la integración entre los paradigmas de programación estructurada y orientada a objetos, y apoya en gran medida la enseñanza temprana de este último. Además,el proyecto tiene una justificación real basada en los resultados de encuestas a profesores, estudiantes y administrativos de diversas instituciones de educación superior del departamento de Antioquia - Colombia; y una justificación conceptual basada en componentes pedagógicos del modelo constructivista.

El proyecto MIPSOO armoniza con otras experiencias investigativas nacionales [2], [3] y extranjeras [4]; su didáctica implícita propone el seguimiento de cuatro fases para la solución de problemas:

- Definición de la tabla de requerimientos.
- Diseño del diagrama de clases.
- Definición de las responsabilidades de las clases.
- Escritura del seudocódigo (tiene una gran similitud con los lenguajes de programación Java, C# y Visual Basic.Net).

Cada etapa conlleva una serie de formalismos que se explican a continuación.

a) Definición de la tabla de requerimientos

La construcción de la tabla de requerimientos [5] forma parte del análisis del problema. Los requerimientos hacen referencia a las necesidades de los usuarios, es decir, identifican los aspectos que los usuarios del programa desean resolver mediante software. Estos requerimientos se denominan *funcionales* al sostener una relación directa con la funcionalidad del sistema.

La tabla de requerimientos está compuesta por cuatro columnas (ver tabla 1):

- **Identificación del requerimiento**: es un código que identifica al requerimiento, generalmente compuesto por una letra y un dígito. Identificadores comunes para los requerimientos son R1, R2, R3, etc.
- **Descripción**: consiste en una descripción concisa y clara, en lenguaje natural, del requerimiento.

- **Entradas**: son los insumos o datos necesarios para que el requerimiento se pueda suplir con éxito.
- **Resultados o salidas**: constituyen el cumplimiento del requerimiento, es decir, son los resultados que dan solución a un requerimiento funcional definido por el usuario.

Tabla 1. Identificación de requerimientos.

Identificación del requerimiento	Descripción	Entradas	Resultados (Salidas)

Es común que la salida de un requerimiento se convierta en la entrada de otro; los requerimientos funcionales son *casos de uso* que describen de una manera detallada el comportamiento del sistema con los distintos actores que interactúan con él. Un caso para citar es la descripción de un conjunto de secuencias de acciones que un sistema ejecuta y que produce un resultado observable de interés para un actor particular [6]. Un actor es un usuario del sistema.

b) Diagrama de clases

La abstracción de clases también forma parte del análisis del problema y es el primer momento para diseñar su solución. Consiste en una representación gráfica del problema, plano de software, donde se dibujan abstracciones de la realidad relacionadas con el mundo del problema, modelables con software. El plano construido se puede presentar en dos fases, que comprenden un *diagrama conceptual* y un *diagrama de clases*. Es de aclarar que el primero de ellos es opcional en tanto no se requiere cuando los problemas a tratar son pequeños o con cierto margen de trivialidad: mostrar un mensaje, comparar dos cadenas, hallar una sumatoria, entre otros. Un diagrama de clases típico se presenta en la figura 1.

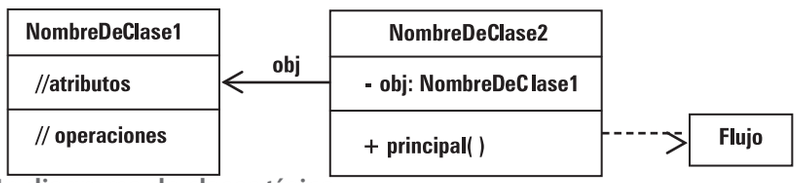

Figura 1. Un diagrama de clases típico.

c) Definición de responsabilidades de las clases

El análisis de responsabilidades de las clases conlleva la descripción de los métodos de cada clase mediante contratos que incluyen los requerimientos asociados, la precondición o estado del objeto antes de ejecutar el método, la postcondición que aclara el estado del objeto luego de ejecutar el método, y el modelo verbal que consiste en una descripción en lenguaje natural de la solución planteada, algo similar al denominado algoritmo cualitativo.

La identificación de responsabilidades forma parte de la documentación de la solución o del futuro sistema basado en software.

Tabla 2. Esquema de un contrato.

Nombre del método	Requerimientos asociados	Precondición	Postcondición	Modelo verbal

d) Escritura del seudocódigo orientado a objetos

El *seudocódigo orientado a objetos (OO)* especifica la solución del problema, pues da cuenta del cómo obtener una solución. Lo hace de una manera similar al proceso de codificación en un lenguaje de programación como Java o C#.

Esta fase conlleva la aplicación de pruebas manuales para cada uno de los métodos (similar a las denominadas *pruebas de escritorio*), o de manera automática con el traductor SISMOO.

Los tipos de datos estándar o predefinidos que se pueden usar en la escritura de seudocódigo, se observan en la tabla 3.

Tabla 3. Valores de inicialización por omisión para los tipos estándar de datos.

Tipo	Valor con el que se debe inicializar	Valor de inicialización por omisión (con descripción)
Numérica (**entero, real**)	Cualquier número (depende del problema)	0 (cero)
carácter	Cualquier carácter	' ' (espacio)
cadena	Cualquier cadena	" " (cadena vacía)
lógico	**falso** o **cierto**	**cierto** (verdadero)
Objeto	**nulo**	**nulo**

Otros problemas resueltos y propuestos donde se aplica el método con las cuatro fases se presentan en *Lógica y programación orientada a objetos: un enfoque basado en problemas* [7], texto que se ha constituido en la guía del módulo *Desarrollar Pensamiento Analítico Sistémico I*, dentro del plan de estudios 72 de Tecnología en Sistemas del Tecnológico de Antioquia.

3. EL PROYECTO SISMOO

Como complemento de MIPSOO, SISMOO es un proyecto de investigación aplicada, cuyo producto de desarrollo es una herramienta de software diseñada para que el usuario pueda editar, compilar y ejecutar los problemas diseñados empleando el seudolenguaje propuesto en MIPSOO. Esta herramienta, denominada en el medio informático *Entorno Integrado de Desarrollo o IDE* (*Integrated Development Environment),* fue desarrollada en lenguaje Java y su diagrama estructural se asemeja al de cualquier compilador (ver figura 2).

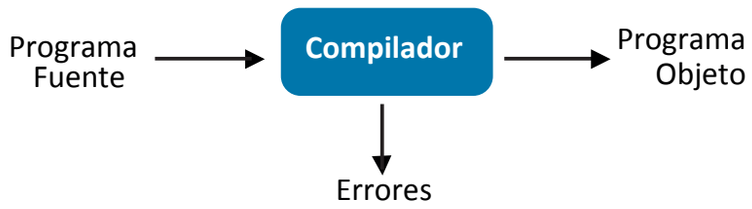

Figura 2. Esquema general de un compilador.

4. MENCIONES Y LOGROS

Los proyectos MIPSOO y SISMOO han logrado una mención nacional y un premio internacional. En Colombia los profesores Carlos Castro, Eucario Parra y Ricardo Botero, en representación de la Universidad de San Buenaventura (Medellín), la Fundación Universitaria Católica del Norte y el Tecnológico de Antioquia, respectivamente, sustentaron en el *VI Encuentro Iberoamericano de Instituciones de Enseñanza de la Ingeniería* [8], el póster de la figura 3, titulado *Método de aprendizaje en fundamentos de programación con orientación a objetos*, por lo cual les otorgaron una Mención Especial, expuesta en la figura 4.

Figura 3. Póster presentado en el VI Encuentro Iberoamericano de Instituciones de Enseñanza de la Ingeniería, organizado por ACOFI.

También, para el *Sexto Simposio Iberoamericano en Educación, Cibernética e Informática*-SIECI 2009 [9], en el contexto de la *Octava Conferencia Iberoamericana en Sistemas, Cibernética e Informática*- CISCI 2009, realizado entre el 10 y 13 de Julio de 2009 en Orlando, Florida - EE.UU, se presentó el artículo titulado M*étodo y entorno integrado de desarrollo para el aprendizaje en lógica de programación orientada por objetos* [10], por el cual los profesores Carlos Castro, Gabriel Taborda y Ricardo Botero obtuvieron el premio al Mejor Artículo Sesión (figura 4).

Figura 4. Reconocimientos en ACOFI 2007 y en SIECI 2009.

5. UN PROBLEMA RESUELTO CON EL MÉTODO PROPUESTO

"La famosa ecuación de Einstein para conversión de una masa *m* en energía, viene dada por la fórmula $E = mc^2$, donde *c* es la velocidad de la luz, *c* = 2.997925 x 1010 m/s. Leer la masa de un objeto en gramos y obtener la cantidad de energía producida en ergios".

La solución a este problema se presenta siguiendo las cuatro etapas planteadas en MIPSOO:

a) Identificación de requerimientos

Se identifican dos requerimientos donde se observa que la entrada del requerimiento R2 es la salida del requerimiento R1 (ver tabla 4).

Tabla 4. Requerimientos para el problema de la ecuación de Einstein.

Identificación del requerimiento	Descripción	Entradas	Salidas
R1	Conocer la masa del objeto en gramos.	Un número real digitado por el usuario.	La masa del objeto está almacenada en memoria.
R2	Calcular la cantidad de energía producida por un objeto.	La masa del objeto (en gramos).	La energía del objeto (en ergios).

b) Diseño del diagrama de clases

El diagrama de clases de la figura 5 conlleva la definición de las abstracciones *Energía* y *Proyecto*, y a la reutilización de las clases de uso común *Flujo* y *Mat*.

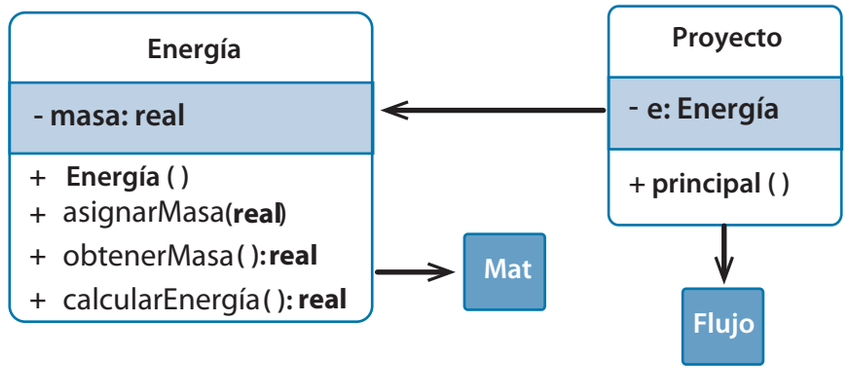

Figura 5. Diagrama de clases para el problema de la ecuación de Einstein.

c) Especificación de las responsabilidades de las clases

Las responsabilidades de las clases se expresan mediante los contratos de cada uno de sus métodos. En términos generales, la clase *Energía* es responsable de almacenar la masa del objeto y calcular su energía; la clase *Proyecto* es responsable de establecer comunicación con el usuario para la captura de la masa del objeto, crearlo, asignarle un estado y visualizar resultados para cumplir con los requerimientos. Las tablas 5 y 6 presentan los contratos de las clases de uso no común *Energía* y *Proyecto*.

Tabla 5. Contratos de la clase Energía.

Método	Requerimiento asociado	Precondición	Poscondición
Energía	R1	No existe un objeto para calcularle su energía.	Existe un objeto en memoria listo para ser procesado.
asignarMasa	R1	El objeto tiene una masa igual a 0 (cero)	El objeto tiene una masa (número real positivo) igual a la especificada por el usuario.
obtenerMasa	R2	El objeto tiene una masa distinta de cero, desconocida para el mundo exterior al objeto.	El mundo exterior al objeto conoce la masa del objeto.
calcularEnergía	R2	Se desconoce la energía producida por el objeto.	Se conoce la energía producida por el objeto.

Tabla 6. Contratos de la clase Proyecto.

Método	Requerimiento asociado	Precondición	Poscondición
principal	R1 y R2	No hay entradas de datos y no se ha efectuado proceso alguno.	Se tiene un objeto en memoria con una masa definida y una energía conocida.

Las responsabilidades de las clases de utilidad *Flujo* y *Mat*, conocidas también como clases de uso común, no se definen de forma explícita porque se asumen comprendidas por el analista; *Flujo* se responsabiliza de las operaciones de entrada y salida estándar y *Mat* de las operaciones con funciones matemáticas.

d) Escritura de seudocódigo

El seudocódigo guarda similitud con la sintaxis de lenguajes de producción como Java y Visual Basic.Net; es un buen preámbulo a la etapa de codificación en alguno de estos u otros lenguajes. La figura 6 ilustra el seudocódigo para este problema, donde las palabras reservadas del seudolenguaje se escriben en negrita, en contraposición a los demás identificadores para nombres de clase – *Energía* y *Proyecto* –, atributos – *masa* y *e* –, métodos –*Energía( )*, *asignarMasa( )*, *obtenerMasa( )* y *calcularEnergía( )*– , variables locales – *ener* – y objetos –*e*.

```
clase Energía
   privado real masa
   público Energía( )
   fin_metodo
   //-------------------------------
   asignarMasa(real m)
      masa = m
   fin_metodo
   //-------------------------------
   real obtenerMasa( )
      retornar masa
   fin_metodo
   //-------------------------------
```

```
    //------------------------------
    real calcularEnergía( )
        real ener
        const real C = 2.997925 * Mat.elevar  (10, 10)
        ener = masa * Mat.elevar(C,  2)
        retornar  ener
    fin_metodo
fin_clase
//-------------------------------------------------------------------------------------
clase Proyecto
    Energía  e
    estatico principal( )
        e = nuevo Energía( )
        Flujo.imprimir  ("Ingrese  la masa del objeto en gramos:")
        real ms = Flujo.leerReal( )
        e.asignarMasa(ms)
        Flujo.imprimir("Energía  producida  =  " + e.calcularEnergía( ) + " ergios")
    fin_metodo
fin_clase
```

Figura 6. Seudocódigo para el problema de la ecuación de Einstein.

6. CONCLUSIONES

- Los cambios curriculares en el área específica de los programas de ingeniería de sistemas y afines, se deben promover desde los primeros niveles de formación, para garantizar avances significativos en un menor tiempo en los niveles intermedios y avanzados.
- Los componentes pedagógicos, didácticos, cognoscitivos, son fundamentales al momento de iniciar estudios de introducción a la programación. El constructivismo y la didáctica con elementos del aprendizaje por descubrimiento, significativo y basado en problemas, pueden garantizar un proceso cognitivo exitoso.
- El currículo, la didáctica y la pedagogía para el aprendizaje de la programación de computadores, debe sustentarse sobre reflexiones serias que conduzcan a metodologías inspiradas en las perspectivas de la ingeniería de sistemas, las tecnologías dominantes de la informática, la aproximación a problemas reales, el rigor abstracto y lógico y la puesta en común entre las necesidades expresas por el sector productivo, las expectativas de los estudiantes y las propuestas de los docentes.
- El paradigma de programación orientado a objetos no debe esperar a saltos curriculares. De esta forma, la abstracción de objetos y comportamientos debe hacer parte de la modelación desde los inicios del proceso de aprendizaje, en los cursos de lógica de programación y estructuras de datos, de tal manera que se traten de forma natural en otras asignaturas relacionadas con lenguajes de programación, bases de datos e ingeniería de software.
- Un enfoque constructivista crea variadas estrategias de enseñanza-aprendizaje: el *aprendizaje por descubrimiento*, donde la nueva información o conocimiento se adquiere a través de los propios esfuerzos del estudiante (tiempo de trabajo independiente), con los aportes del aprendizaje por exposición o instrucción impartido por el docente (horas de trabajo presencial); el *aprendizaje colaborativo*, donde el estudiante puede retroalimentar sus conocimientos con los colegas de su grupo o con comunidades externas contactadas a través de la web; el *aprendizaje significativo*, donde el aprendizaje del alumno depende de la estructura cognitiva previa que se relaciona con la nueva información; y el *aprendizaje basado en problemas*, que incluye características de los ambientes antes citados, donde tanto la adquisición de conocimientos como el desarrollo de habilidades y actitudes resulta importante, dado que un grupo pequeño de alumnos se reúne, con la facilitación de un tutor, a estudiar y resolver un problema seleccionado o diseñado especialmente para el logro de ciertos objetivos de aprendizaje.

REFERENCIAS BIBLIOGRÁFICAS

[1] Sommerville, I. Ingeniería del software. Madrid:Pearson Educación, 2005.

[2] Léxico en programación orientada a objetos. Consultado en agosto 10 de 2010. Disponible: http://ojarami.com/riosur.net/index.php

[3] Proyecto Cupi2, Universidad de los Andes. Facultad de Ingeniería, Departamento de Ingeniería de Sistemas y Computación. Consultado en junio 23 de 2010. Disponible: http://cupi2.uniandes.edu.co/sitio/

[4]Gayo, D.et al. Reflexiones y experiencias sobre la enseñanza de la programación orientada a objetos como único paradigma. Actas de las IX Jornadas de Enseñanza Universitaria de Informática. Cádiz. Julio de 2003. Consultado en abril de 2007. Disponible: http://www.di.uniovi.es/~dani/publications/jenui03.pdf

[5] Villalobos, J. y Casallas, R. Fundamentos de programación: aprendizaje activo basado en casos. México: Pearson Educación, 2006.

[6] Booch, G., Rumbaught, J. y Jacobson, I. El Proceso unificado de desarrollo de software. Madrid: Pearson Educación, 2000.

[7] Botero, R., Castro, C., Taborda, G., Valencia, M. y Maya, J. Lógica y programación orientada a objetos: un enfoque basado en problemas. Grupo GIISTA. Medellín: Divegráficas, 2009.

[8] XXVII Reunión Nacional y VI Encuentro Iberoamericano de Instituciones de Enseñanza de la Ingeniería. El profesor de ingeniería, profesional de la formación de ingenieros. Cartagena de Indias: ACOFI, 2007.

[9] Sexto Simposio Iberoamericano en Educación, Cibernética e Informática- SIECI 2009, Orlando, Florida - EE.UU. Disponible: http://www.iiis.org/CDs2008/CD2009CSC/CISCI2009/

[10] Castro, C., Taborda, G., Botero, R. (2009). Método y entorno inte¬grado de desarrollo para el aprendizaje en lógica de programación orien¬tada por objetos. Revista Iberoamericana de Sistemas, Cibernética e In¬formática. [En línea]. 6, (2). Disponible: http://www.iiisci.org/journal/risci/

ARTÍCULO II

INTRODUCCIÓN A LOS MODELOS DE CALIDAD DEL SOFTWARE BASADOS EN PROCESOS

Darío Enrique Soto Durán
Profesor de Tiempo Completo, Facultad de Informática
Tecnológico de Antioquia - Institución Universitaria
dsoto@tdea.edu.co

Adriana Xiomara Reyes Gamboa
Profesora de Tiempo Completo, Facultad de Ingeniería
Politécnico Colombiano Jaime Isaza Cadavid

RESUMEN

En un mundo globalizado, en donde las organizaciones se ven enfrentadas a competencias de orden mundial, la calidad, además de aumentar la satisfacción general del cliente, disminuir costos y optimizar los recursos, se convierte en un importante punto diferenciador. Los productos o servicios que ostentan certificados de calidad son preferidos por los compradores porque trasmiten seguridad y confianza. Esto también constituye un atributo de valor para las estrategias de comercialización en el exterior. Si bien la industria del software es nueva, ha tenido que madurar rápidamente, tal como lo exigen los avances tecnológicos y su alta participación al interior de las empresas. Esta industria comparte con las demás, el interés por la calidad y la competitividad. Lo anterior no debe ser ajeno a la academia, por el contrario, esta debe incorporar estas temáticas dentro de sus procesos de formación en los programas académicos del área tecnológica, con el fin de ofrecer profesionales competitivos en el país. Para esto, debe retomar y evaluar experiencias exitosas empleadas en otros países y aplicar las más viables en nuestro contexto.

Palabras clave: Calidad, Procesos, Software, Organización.

1. EVOLUCIÓN DE LA CALIDAD DEL SOFTWARE

A finales de la década de los sesenta se acuñó el término *Crisis del software* para referirse a la dificultad de construir programas libres de defectos, fácilmente comprensibles y que fueran verificables. La *crisis del software* resumió una serie de problemas que se venían observando en los procesos de desarrollo de software, tales como: los proyectos no terminaban en el plazo establecido, no se ajustaban al presupuesto inicial, baja calidad del software generado, software que no cumplía las especificaciones y código difícil de mantener que obstaculizaba la gestión y evolución del proyecto, entre otros.

A partir de este momento crucial en la evolución del desarrollo de software, nace formalmente la Ingeniería de Software y con ella la gestión de la calidad del software.

Al hablar de gestión de la calidad del software[1] es necesario incluir los principales factores que intervienen en el desarrollo del mismo: las personas, los procesos, el producto y el proyecto. Dentro de ellos, un factor

crítico son las personas, pues son ellas quienes finalmente logran el producto software, como resultado tangible del proceso de desarrollo.

En el ámbito internacional del software, a partir de la segunda mitad de los años ochenta, surgieron metodologías, modelos y estándares que abordan cada uno de los factores que intervienen en el proceso de desarrollo, cuya adopción muestra inicialmente un énfasis en aquellos orientados a los procesos, el producto y el proyecto. Un poco más tarde, surgen metodologías orientadas a medir los niveles de calidad de los procesos a los productos de software.

Al dar una mirada a la historia formal del aseguramiento de la calidad del software en Colombia, encontramos hitos importantes como la adopción de modelos para sistemas de gestión de la calidad y de mejoramiento del proceso de desarrollo de software, a partir de iniciativas del Estado y de la empresa privada; asimismo, de modelos de calidad de producto y de evaluación del mismo, con logros significativos a nivel organizacional, de proyecto y de producto, incluyendo certificaciones en diferentes niveles.

Con relación al factor persona, específicamente al desarrollador de software, a pesar de ser crítico para lograr el éxito de los proyectos de desarrollo, no se conocen iniciativas (tanto en el sector productivo como en el ámbito educativo) en cuanto a la adopción de modelos o metodologías orientados a mejorar sus competencias en la estimación y planeación de su trabajo, definir compromisos que se puedan cumplir, administrar la calidad de su trabajo y reducir el número de defectos en sus productos. Modelos como estos se han adoptado en otros países, como es el caso de la metodología Proceso de Software Personal y grupal denominada*Personal Software Process* - PSP) y *Team Software Process* - TSP respectivamente.

2. MODELOS DE CALIDAD ORIENTADOS AL PROCESO

Es bien sabido que para desarrollar software de calidad de manera consistente se requiere contar con una alta madurez de procesos. A nivel internacional el modelo de madurez de procesos más popular es denominado **Capability Maturity Model Integration**- CMMI**.** Sin embargo, este modelo es complejo para implementar en empresas pequeñas. La estrategia para incrementar la madurez en la industria de software, no contempla solamente los procesos de las empresas, sino que incluye el mejoramiento del elemento básico que da sustento a la industria: las personas. Precisamente en las personas se enfoca el Personal Software Process - PSP y a los equipos de desarrollo el Team Software Process - TSP, creados por el Dr. Watts Humphrey del Software Engineering Institute - SEI.

2.1 Proceso de Software Personal - PSP

El PSP es una línea de trabajo de medición y análisis para ayudarnos a caracterizar nuestro proceso. También es un procedimiento definido que nos ayuda a mejorar nuestro desempeño. Algunos de sus principios son [2]:

- La calidad de un sistema de software está dada por la calidad del proceso utilizado para desarrollarlo y mantenerlo.
- La calidad de un sistema de software está determinada por la calidad de sus componentes más deficientes.
- La calidad de un componente de software está dada por el individuo que lo desarrolla.
- El desempeño individual está dado por el conocimiento, la disciplina y el compromiso del individuo.
- Se pretende lograr que como profesional del software, conozcamos nuestro desempeño personal.
- Debemos medir, darle seguimiento y analizar el trabajo.
- Debemos aprender a partir de las variaciones del desempeño personal.
- Debemos incorporar las lecciones aprendidas en las prácticas personales.

La implementación del marco PSP, se realiza por fases como se muestra en la figura 7:

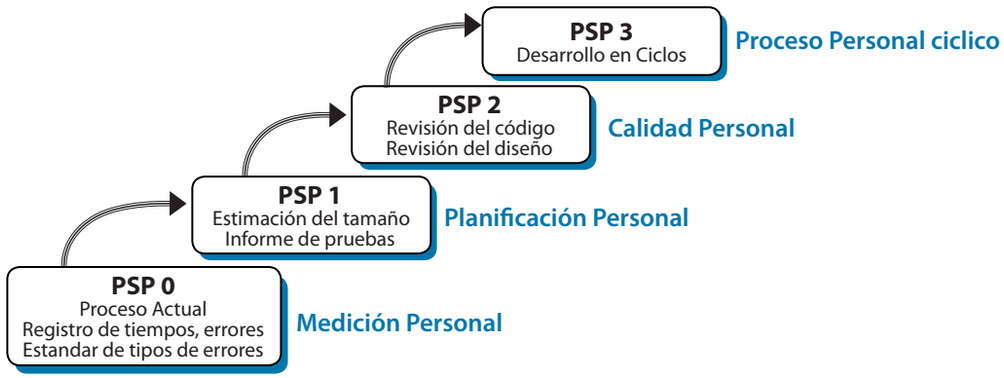

Figura 7. **Niveles del proceso personal de software.**

2.2 Proceso de Software en Equipo - TSP

El modelo TSP proporciona directrices para ayudar a un equipo a establecer sus objetivos, a planificar sus procesos y a revisar su trabajo con el fin de que la organización pueda establecer prácticas de ingeniería avanzadas y así obtener productos eficientes, fiables y de calidad.

El TSP provee un esquema de trabajo, donde cada desarrollador tiene perfectamente definidos sus roles, actividades y responsabilidades. Asimismo, el TSP incluye procedimientos para la mejora continua del proceso de desarrollo, para mejorar la calidad del software producido, para mejorar la estimación del tiempo de desarrollo, para la disminución de defectos en el producto y para promover la integración del equipo de desarrollo. Es decir, el TSP apoya tanto al equipo de desarrollo como a los administradores del proyecto para la culminaciónde proyectos de desarrollo de software, dentro del tiempo y presupuesto establecido (ver figura 8).

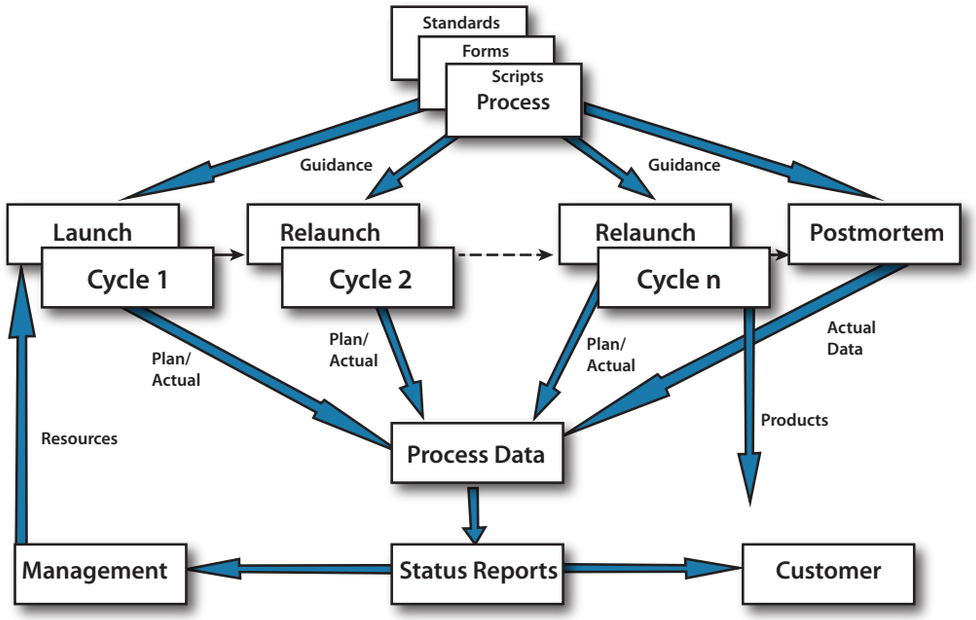

Figura 8. Flujo de proceso de TSP.

TSP se basa en los siguientes principios[3]:

- Un seguimiento preciso de un proyecto requiere planes bien detallados y datos precisos. Únicamente el personal que realiza el trabajo es capaz de recoger con precisión dichos datos.
- Para minimizar el tiempo del proyecto, los ingenieros deben equilibrar su carga de trabajo para maximizar la productividad, el primer foco de atención debe ser la calidad.
- TSP está formado por dos componentes primarios bien diferenciados que abarcan distintos aspectos del trabajo en equipo y que pueden observarse de manera resumida en la figura 9:

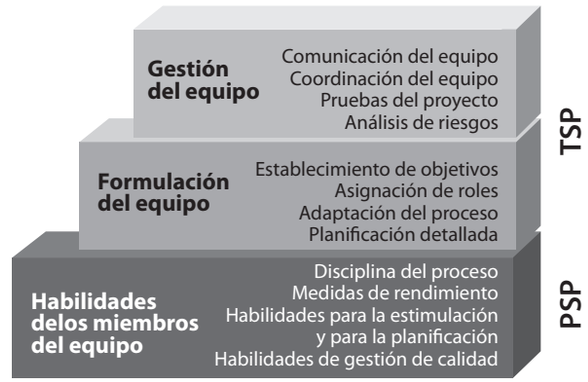

Figura 9. Aspectos de trabajo PSP y TSP.

2.3 CMMI

Este es un modelo para la mejora de procesos que proporciona a las organizaciones los elementos esenciales para procesos de desarrollo y mantenimiento de software. Durante los años 90, SEI desarrolló modelos para la mejora y medición de la madurez específicos para varias áreas[4]:

- CMM-SW: CMM for software
- P-CMM: People CMM.
- SA-CMM: Software Acquisition CMM.
- SSE-CMM: Security Systems Engineering CMM.
- T-CMM: Trusted CMM
- SE-CMM: Systems Engineering CMM.
- IPD-CMM: Integrated Product Development CMM.

Luego del uso y aplicación individual de estos modelos de madurez, el SEI desarrolló CMMI para facilitar y simplificar la adopción de forma simultánea de CMM-SW(CMM for Software), SE-CMM(SystemsEngineering Capability Maturity Model) e IPD-CMM(Integrated Product Development) y de ahí la palabra Integración en la sigla. Antes de CMMI el modelo más común era CMM-SW y se puede ver CMMI como la evolución de este último [5] (Figura 10).

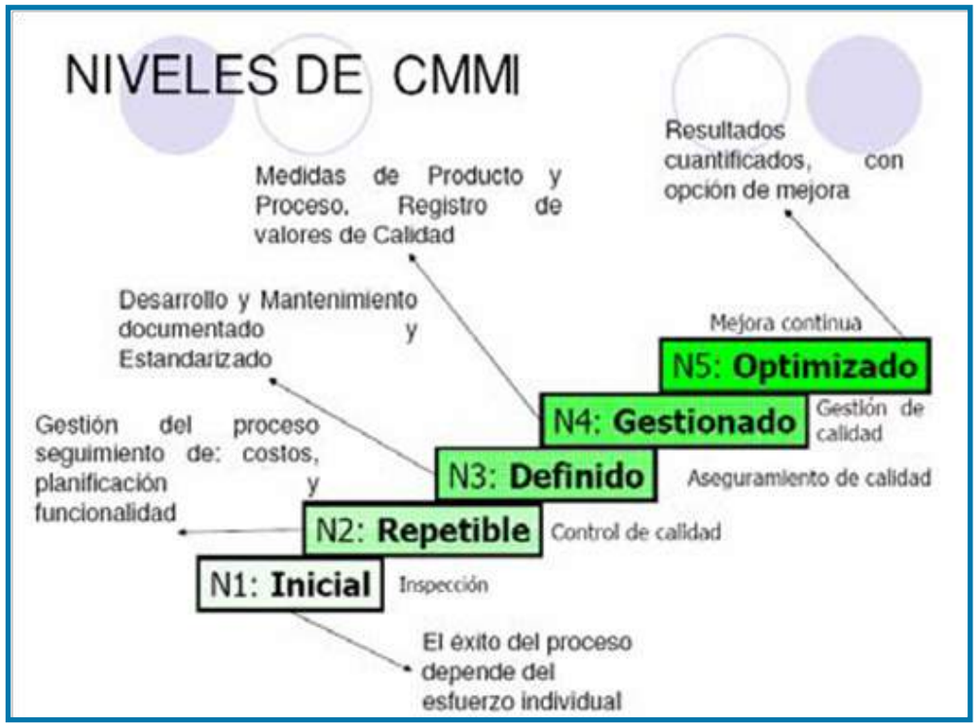

Figura 10. Niveles de madurez del modelo CMMI.

Este modelo presenta una estructura de cinco niveles de madurez, en los cuales una organización puede determinar su madurez en la producción de software en función de la consecución de los objetivos establecidos en cada nivel.

Según el nivel de madurez en que se encuentre la empresa, las medidas se enfocarán más al grupo de objetivos del nivel correspondiente, para que mejore la capacidad de producir software y pueda avanzar hacia el siguiente nivel.

Los niveles de madurez de una organización en CMMI son:

- Inicial.
- Gestionado.
- Definido.
- Gestionado cuantitativamente.
- Optimizando o en optimización continua

3. RELACIÓN ENTRE PSP/TSP Y CMMI

PSP forma técnicos de equipos establecidos con TSP en la mayoría de las prácticas genéricas de CMM/CMMI. TSP por sí solo, aunque sea aplicado a todos los equipos de desarrollo, no cubre todas las prácticas de cada área de proceso de CMM/CMMI, razón de más para que sea utilizado de forma complementaria a este modelo, no de forma aislada (ver figura 11).

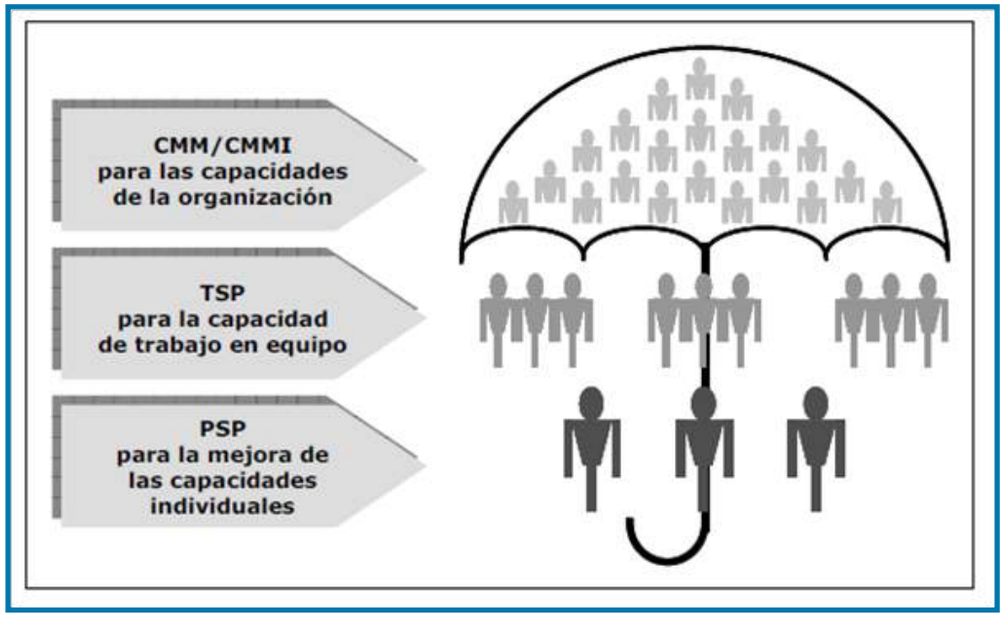

Figura 11. Relación de los modelos de calidad PSP, TSP y CMMI.

Debido a que las actividades de medición y análisis de los resultados son fundamentales en PSP y en TSP, su utilización durante la aplicación de CMM/CMMI en una organización, permite acelerar el progreso y aumentar el nivel de capacidad de la empresa en un tiempo menor que sin su uso.

El objetivo de cualquier esfuerzo de mejora de procesos de software es aumentar el rendimiento de la organización, y ello requiere cambios en el funcionamiento de los procesos de ingeniería, cualquier esfuerzo de mejora debe estar acompañado de pasos que puedan demostrar cambios en los comportamientos de ingeniería. TSP y PSP se utilizan para este fin. TSP genera una mejora substancial en el rendimiento de los grupos de software en una organización.

3.1. ¿Por qué incorporar PSP y TSP en una organización desarrolladora de software?

Las razones fundamentales para seleccionar los modelos de calidad denominados: Personal Software Process - PSP y Team Software Process - TSP son:

- La gran mayoría de las empresas que desarrollan software en nuestro contexto, son menores a 50 empleados.
- El modelo que utilizan nuestros competidores (CMMI) es complejo y apropiado para organizaciones grandes.
- El TSP/PSP, cuando se implementa correctamente, ha probado ser más eficaz que el CMMI Nivel 5.

Partiendo de estos modelos se puede escalar de forma acelerada a los diferentes niveles del modelo corporativo denominado CMMI. A este método de mejora acelerado en la implementación de las prácticas CMMI utilizando el modelo TSP como plataformase denomina TCAIM.

El método TCAIM [6], permite responder los interrogantes propios de una mejora de proceso en una organización, como son: "¿qué cambiar?" y "¿cómo hacer esto?". Así: el primer interrogante, desde la visión holística derivada del modelo CMMI y el segundo interrogante, abordado por el control detallado que permite realizar los modelos TSP y PSP, desde las métricas definidas en el proceso de desarrollo.

4. ¿CÓMO INCORPORAR DESDE LA ACADEMIA ESTOS MODELOS DE CALIDAD EN LOS PROCESOS DE ENSEÑANZA?

Los modelos de calidad dentro de la oferta académica no deben ser islas, sino por el contrario estructurar el conocimiento técnico en el estudiante, para que le permita al profesional entrar en una cultura de automejoramiento proactivo y realizable.

Por esta razón se deben buscar estrategias en el aula que surtan efecto en las competencias asociadas a las áreas técnicas; articulando desde los primeros semestres, prácticas de programación y calidad propicias para su nivel de formación, y así vivenciar las prácticas de calidad.

En el ámbito académico vale la pena resaltar experiencias de orden internacional, como el caso de México, que ha implementado las siguientes estrategias:

- La universidad y la empresa deben fortalecer la relación existente generando espacios compartidos para la discusión y la búsqueda de oportunidades creando un lenguaje común alrededor de la calidad de software.
- La enseñanza de los conceptos de calidad debe iniciar con el modelo PSP, para que permitan al estudiante conocer los beneficios de utilizar un proceso definido y crear una cultura de autocontrol.
- Los métodos de calidad requieren de una madurez en el estudiante a nivel técnico y conceptual, que solo se alcanzan en los semestres finales, porque en este momento el estudiante ha consolidado los conocimientos de diseño y programación necesarios para avanzar en una mejora constante en su proceso de desarrollo, evidenciable y controlable desde las métricas de calidad reflejadas en el marco PSP a nivel de indicadores de productividad.
- A su vez, el modelo TSP integra las condiciones de mejora identificadas en el proceso de desarrollo a nivel individual, que solo serán apropiadas por el estudiante en un equipo de desarrollo siguiendo la dinámica que los proyectos reales imponen en este sector. Teniendo como premisa que el modelo TSP, es un marco complementario al modelo PSP, el cual le da visibilidad a este último en el ambiente empresarial. El estudiante solo se apropia de estas competencias realizando prácticas empresariales simulando un ambiente que demanda el dinamismo productivo.

REFERENCIAS BIBLIOGRÁFICAS

[1]Sistemas de gestión de calidad: conceptos y vocabulario (traducción certificada), ISO 9000:2000,ISO 2000.

[2] Watts S. Humphrey. PSP(sm): A Self-Improvement Process for Software Engineers. Addison Wesley, 2005.

[3] Watts S. Humphrey. Winning with Software: An Executive Strategy. Addison Wesley, 2001.

[4]CMMI. Calidad. Ingeniería del Software. Consultado en Septiembre de 2010.Disponible:http://www.ingenierosoftware.com/calidad/cmmc-mmi.php

[5] CMMI Transition Plan. Consultado en Septiembre de 2010. Disponible: http://www.sei.cmu.edu/cmmi/background/transplan.html

[6] Miluk, Gene; McHale, Jim; y Chick, Tim. Accelerating CMMI Adoption with PSP/TSP-TCAIM. Proceedings of the TSP Symposium (September 2008. Consultado en Septiembre de 2010. Disponible: http://www.sei.cmu.edu/tspsymposium, visitada en Septiembre de 2010.

ARTÍCULO III

ESTADÍSTICA BÁSICA EN EXCEL

Leonardo Ceballos Urrego

Profesor de Tiempo Completo, Facultad de Informática

Tecnológico de Antioquia - Institución Universitaria

lceballos@tdea.edu.co

RESUMEN

Un curso de estadística básica se puede dictar ágil y efectivamente utilizando las herramientas que provee la hoja de cálculo Excel bajo Windows. Con esta es posible obtener en forma rápida las medidas de tendencia central y dispersión, así como agrupar datos, efectuar correlaciones y análisis de varianza de un factor.

Palabras clave: Estadística básica, Excel, Medidas de ubicación.

INTRODUCCIÓN

Una de las definiciones más conocidas sobre la estadística dice que es el estudio científico de la recolección, organización, sistematización, análisis y divulgación de informaciones (Spiegel, 1993) procedentes de estudios o investigaciones con un fin específico. También se sabe que el estudio de la estadística se divide en estadística paramétrica y no paramétrica, y que la primera comprende varias ramas entre las que se distinguen: la estadística descriptiva, inferencial y la teoría de probabilidades.

El estudio profundo de los diferentes campos de la estadística estuvo restringido durante mucho tiempo a personas interesadas por este saber, debido, en gran parte a la falta de herramientas que permitieran obtener en forma eficiente, efectiva y confiable los resultados esperados sobre los datos provistos por fuentes con altos volúmenes de información (Walpole, 1999). Es precisamente en la cantidad de información y no en los cálculos en los que se puede justificar un retraso de las prácticas estadísticas, ya que en general, los fundamentos de esta ciencia, como se puede apreciar en el estudio de la estadística descriptiva, requieren solo del conocimiento de las operaciones básicas de la matemática y en pocas ocasiones de algunos cálculos matemáticos, o la aplicación de fórmulas de baja exigencia neuronal.

La aparición de los ordenadores personales, fomentó la generación de una amplia gama de software estadístico básico y especializado, dentro de los cuales se conocen hoy paquetes como: Sattgrafics; SAS; Spss; Spad, y muchos otros con los cuales los investigadores y estadísticos han encontrado formas efectivas y ágiles de resolver sus problemas. Con estos aparatos el estudio de las diferentes ramas de la estadística se ha potenciado a tal grado, que resulta inocuo ofrecer un curso sin el acompañamiento de la computadora personal,

o al menos una pantalla visible para la clase, donde se puedan evidenciar los cálculos estadísticos que se pueden realizar sobre significativos volúmenes de información.

Igualmente útiles para el posicionamiento de la estadística, como ciencia básica, resultaron las hojas de cálculo incorporadas a los computadores personales, como "lotus 1,2,3" que se ejecutaba bajo DOS; o la actual "Excel bajo Windows" en la cual se encuentran herramientas que sirven para realizar, entre otros, estudios de regresión y correlación, pruebas de hipótesis, análisis de varianza de uno y dos factores, y muchos otros cálculos atinentes tanto a la estadística paramétrica como a la no paramétrica

Como sustento de todo lo anterior se presentan a continuación dos rutas en Excel (Excel bajo Windows, 2007) con un ejemplo práctico, con el que se evidencia el gran aporte que para la enseñanza de la estadística básica representa esta hoja de cálculo.

1. CÁLCULO, PASO A PASO, DE VALORES ESTADÍSTICOS PARA VARIABLES CUANTITATIVAS

1.1 Abrir excel.

1.2 Ingresar los datos suministrados en una sola columna.

1.3 Ubicarse en otra columna distinta a la de los datos y escribir hacia abajo: Moda, Mediana, Media; Varianza, Desviación Típica, Cuartil 1, Cuartil 3, Percentil 10, Percentil 30, Percentil 80, Percentil 37, Percentil 88.

1.4 Ubicarse en la celda contigua a donde escribió Moda.

1.5 Buscar la pestaña: Insertar, y seguir la secuencia: Función, Estadística, Moda.

1.6 Cuando llegue a la Moda, se abre un cuadro que le pide los valores a los que les desea calcular la Moda, entonces ubica el cursor en el primer número y con el clic sostenido, señala todos los números suministrados. Suelta y le da aceptar al cuadro.

1.7 El valor que aparece corresponde a la moda de los datos suministrados.

1.8 Repetir los pasos 4 a 7 para calcular los demás valores nombrados: Mediana, Media, etc.

2. CÁLCULO RÁPIDO Y GLOBAL DE DATOS ESTADÍSTICOS PARA UNA INFORMACIÓN CON DATOS CUANTITATIVOS

1. Abrir excel.

2. Desplegar en el orden indicado las pestañas: Datos, y verificar si está la opción: Análisis de datos, en caso negativo desplegar: Herramientas, Complementos, Ir, y cuando esta se despliegue elegir la opción: Herramientas para análisis. Así se incluirá la opción; Análisis de datos dentro de la pestaña: Herramientas, en versiones 2003 o anteriores, y a la derecha de la pestaña datos en versiones 2007 en adelante.

   Desplegar: Análisis de datos, y elegir: Estadística descriptiva; dar aceptar, y aparece un cuadro que le pide un rango de entrada.

   Con el mouse se señalan los datos suministrados, los cuales deben estar distribuidos en columnas etiquetadas, y luego los campos: En una hoja nueva y Resumen de estadísticas, luego: aceptar, y aparecen en una nueva hoja todos los datos estadísticos básicos de la información señalada.

A continuación se presentan los resultados arrojados al aplicar la ruta dos anterior, a un grupo de 1083 individuos a los cuales se les consultó, entre otros aspectos: la edad, el peso y la estatura; dichos resultados los arroja la hoja de cálculo en menos de 3 segundos, desde que se apliquen las instrucciones correctamente.

Edad		Peso/kg		Estatura/cm	
Media	21,4395199	Media	60,5374753	Media	165,453017
Error típico	0,16798917	Error típico	0,34094907	Error típico	0,27655022
Mediana	20	Mediana	59	Mediana	165
Moda	17	Moda	60	Moda	165
Desviación estándar	5,52834977	Desviación estándar	10,8569663	Desviación estándar	8,79325339
Varianza de la muestra	30,5626512	Varianza de la muestra	117,873718	Varianza de la muestra	77,3213052
Curtosis	5,07311021	Curtosis	0,25587557	Curtosis	-0,54051062

Edad		Peso/kg		Estatura/cm	
Coeficiente de asimetría	1,89375412	Coeficiente de asimetría	0,7241226	Coeficiente de asimetría	0,26662081
Rango	42	Rango	69	Rango	48
Mínimo	15	Mínimo	31	Mínimo	142
Máximo	57	Máximo	100	Máximo	190
Suma	23219	Suma	61385	Suma	167273
Cuenta	1083	Cuenta	1014	Cuenta	1011

En una entrega posterior se presentarán ejemplos resultantes de la aplicación de otras rutas que permiten, entre otras cosas, el agrupamiento de datos, el cálculo de estudios de correlación entre dos variables, y la aplicación de análisis de varianza para uno y dos factores (Hopkins, Hopkins, & Glass, 1997).

LISTA DE REFERENCIAS

Hopkins, K. D., Hopkins, B., & y Glass, G. (1997). *Estadística básica.* México: Prentice-Hall.

Microsoft. (2007). *Excel bajo Windows.*

Spiegel, M. R. (1993). *Estadística.* Madrid: Mc Graw Hill.

Walpole, R. E. (1999). *Probabilidad y estadística para ingenieros*. México: Prentice-Hall.

ARTÍCULO IV

AVANCES EN LA GENERACIÓN AUTOMÁTICA DE CÓDIGO A PARTIR DE ESQUEMAS PRECONCEPTUALES

Carlos Mario Zapata Jaramillo, Ph. D.
Líder del Grupo de Investigación
en Lenguajes Computacionales
Escuela de Sistemas
Universidad Nacional de Colombia Sede Medellín

RESUMEN

Pese al desarrollo actual de las herramientas *Computer-Aided Software Engineering* (CASE) y a la existencia de múltiples métodos de desarrollo de software, la generación automática de código ejecutable y funcional sigue siendo una de las promesas incumplidas de la Ingeniería de Software. A las dificultades normales que exhiben las herramientas CASE, que generan sólo parcialmente el código desde esquemas conceptuales difíciles de entender para los interesados, se suma el hecho de que estos comunican de manera vaga e imprecisa sus necesidades y expectativas. Por las razones anteriores, en este artículo se hace un compendio de los avances en la generación automática de software tomando como punto de partida los denominados esquemas preconceptuales, que son representaciones gráficas del discurso del interesado para facilitar la comunicación y la validación del dominio del problema desde fases preliminares del ciclo de vida del software.

Palabras clave: Esquemas preconceptuales, Herramientas CASE, Reglas heurísticas, Generación automática de código fuente.

INTRODUCCIÓN

Las herramientas CASE se crearon, principalmente, para asistir a los analistas en labores como el trazado de diagramas, la ingeniería inversa y la generación de código. En esta última tarea, herramientas de tipo comercial como Rational Rose® (IBM Corporation, 2010), Together™ (BORLAND Software Corporation, 2010) y Poseidon® (Gentleware, 2010) y otras de software libre, como ArgoUML® (Tigris.org, 2010) suelen generar sólo una parte del código, que generalmente se asocia con la definición de clases, atributos y el encabezado de los métodos en varios lenguajes de programación. Si bien esta actividad reemplaza otras de tipo manual que deben realizar los analistas en ausencia de herramientas CASE, los esfuerzos aún son insuficientes y la promesa de generación de código completamente funcional todavía sigue sin tener respuestas concretas. Por otra parte, los diferentes métodos de desarrollo de software, ya sean basados en planes o ágiles, presentan la misma limitación en cuanto a la generación automática de código, con el agravante de que muchas de ellas, incluso, no poseen un esquema de consistencia que permita realizar traducciones sucesivas de los modelos de requisitos hasta llegar al código ejecutable.

La situación se agrava aún más cuando se toma en consideración que los interesados, principal fuente de información para el levantamiento de los requisitos de una aplicación, suelen expresar sus necesidades y expectativas en lenguaje ambiguo y con problemas de precisión. Esta situación la reflejan Zapata y Olaya (2007) en forma de cómic, como se muestra en la figura 12.

Figura 12. Parodia de la expresión de necesidades y expectativas del interesado (tomado de Zapata & Olaya, 2007).

A esta situación, los desarrolladores responden con una necesidad adicional, que también muestran en forma de cómic Zapata y Olaya (2007) en la figura 13. En este caso, es el deseo de los desarrolladores de que el software sea como un kit prediseñado y listo para usar, sin mediar proceso alguno.

Figura 13. Parodia del deseo de los desarrolladores de una aplicación (tomado de Zapata & Olaya, 2007).

Finalmente, la visión de los vendedores de software, expresada de forma similar en los cómic de Zapata y Olaya (2007), se muestra en la figura 14. En ella, el vendedor de software presenta unos argumentos, pero otra es la realidad que rodea el desarrollo de la aplicación, ausente de propuestas metodológicas y con un bajo uso de estándares en las diferentes fases del ciclo de vida del software.

Figura 14. Parodia de la opinión de los vendedores de software en relación con la estandarización y los procesos de calidad (tomado de Zapata & Olaya, 2007)

La problemática esbozada constituye la motivación para que, en este artículo, se presente un recuento de los avances en la generación automática de código fuente para el software, fundamentando la propuesta en los denominados esquemas preconceptuales. Estos avances corresponden al trabajo que, en esta materia, viene realizando el Grupo de Investigación en Lenguajes Computacionales de la Universidad Nacional de Colombia, sede Medellín. Los esquemas preconceptuales constituyen un punto de partida para esta aproximación, puesto que por su cercanía con el discurso del interesado, posibilitan la interacción entre este y el analista para facilitar la validación de sus conceptos y relaciones.

El resto de este artículo se estructura de la siguiente manera: en la sección 2 se presenta una conceptualización de la problemática, que incluye el trabajo previo del Grupo de Investigación en Lenguajes Computacionales relacionado con este tema; en la sección 3 se presenta el compendio de los avances que rodean la generación automática de código, desde la óptica del Grupo de Investigación en Lenguajes Computacionales; finalmente, en la sección 4 se discuten las conclusiones y el trabajo futuro.

1. CONCEPTUALIZACIÓN DE LA PROBLEMÁTICA

Se denomina educción de requisitos al proceso que se realiza para capturar las necesidades y expectativas de los interesados para traducirlas, posteriormente, en especificaciones formales o semiformales de software (Leite, 1987). La educción de requisitos es un proceso complejo, porque se basa en la interacción humana y el nivel de comprensión que puedan alcanzar los actores del proceso sobre el dominio del problema.

En la figura 15 se esquematizan las principales dificultades del proceso (denotadas por símbolos de interrogación en las líneas discontinuas) que se resumen en la baja comprensión que tienen los analistas en relación con el discurso del dominio y el exiguo conocimiento que exhiben los interesados para entender esquemas conceptuales o representaciones sintácticas o semánticas del discurso (Zapata & Olaya, 2007).

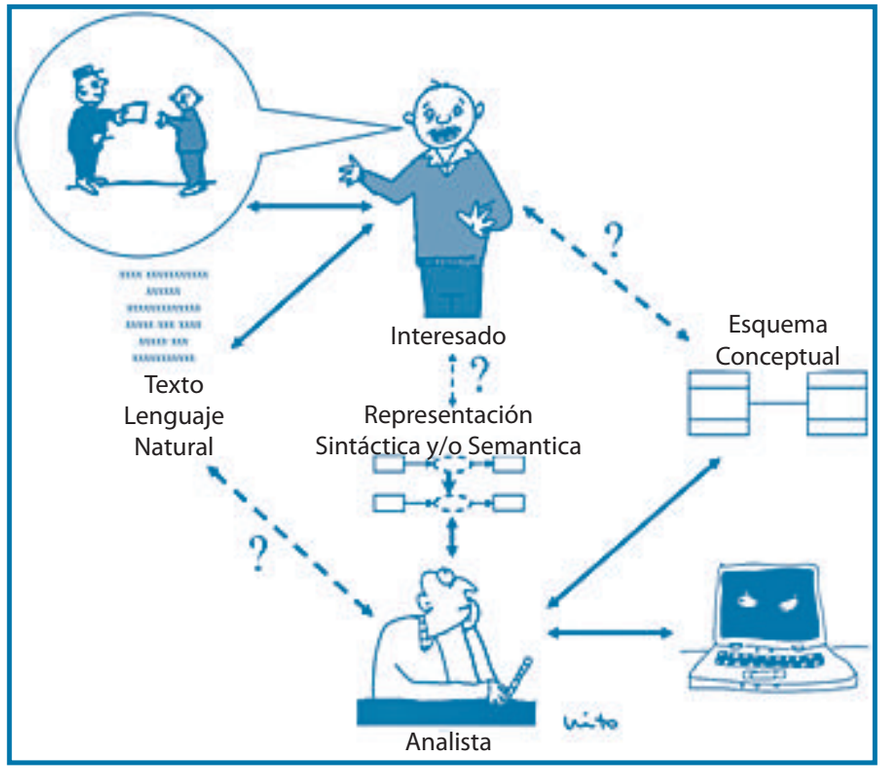

Figura 15. Representación de las dificultades de la educción de requisitos (tomado de Zapata & Olaya, 2007).

Los principales esfuerzos que se vienen realizando en educción de requisitos se suelen concentrar en la conversión automática de esquemas conceptuales en código fuente (cuya representación es el computador en la figura 4). Esto se suele hacer con herramientas CASE convencionales. Sin embargo, hacerlo de esta manera presenta tres problemas principales: el desconocimiento de los interesados en relación con los esquemas conceptuales, el descuido en automatización de los elementos iniciales del proceso (en la figura 4, el texto en lenguaje natural y su representación sintáctica y/o semántica) y los proble-

mas de completitud que tiene el código generado, pues se suele extraer de diagramas de clases y secuencias, generando tan sólo una mínima parte del código necesario para una aplicación ejecutable.

En la figura 16 se presenta otra posible solución al problema, que compendia los esfuerzos del Grupo de Investigación en Lenguajes Computacionales de la Universidad Nacional de Colombia, sede Medellín. Allí, se muestran los denominados esquemas preconceptuales (Zapata *et al.*, 2006) como una representación intermedia del dominio del problema, que se origina desde un discurso en el lenguaje controlado UN-LENCEP (Zapata *et al.*, 2008), que se combinan en el entorno UNC-Diagramador para generar esquemas conceptuales de UML desde el lenguaje controlado. El uso de estos elementos y el resultado que se genera se puede apreciar en la figura 17, donde el dominio del problema se representa gráficamente (por facilidad, aunque se supone que hay un equivalente en lenguaje natural del mismo). Luego, se define el discurso en UN-Lencep, del cual se obtiene el esquema preconceptual para, finalmente, generar los diferentes diagramas de UML (Clases, Comunicación y Máquina de Estados). Sin embargo, en este proceso no se genera código fuente, sino únicamente los diagramas que especifican el dominio del problema, lo que deja aún en manos de los desarrolladores la traducción de los esquemas conceptuales en la aplicación de software que sirve para solucionar los diferentes problemas inherentes al dominio.

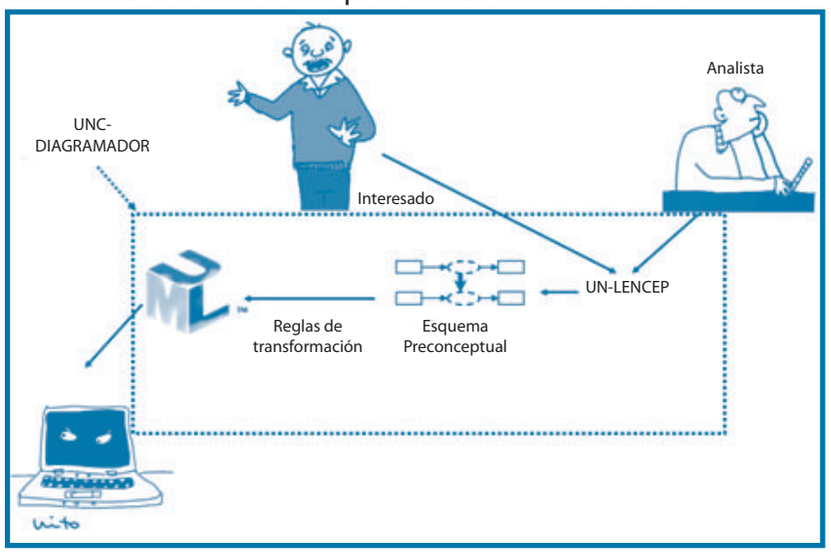

Figura 16. Compendio de la solución que propone el Grupo de Investigación en Lenguajes Computacionales (Tomado de Zapata & Olaya, 2007)

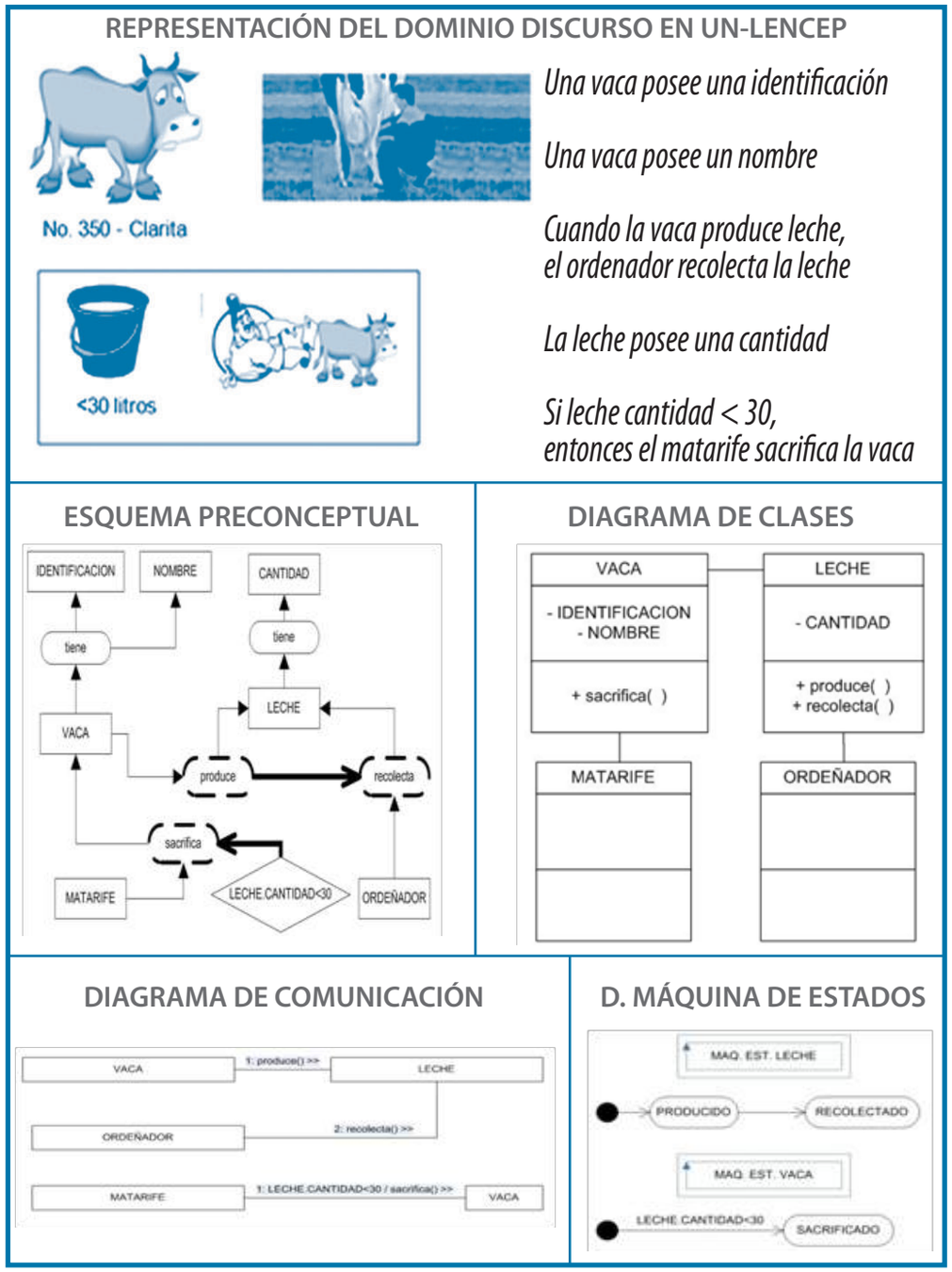

Figura 17. Representación del proceso desde la perspectiva del Grupo en Lenguajes Computacionales (tomado de Zapata & Olaya, 2007).

2. AVANCES EN GENERACIÓN AUTOMÁTICA DE CÓDIGO DESDE LENGUAJE CONTROLADO: UNA VISIÓN DEL GRUPO DE INVESTIGACIÓN EN LENGUAJES COMPUTACIONALES

Zapata y Chaverra (2010) emplean el trabajo previo del grupo en Lenguajes Computacionales y lo complementan para generar código fuente correspondiente a las interfaces gráficas de usuario de una aplicación ejecutable. Definen un conjunto de plantillas que permiten obtener código en el lenguaje Java para trazar automáticamente las interfaces gráficas de usuario. Así, es posible generar campos (figuras 18 y 19), listas de valores (figura 20), botones radio o listas desplegables (figura 21) y cajas de chequeo o listas de selección múltiple (figura 22).

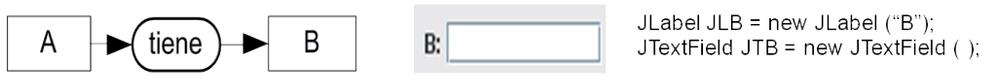

Figura 18. Generación del código de un campo sencillo a partir de un concepto hoja (tomado de Zapata & Chaverra, 2010).

Figura 19. Generación de varios campos a partir de conceptos hoja de diferentes conceptos (tomado de Zapata & Chaverra, 2010).

Figura 20. Generación de listas de valores a partir de conceptos únicos (tomado de Zapata & Chaverra, 2010).

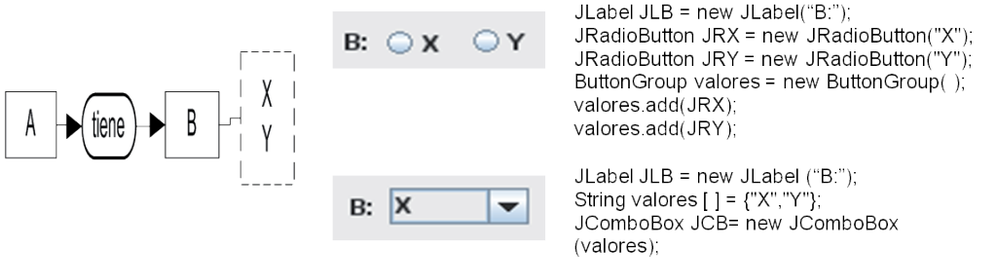

Figura 21. Generación de botones radio o listas desplegables a partir de posibles valores (excluyentes) de un concepto (tomado de Zapata & Chaverra, 2010).

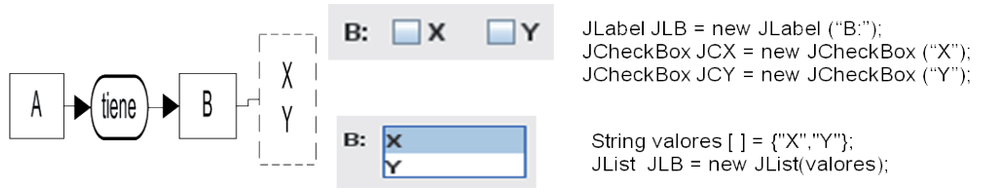

Figura 22. Generación de cajas de chequeo o listas de selección múltiple a partir de posibles valores de un concepto. Es el mismo caso de la Figura 10 pero con posibles valores no excluyentes (tomado de Zapata & Chaverra, 2010).

Con fines de ejemplificación de las reglas mencionadas, se presenta el esquema preconceptual de la figura 23. Las interfaces que se pueden generar, incluyendo la porción del esquema que las origina, se pueden apreciar en las figuras 24, 25 y 26.

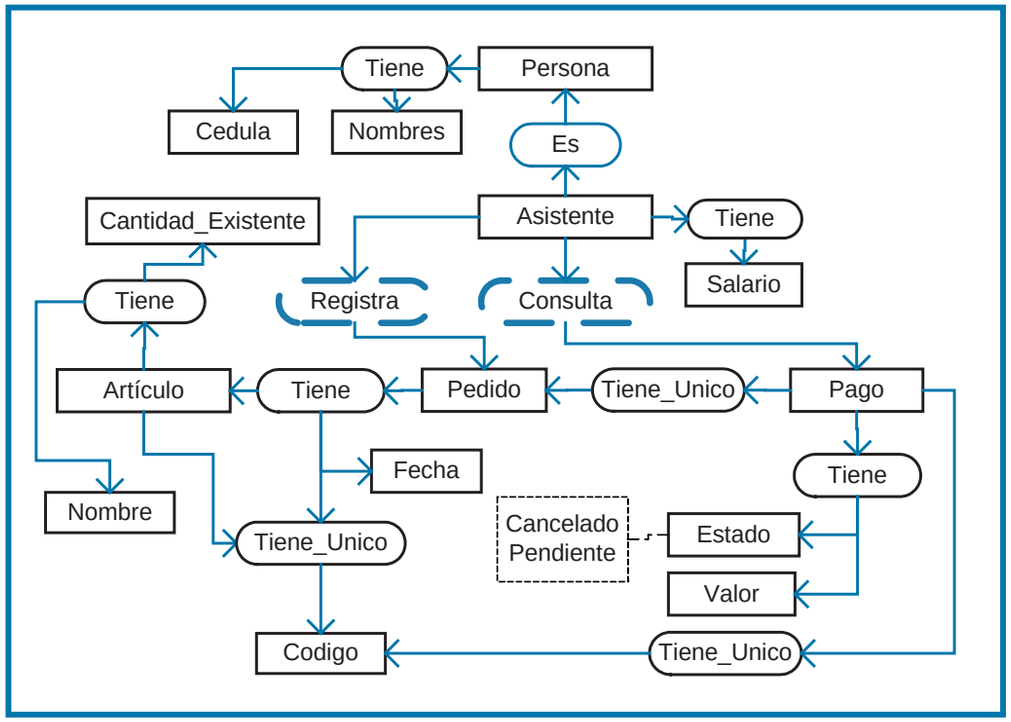

Figura 23. Ejemplo de un esquema preconceptual (tomado de Zapata y Chaverra, 2010).

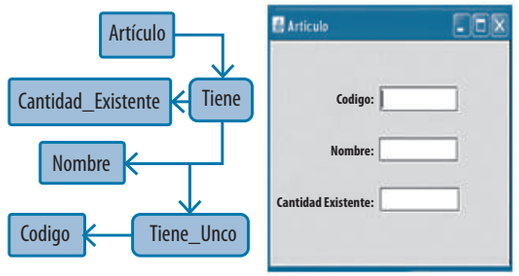

Figura 24. Generación de campos de texto en el ejemplo (Zapata & Chaverra, 2010).

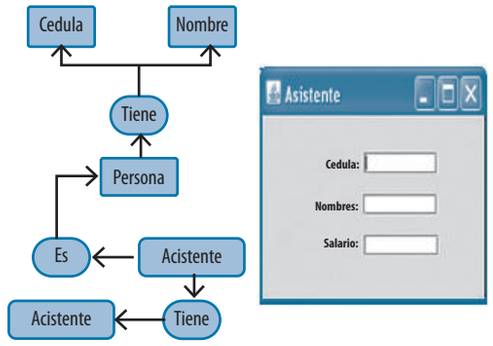

Figura 25. Generación de campos de texto desde estructuras más complejas (tomado de Zapata & Chaverra, 2010)

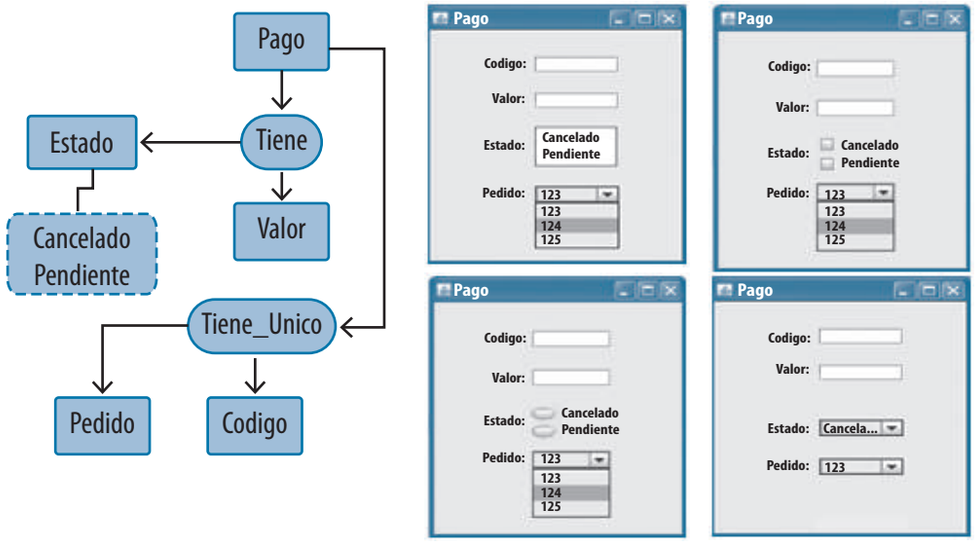

Figura 26. Generación de diferentes opciones de diseño para posibles valores (tomado de Zapata & Chaverra, 2010).

3. CONCLUSIONES Y TRABAJO FUTURO

El proceso de educción de requisitos se apoya, tradicionalmente, en herramientas CASE convencionales, que posibilitan el trazado de los diferentes diagramas que especifican el dominio del problema y luego permiten realizar su traducción a código fuente. El resultado de ese proceso es, todavía, código incompleto, puesto que se generan sólo porciones de algunos elementos del código sin que se pueda considerar aún ejecutable. Además, el punto de partida carece de la validación que puede suministrar el interesado, puesto que los lenguajes de partida de las herramientas CASE son esquemas conceptuales, lejanos del conocimiento y entendimiento del interesado.

El Grupo de Investigación en Lenguajes Computacionales, consciente de estas falencias, viene desarrollando una estrategia que permita resolver parcialmente estos dos problemas. Así, se generó, en principio, un entorno para capturar el dominio del interesado para luego representarlo en esquemas preconceptuales y, finalmente, transformarlo en esquemas conceptuales de UML. Posteriormente, se generó un conjunto de reglas heurísticas que permite la generación de las interfaces gráficas de usuario, en lenguaje Java, completamente ejecutables y funcionales. Algunas de esas reglas permiten diferentes opciones de diseño, dependiendo de las características de los datos y de las preferencias del interesado.

De esta manera, se involucra completamente al interesado en el proceso de creación de la aplicación, a la vez que se logra su intervención en la validación desde etapas tempranas del desarrollo. El código generado es consistente con el discurso representado en el esquema preconceptual y con los diferentes diagramas de UML que se pueden generar, dado que provienen todos del mismo discurso en el lenguaje controlado UN-Lencep. Se espera que esta solución se pueda traducir en mayor celeridad para desarrollar aplicaciones de software y, por ende, en costos de desarrollo inferiores para este proceso.

Las líneas de trabajo futuro que se pueden derivar de este trabajo son las siguientes:

- Adición de nuevos elementos a los esquemas preconceptuales que, sin sacrificar la sintaxis sencilla de dichos esquemas, permita una especificación más precisa, lo que se traduciría en más elementos para la transformación a código fuente.

- Determinación de equivalencias del UN-Lencep en lenguaje natural. Aún los discursos en UN-Lencep son muy controlados y eso, en ocasiones, puede dejar por fuera de la especificación elementos que se mencionan en el discurso en lenguaje natural.
- Definición de nuevas reglas heurísticas que permitan tomar decisiones de diseño en relación con las interfaces gráficas de usuario. Entre más se acerquen estas decisiones a las preferencias del interesado, mayor probabilidad de éxito tendrá la aplicación que se desarrolle.
- Definición de reglas heurísticas que tomen como punto de partida otros diagramas de UML que se puedan generar desde los esquemas preconceptuales. Además, se requieren otras reglas heurísticas para mejorar la especificación de la aplicación que se expresa en el código fuente.
- Definición de un esquema de persistencia de los datos que posibilite la interacción de las interfaces así generadas con sistemas gestores de bases de datos. Esta línea de trabajo futuro permitiría cerrar la brecha entre los esquemas preconceptuales y los lenguajes de programación, de forma que el trabajo de los desarrolladores se especializaría y se alejaría de las tareas simples y repetitivas que se encuentran en la elaboración de aplicaciones.
- Incorporación del entorno completo en las herramientas CASE, de modo que los analistas lo puedan emplear en la solución de los problemas en diferentes dominios.

AGRADECIMIENTOS

Una gran parte de este trabajo se realizó en el marco de dos proyectos de investigación:

- "TRANSFORMACIÓN SEMIAUTOMÁTICA DE LOS ESQUEMAS CONCEPTUALES, GENERADOS EN UNC-DIAGRAMADOR, EN PROTOTIPOS FUNCIONALES", bajo la financiación de la Vicerrectoría de Investigación de la Universidad Nacional de Colombia.
- "UN MODELO DE DIÁLOGO PARA GENERACIÓN AUTOMÁTICA DE ESPECIFICACIONES EN UN-LENCEP", bajo la financiación de la Dirección de Investigación de la Sede Medellín de la Universidad Nacional de Colombia.

LISTA DE REFERENCIAS

BORLAND Software Corporation. Borland Together™ Architect. En: http://www.borland.com/us/products/together/index.html. [Consultado 10 de Agosto de 2010].

Gentleware Business to model. Poseidon® http://www.gentleware.com/. [Consultado 10 de Agosto de 2010].

IBM Corporation. Rational Rose Architect™. En: http://www-306.ibm.com/software/awdtools/architect/swarchitect/index.html. [Consultado 10 de Agosto de 2010].

Leite, J. (1987). A survey on requirements analysis, Advanced Software Engineering Project. Technical Report RTP-071, Department of Information and Computer Science, University of California at Irvine.

Tigris.org. ArgoUML®. http://argouml.tigris.org/. [Consultado 10 de Agosto de 2010].

Zapata, C. M. & Chaverra, J. (2010). *Generación automática de interfaces gráficas de usuario a partir de esquemas preconceptuales*. Memorias del Quinto Congreso Colombiano de Computación (5CCC), Cartagena.

Zapata, C. M., Gelbukh, A. & Arango, F. (2006). Pre-conceptual Schemas: A Conceptual-Graph-Like Knowledge Representation for Requirements Elicitation. *Lecture Notes in Computer Science*, Vol. 4293, pp. 17–27.

Zapata, C. M., Gelbukh, A. & Arango, F. (2008). UN-LENCEP: A Controlled Language for Pre-conceptual Schema Specification. Memorias de las VII Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento, Guayaquil, pp. 269-276.

Zapata, C. M. & Olaya, Y. (2007). *Ingeniería de Software para analistas*. Medellín: C. Zapata (Ed.).

ARTÍCULO V

EL PROYECTO PEDAGÓGICO INTEGRADOR: UNA EXPERIENCIA EN EL PROCESO DE ARTICULACIÓN CON EL PROGRAMA TÉCNICO PROFESIONAL EN SISTEMAS

Manuel Alexánder Valbuena Henao
Ingeniero Informático
Especialista en Educación Superior
Asesor del Proyecto Pedagógico Integrador de la Media Técnica
Facultad de Informática
Tecnológico de Antioquia
AlexanderValbuena@gmail.com

RESUMEN

La integración de los conocimientos, habilidades, actitudes y valores adquiridos por los estudiantes de la especialidad en desarrollo de software de las cinco instituciones educativas articuladas con el Tecnológico de Antioquia en el marco de la Alianza Futuro Digital Medellín, es el propósito fundamental del equipo de trabajo de la media técnica. Esta meta es posible mediante la aplicación de una estrategia metodológica que impulsa el desarrollo de proyectos que evidencian la aplicación de las temáticas impartidas en los módulos de los dos primeros niveles del programa Técnico Profesional en Sistemas, en el marcodel Proyecto Pedagógico Integrador(PPI).

Palabras clave: Articulación, Competencia, Integralidad, Proyecto.

¿ARTICULAR LA EDUCACIÓN MEDIA Y LA EDUCACIÓN SUPERIOR?

Para aquellos que aún no están familiarizados con esta propuesta, la articulación de la educación media con la educación superior es un proceso mediante el cual se propicia el encuentro de los conocimientos, los valores, la filosofía y la dinámica organizacional de estos dos niveles educativos con el propósito de garantizar la continuidad de los jóvenes en su proceso formativo. En otros términos, las instituciones de educación superior acogen las políticas emanadas de la Ley 749 de 2002 para estructurar sus planes de estudio con una articulación basada en ciclos propedéuticos secuenciales y complementarios, que brindan una formación integral para el desempeño laboral o para la continuidad en el ciclo siguiente.

En el caso de la Facultad de Informática del Tecnológico de Antioquia, se viene desarrollando la articulación del programa Técnico Profesional en Sistemas con principios de calidad, pertinencia y coherencia; favoreciendo la formación integral, acercando el mundo laboral a las aulas de clase y preparando el recurso humano requerido por el sector productivo en cinco instituciones de Medellín: CASD José María Espinosa Prieto, Fe y Alegría Villa de la Candelaria, Gabriel García Márquez, Javiera Londoño y Santa Elena.

Para cumplir a cabalidad con los principios anteriormente mencionados y ser coherentes con las exigencias del medio, es necesario transformar la forma de

enseñanza dejando a un lado las estrategias mecánicas, individuales y memorísticas propias de la educación tradicional, y trascender a prácticas interdisciplinarias que estimulen a los estudiantes, los comprometan con su proceso formativoy les permitan relacionar y aplicar las competencias que adquieren en los diferentes módulos de aprendizaje. Esto es posible con los Proyectos Pedagógicos Integradores (PPI).

¿QUÉ ES EL PROYECTO PEDAGÓGICO INTEGRADOR (PPI)?

Es una estrategia metodológica que pretende integrar los conocimientos, habilidadesy actitudes adquiridos mediante su aplicación en la solución de problemas de la vida real. Se constituye en una propuesta apropiada para la *gestión educativa compartida*, donde cada módulo del currículo aporta las competencias necesarias que permiten el desarrollo del proyecto y la materialización de los productos requeridos.

INCORPORACIÓN DEL PPI EN LA DINÁMICA DE ARTICULACIÓN

El PPI se incorporó a la dinámica del proceso de articulación de la Facultad de Informática del Tecnológico de Antioquiaen el 2008, gracias al surgimiento de la Alianza Futuro Digital Medellín (AFDM), cuya finalidad es transformar los programas de formación en las instituciones de educación superior implementando un modelo educativo basado en competenciasque articula, mediante la estrategia de ciclos propedéuticos, la educación media técnica, técnica profesional y tecnológica de programas relacionados con el desarrollo de software.

PLANIFICACIÓN DEL PPI EN EL MARCO DE AFDM

De acuerdo conel objetivo de la AFDM orientado a la formación de técnicos y tecnólogos altamente competentes para el desarrollo de software,que respondan efectiva y rápidamente a los requerimientos de la industria que produce y comercializa productos o servicios de software, se adoptó un plan de trabajo cuyos elementos esenciales se fundamentan en los siguientes pilares:

- Adopción de la evaluación por competencias y elprincipio de la formación integral de los estudiantes.
- Metodologías basadas en el trabajo interdisciplinario que facilitan a los estudiantes la relación de contenidos y su posible aplicación.
- Condiciones que garantizan adquisición, asimilación y retención de las competencias y su materialización en productos creativos, innovadores y pertinentes.
- Relación entre la teoría y la práctica en contextos reales orientados al ámbito laboral, específicamente al sector de desarrollo de software.
- Concepción del estudiante como un ser perfectible, reflexivo y en evolución constante, capaz de adquirir compromisos y responsabilidades en su proceso de formación.
- Aplicación de las herramientas tecnológicas para resolver problemas a partir del desarrollo de software.
- Aplicación de buenas prácticas para obtener productos de calidad.
- Articulación con el sector productivo para la validación conjunta de temáticas y competencias desarrolladas en los módulos de formación.

Una vez revisada la coherencia del plan de trabajo propuesto, se comenzó con su ejecución en dos escenarios: la Facultad de Informática del Tecnológico de Antioquia con el equipo de trabajo de la articulación y las cinco instituciones educativas con los docentes articulados y los estudiantes de la especialidad en desarrollo de software. En el primer escenario se trabajaron los siguientes componentes:

ESTRATEGIA DE INDUCCIÓN PARA DOCENTES

En este componente se socializaron los fundamentos de la Alianza Futuro Digital Medellín (AFDM), las características de la propuesta de articulación del Tecnológico de Antioquia, el propósito del Proyecto Pedagógico Integrador (PPI) y el rol de cada uno de los módulos de formación para su desarrollo.

REVISIÓN DE LOS MÓDULOS DE FORMACIÓN

Una vez los docentes comprendieron la filosofía de la articulación, se pasó a un proceso de revisión de cada uno de los módulos de formación para verificar su coherencia con el perfil propuesto por AFDM. Esta actividad proporcionóespacios para unificar criterios con respecto a los conocimientos, habilidades y valores necesarios enla formaciónde los estudiantes.De igual manera, se efectuaron ajustes y complementos que fueron realizados por los docentes de cada una de las áreas del conocimiento.

SISTEMA DE CUALIFICACIÓN DOCENTE

Con base en que "la preparación del personal docente que orienta el proceso de formación en la articulación es crucial y debe adoptarse de manera permanente", se originó un plan de cualificación docente relacionado con los ajustes realizados a los módulos de formación. Este plan se estructuró con temáticas como la Programación Orientada a Objetos(POO), el Proceso Personal de Software(PSP) y los lenguajes de programación.

PLANEACIÓN DE LAS ACTIVIDADES

Con los insumos anteriormente mencionados se coordinó un plan de trabajo conformado por actividades, tales como:

- La planeación de los aportes de cada módulo de formación al desarrollo del Proyecto Pedagógico Integrador.
- La definición de fechas de reunión con los docentes articuladores para la verificación de los avances obtenidos en el proceso.
- La preparación de un cronograma de mesas de trabajo para evaluar el avance en cada una de las instituciones de educación media articuladas (IEM).
- La determinación de los cronogramas de entregables y de asesoríaspara los estudiantes de cada IEM.
- La definición de las evidencias que cada actor del proceso de articulación debía conservar para presentar ante la interventoría del mismo.

En el segundo escenario, es decir, en las cinco IEM articuladas se trabajaron los siguientes componentes:

PLANEACIÓN ENTRE LOS DOCENTES Y EL ASESOR DEL PPI

La planeación de actividades realizada con el equipo de trabajo del proceso de articulación del Tecnológico de Antioquia fue socializada de forma periódica con los docentes articulados. De esta forma, el asesor del PPIestableció mecanismos para coordinar, supervisar y verificar las actividades alrededor del proyectoen cada IEM articulada.

PLANEACIÓN ENTRE DOCENTES ARTICULADORES Y ARTICULADOS

Atendiendo los lineamientos establecidos por AFDM, los docentes articuladores y articulados realizaron planeaciones periódicas para definir las funciones, actividades y tareas que realizarían conjuntamente en la operación de cada uno de los módulos de formación. De igual forma, se determinaron las metodologías apropiadas para orientar el aprendizaje integral y las competencias objetivo de cada módulo.

ESTRATEGIA DE MOTIVACIÓN PARA ESTUDIANTES

Conformada por actividades para orientar a los estudiantes delaespecialidad en desarrollo de software de las cinco IEM articuladascon respecto a la AFDM, el perfil profesional propuesto, las competencias requeridas por el sector del desarrollo de software, los beneficios del proceso de articulación y la importancia del PPI en el proceso formativo.

ASESORÍA DE LOS PROYECTOS

De acuerdo con el cronograma de asesoríasy entregables programado, el asesor del PPI revisa, corrige y retroalimenta los avances de cada equipo de trabajo. El propósito de esta estrategia es el perfeccionamiento del entregable por medio de asesorías específicas para cada equipo de estudiantes, adoptando el rol de "gerente de proyecto"para los estudiantes del grado décimo y de "cliente del proyecto" para los estudiantes del grado once. De igual manera, se obtiene información sobre las competencias que ya han sido adquiridas por los estudiantes y aquellas que se deben potenciar para realizar la retroalimentación con los docentes articuladores y articulados.

EVALUACIÓN INTEGRAL DE LOS ENTREGABLES

Atendiendo el principio de la formación integral de los estudiantes, se propician espacios para valorar la adquisición de conocimientos, habilidades y actitudes por parte de los estudiantes y el nivel de integración de estos elementos en el desarrollo del proyecto. Estos espacios se realizan en una exposición en la que están presentes los docentes articulados y articuladores de cada módulo, quienes se encargan de retroalimentar el proceso partiendo de una lista de criterios.

PARTICIPACIÓN EN LAS MESAS DE TRABAJO

Las mesas de trabajo institucional programadas se emplean para la presentación de informes, la coordinación y supervisión de las acciones articuladoras en cada una de las IEM y como espacio para la definición de lineamientos claros que garanticen el seguimiento y control del proceso de articulación de la media técnica con la educación superior. En estas reuniones se exponen los avances relacionados con el desarrollo del PPI, se verifican los compromisos pendientes, se retroalimenta el proceso y se programan actividades.

RESULTADOS OBTENIDOS

Con la incorporación del PPIal proceso de articulación del Tecnológico de Antioquia se hanobtenido beneficios de todo tipo, principalmente en el proceso de formación de los estudiantes de los grados décimo y undécimo beneficiarios de la propuesta, quienes han potenciado las siguientes competencias y actitudes requeridas por el sector del software.

GRADO DÉCIMO

Competencias laborales generales

- Creatividad.
- Solución de problemas.
- Comunicación.
- Trabajo en equipo.
- Gestión de la información.
- Uso de herramientas ofimáticas.

Competencias específicas

- Producción de documentos digitales apoyados en las herramientas ofimáticas.
- Redacción de textos en forma lógica y coherente.
- Análisis e interpretaciónde un problema en un contexto determinado.
- Razonamiento lógico matemático.
- Aplicaciónde conceptos generales de la arquitectura de un sistema.
- Interpretacióndel análisis y el diseño para desarrollar un sistema.

GRADO UNDÉCIMO

Competencias laborales generales

- Toma de decisiones.
- Creatividad.
- Solución de problemas.
- Comunicación.
- Trabajo en equipo.
- Gestión de la información.
- Gestión y manejo de recursos.

Competencias específicas

- Delimitaciónde problemas de investigación en el contexto de la informática.
- Programación en el lenguaje seleccionado de acuerdo con el diseño establecido
- Aplicación de estándares de programación.
- Razonamiento lógico matemático.
- Aplicación de las diferentes técnicas de dirección de equipos.
- Aplicación conceptos sobre tendencia central en el análisis de datos.
- Diseñode páginas web con el lenguaje HTML.

Actitudes para ambos grados

- Aceptación del otro.
- Compromiso.
- Disciplina.
- Orden.
- Orientación al logro.
- Proactividad.
- Responsabilidad.
- Respeto.

Para finalizar, es importante que como docentes reflexionemos sobre nuestro quehacer y reevaluemosnuestra metodología para formar a los técnicos, tecnólogos y profesionales que necesita la sociedad, aplicando metodologías pedagógicas incluyentes y creativas que faciliten a los estudiantes actuales adquirir, desarrollar y potenciarlas competencias, valores y actitudes que les permitirán desempeñarse efectivamente en el contexto laboral. En nuestras manos está laresponsabilidadde formar no sólo profesionales, sino también personas integrales en todo el sentido de la palabra.

BIBLIOGRAFÍA

TECNOLÓGICO DE ANTIOQUIA. Articulación de la media técnica con la educación superior. Modelo técnico pedagógico. Medellín: Editorial Tecnológico de Antioquia – Institución Universitaria, 2005.

Proyecto Integrador. Documento base para la ejecución del Proyecto Integrador. Equipo asesor Alianza Futuro Digital,2009.

Aprendizaje por proyectos. NorthWest Regional Educational Laboratory

ARTÍCULO VI

VISIÓN GENERAL DEL TESTING

Luis Emilio Velásquez Restrepo
Ingeniero de Sistemas
Universidad Eafit
Especialista en Docencia Universitaria
Universidad de Antioquia
Docente de Cátedra Facultad Informática - Tecnológico de Antioquia

RESUMEN

Las técnicas de pruebas (*testing*) se emplean en las empresas, y deben ajustarse a las políticas internas. Aunque existen varios tipos de pruebas, falta mucho por explorar y por estandarizar para formalizar del proceso; no existen unas pautas que reglamenten los aspectos asociados a una política de *testing*; por lo general, esta se basa en la experiencia que tenga el equipo de desarrollo, adaptada a la empresa para la cual se crea el sistema. Por ello, se pretende desarrollar las destrezas de probador en las universidades, con el fin de mermar el tiempo de adaptación del recién egresado al equipo de pruebas.

Palabras clave: Artefacto, Ciclo de vida de las pruebas, Defecto, Derrotero, Falla, Lista de chequeo, Plan de pruebas, Probador, Requerimientos (funcionales y no funcionales).

INTRODUCCIÓN

Los técnicos y tecnólogos de sistemas llegan al sector productivo sin tener todas las competencias requeridas para probar productos de software, razón por la cual es necesario brindarles capacitación por parte de la empresa para aplicar satisfactoriamente modelos de pruebas de software.

Las pruebas sirven para chequear el funcionamiento de la aplicación, con el fin de hacer los ajustes requeridos hasta garantizar el cumplimiento de los requisitos, los requerimientos funcionales y no funcionales. Las pruebas se clasifican según su alcance. Primero se prueba cada módulo por separado y después dichos módulos integrados. A continuación, algunas definiciones de términos asociados a las competencias que deben desarrollar las personas dedicadas al *testing* (pruebas).

Pruebas: son los chequeos realizados al sistema para encontrar las diferencias entre el proceso actual y el esperado. El equipo de desarrollo se encarga de corregir las fallas hasta encontrar el comportamiento esperado. Dependiendo del objetivo de la prueba se clasifican en: pruebas unitarias, pruebas funcionales, pruebas estructuradas, pruebas de caja negra, pruebas de caja blanca, pruebas de frontera, pruebas de rutas, pruebas de estrés, pruebas de integración, entre otras.

Falla: es cualquier desviación del comportamiento observado con respecto al especificado.

Error: significa que el sistema está en un estado tal que el procesamiento adicional del sistema conducirá a una falla; lo cual causa que el sistema se desvíe del comportamiento pretendido.

Defecto: es la causa mecánica o algorítmica de un error. El objetivo de las pruebas es maximizar la cantidad de defectos descubiertos, lo cual luego permite que los desarrolladores los corrijan e incrementen la confiabilidad del sistema.

Caso de prueba: se definen a partir de los casos de uso que describen las funcionalidades del sistema, las entradas, salidas y flujos determinados por condiciones y reglas requeridas, para el correcto funcionamiento. Dichos flujos están definidos por un conjunto de pasos secuenciales agrupados por escenarios, los cuales pueden clasificarse como simples, básicos y alternos o excepcionales (ver tabla 7).

Tabla 7. Caso de prueba

ATRIBUTO	DESCRIPCIÓN
Nombre	Nombre del caso de prueba
Ubicación	Nombre de ruta completo del ejecutable
Entrada	Datos de entrada o comandos
Oráculo	Resultados esperados de la prueba contra los que se compara la salida de la prueba
Bitácora	Salida producida por la prueba

Evaluador: los evaluadores garantizan que los artefactos, los elementos de sitio desarrollados y el contenido funcionan como se espera antes de implementarlos en el entorno de producción*.

Ciclo de prueba: con cada nueva versión del producto se realizan algunas o todas las tareas asociadas a las pruebas, a esto se le llama un ciclo de prueba**.

Pruebas piloto: se selecciona una muestra de datos para hacer los chequeos antes de trabajar toda la carga del sistema, hay más procesos que pueden salir mal o resultar nuevos para los diseñadores y orientadores que participen en las pruebas; también pueden surgir dificultades con las tecnologías de apoyo.

Plan de pruebas: es una forma de planear las pruebas y casos de pruebas. El documento del plan de pruebas contiene: introducción, relación con otros documentos, panorama del sistema, características para probar y que no se prueban, criterios de aprobación o falla, enfoque, suspensión y reanudación, materiales para la prueba (requerimientos de hardware y software), casos de prueba, calendario de pruebas. El plan de pruebas se enfoca en los aspectos administrativos de las pruebas, documenta el alcance, enfoque, recursos y calendarización de las actividades de pruebas. En este documento se identifican los requerimientos y componentes para probar.

DOCUMENTACIÓN DE LAS PRUEBAS

Las actividades de las pruebas se documentan en cuatro tipos de documentos, *Plan de pruebas, Especificaciones de casos de prueba, Reportes de incidentes de pruebas* y *Reporte de resumen de pruebas.* En la documentación se emplea el estándar IEEE 829, También se puede cualquier otro estándar empleado por la empresa que esté implementando el sistema (ver figura 27).

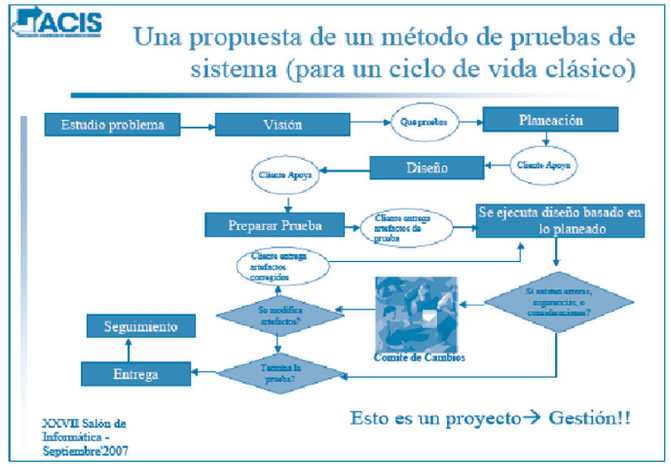

Figura 27. Propuesta de un método de pruebas

* Para ampliar información: http://technet.microsoft.com/es-es/library/cc262247.aspx#section5
** Véase: http://athenea.ort.edu.uy/publicaciones/ingsoft/ortsf/seminarios05/prueba.htm

Comparación entre el método científico y las etapas de la Ingeniería[***]:

Tabla 2. Comparación entre el método científico y las etapas de la Ingeniería

Etapas del método científico	Etapas de proceso de solucionador de problemas en Ingeniería
Observación e identificación de un problema de investigación	Vista panorámica del problema, formulación y análisis. Formulación de Objetivos
Recolección de información	Recolección, análisis y complementación de la información básica
Formulación de hipótesis o posibles soluciones al problema	Formulación de la hipótesis cuando sean necesarias o búsqueda de alternativas de solución con base en la información recolectada
Verificación de las hipótesis	Evaluación de alternativas para seleccionar la más conveniente
Conclusión	Selección de parámetros de diseño, y diseño de la alternativa escogida
Verificación	Especificación total de la alternativa y estructura del proyecto

Para simular al máximo las condiciones del entorno de producción, los evaluadores usan el entorno piloto. Este incluye todos los elementos desarrollados, todos los artefactos y el resto del contenido sin importar cómo se implementó. El entorno piloto existe en las mismas condiciones de red y de seguridad que el entorno de producción. Después de las pruebas realizadas en el entorno piloto, un administrador es responsable de implementar todos los elementos del sitio en el entorno de producción. Se hacen pruebas de carga y rendimiento, validación, aceptación y pruebas de usabilidad[****].

*** TORRES MUÑOZ ALICIA. Metodología del trabajo científico aplicada a la Ingeniería. Bogotá: Universidad Militar Nueva Granada, 2008.

**** Véase: http://www.uiaccess.com/justask/es/ut_prep.html

Al probar se emplean unas competencias, que son una forma de aplicar los contenidos temáticos de cada asignatura al objeto de estudio del programa que puede ser técnico, tecnológico o profesional, las competencias se determinaron siguiendo el derrotero del modelo planteado por el Servicio Nacional de Aprendizaje (SENA).

Durante el ciclo de vida de un producto, sin importar cual sea el proceso de desarrollo, se van generando distintas versiones de la aplicación. Las actividades de la prueba se realizan para una determinada versión del producto, sobre la cual se ejecutan las pruebas y se reportan los incidentes encontrados. Las pruebas que serán ejecutadas sobre una versión son planificadas con anticipación y deberían ser ejecutadas, a menos que las prioridades cambien. En un ciclo de prueba se puede ejecutar una, alguna o todas las pruebas planificadas para el producto.

Cada ciclo de prueba está asociado a una versión del producto por probar, cada nuevo ciclo de prueba implica una nueva versión de uno o más componentes del sistema. Uno de los principales desafíos desde el punto de vista de la prueba independiente es estimar cuántos ciclos de prueba se requieren, ya que no todas las versiones que generan desarrollo llegan a ser probadas por el equipo de prueba; entre dos ciclos de prueba podrían existir más de dos versiones del producto generadas por el equipo de desarrollo.

Las empresas que llevan a cabo proyectos de desarrollo de software tienen problemas al encontrar personal con las competencias necesarias para aplicar pruebas a un producto de software.

Las competencias requeridas para realizar las pruebas de elementos de software amplían el campo de acción del técnico o tecnólogo, ya que las empresas preferirían contratar personal capacitado en vez invertir tiempo y dinero en la formación del personal recién contratado.

Al desarrollar las habilidades para el *testing* (pruebas), se debe escoger entre formar personas solo probadoras (*test roles*) o formar desarrolladores con destrezas para probar software.

Dependiendo del tipo de prueba se establece el perfil de la persona encargada de diseñarla y el perfil de la persona que realizará la prueba. Para poder aplicar las pruebas se desarrollan unas competencias específicas.

Las competencias (habilidades) requeridas para desempeñar el rol de probador en las instituciones educativas son desarrollas en forma aislada; en los planes de estudio se observan en: un modulo, un tema dentro de una asignatura, un simposio, un seminario, un diplomado.

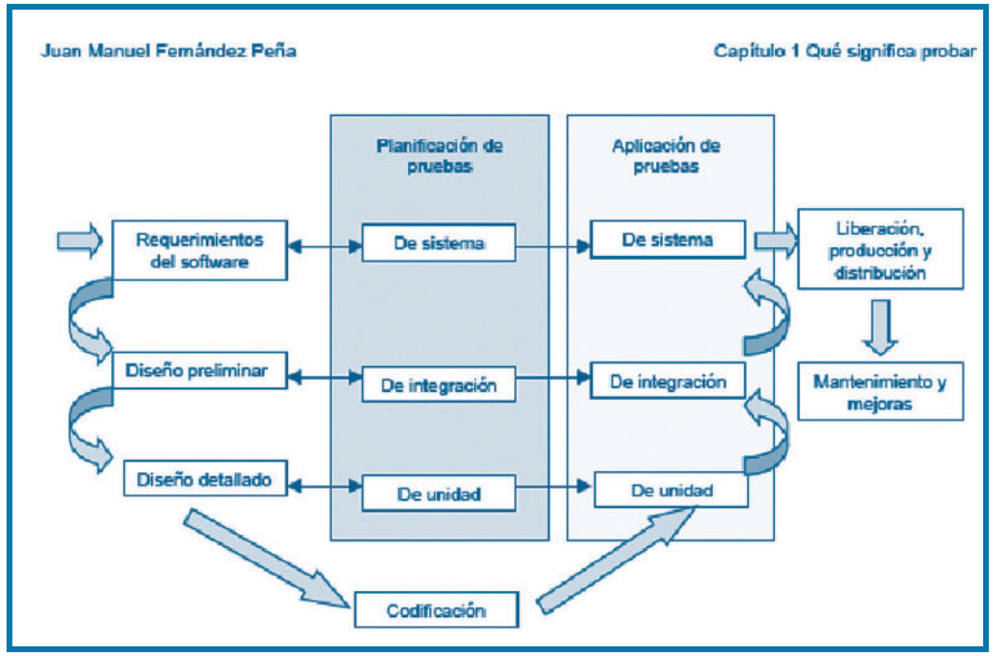

Figura 28. Niveles de pruebas.

Al consultar en internet se encuentran muchos cursos específicos con un tipo de prueba determinado como: técnicas para probar una base de datos en un motor de base de datos específico, pruebas en un lenguaje de programación seleccionado. También se observan postgrados y especializaciones que tienen en uno de sus módulos el tema del *testing*.

Las empresas del sector del *testing* se quejan de que los recién egresados muy poco conocen de pruebas y requieren un entrenamiento para poder pertenecer a un equipo de trabajo dedicado a las pruebas.

Para buscar la forma de incorporar en los contenidos temáticos de las técnicas de pruebas, se conformó un equipo de investigación de pruebas (al cual pertenezco) en las instituciones de la Alianza Futuro Digital. En la investigación se han logrado unos resultados parciales y han identificado tres formas de incorporar las pruebas.

1. Unas electivas que desarrollen las competencias del *testing*.

2. Incorporar actividades de pruebas a los sistemas creados en la asignatura.

3. Ofrecer la especialización en *testing*.

Se está analizando cuál de las tres alternativas se puede aplicar teniendo en cuenta que se desea formar desarrolladores con la habilidad para realizar pruebas, con unos roles definidos para el técnico, para el tecnólogo y para el profesional (ciclos de formación).

BIBLIOGRAFÍA

STEPHEN R., Shach. Análisis y diseño orientado a objetos con UML y el proceso unificado. México: McGraw-Hill, 2005, 450 p.

TORRES MUÑOZ, Alicia. Metodología del trabajo científico aplicada a la Ingeniería. Bogotá: Universidad Militar Nueva Granada, 2008. 303 p.

ARTÍCULO VII

LA AUTOEVALUACIÓN BASE DE UN PLAN DE MEJORAMIENTO CONTINUO Y DE CALIDAD

Jaime Alberto Acosta Gómez
Docente de Tiempo Completo
Tecnológico de Antioquia
jacosta@tdea.edu.co

RESUMEN

El proceso de autoevaluación indica la dirección y las metas por alcanzar en la administración y desarrollo de los programas académicos, con base en las múltiples variables que se desarrollan en un proceso de calidad. Esto nos permite conocer las fortalezas, debilidades y limitaciones para lograr los fines deseables en calidad académica y de reconocimiento de los programas en contexto. La autoevaluación posibilita situarnos en la realidad de los mismos; además, conocer los desafíos internos y externos para un proceso de mejoramiento a partir de la reflexión para planear la gestión, con procedimientos claramente definidos y estructurados, y con una impronta de calidad y desarrollo que dé continuidad y madurez a los programas. Dichos procesos deben ajustarse desde la concepción de la administración y la implementación de procesos académicos, vistos estos desde la operacionalización de los planes de estudio, los programas y los modelos pedagógicos respectivos.

Palabras clave: Autoevaluación, Calidad, Mejoramiento continuo.

Las transformaciones que vivimos producto de los procesos de la globalización inciden en la forma como se deben plantear las cosas en lo político, social, ambiental e institucional, y sobre todo en la forma como debemos mirar la educación. Ello implica la redefinición de los modelos educativos y la participación que deben asumir los actores académicos en los procesos de enseñanza- aprendizaje con miras a la calidad.

El compromiso de la Facultad de Informática es formar estudiantes con un perfil académico y profesional que les permita participar en asuntos de alto nivel especializado, con comprensión prospectiva de la relación del alumno con una sociedad que se dinamiza permanentemente y que trasciende y dinamiza la esfera tecnológica, específicamente en el uso de las tecnologías de la información y comunicación.

Los procesos académicos de los programas implícitamente promueven la reflexión y transformación de los planes de estudio, para responder a las exigencias de las situaciones propias de los tiempos actuales en materia de desarrollo de software, administración de los sistemas de información y automatización de procesos electrónicos.

Específicamente, para el programa Tecnología en Sistemas se desarrolló el convenio Alianza Futuro Digital, en el cual intervienen el sector productivo, la Universidad y el Estado, representado este ultimo por su liderazgo y coordinación en el Ministerio de Educación Nacional y la Secretaría de Educación Municipal, para brindar una dinámica pertinente en la formación del talento humano en buenas prácticas de desarrollo de software aplicando metodologías de procesos de calidad.

Lo anterior ha permitido tener una mirada diferente en el desarrollo curricular de los programas Técnico Profesional en Sistemas y Tecnología en Sistemas, vistos desde una perspectiva de formación que los sitúa como opciones académicas que han considerado los requerimientos y necesidades del sector productivo en el desarrollo de competencias vistas desde el hacer y saber hacer en cada una de las etapas del ciclo de vida del software y su respectiva fundamentación en la formación técnica y tecnológica.

Un proceso de autoevaluación indica la dirección y las metas para lograr en materia de la gestión curricular, la administración y el desarrollo de los programas académicos, nos permite conocer las fortalezas, debilidades y limitaciones que poseemos para alcanzar los fines deseables en calidad. Nos sitúa en la realidad de los mismos y nos revela los desafíos internos y externos para un proceso de mejoramiento a partir de la reflexión, para planear la gestión con procedimientos claramente definidos y estructurados con una impronta de calidad y desarrollo permanente.

Dichos procesos deben ajustarse desde la concepción de la administración y la implementación de procesos académicos, en el marco del desarrollo de los planes de estudio, programas y modelos pedagógicos diseñados para ser operados en el contexto de las estructuras académicas establecidas como políticas en la institución.

La autoevaluación se constituye entonces en un proceso de análisis del programa, realizado de manera participativa por los actores que lo conforman (estudiantes, profesores, egresados, empleadores, directivos entre otros; dando amplitud y precisión a los resultados obtenidos, a través de los diferentes medios de consulta e investigación), cuyo fin es emitir juicios acerca de la calidad educativa, en aras de su mejoramiento continuo y la acreditación.

Ello significa que la apreciación de la calidad debe ser muy objetiva en el sentido pleno de la palabra, con base en los estándares de calidad establecidos por el Consejo Nacional de Acreditación (CNA) y el liderazgo, consenso y participación que se deriva del proceso como tal con una esmerada articulación y planeación.

Esto lleva a plantear que la calidad para los programas de la Facultad no se traduce simplemente en un sistema de normas, sino en la incorporación de principios y la aceptación de referentes que están en relación con las expectativas, prospectivas y potencialidades de los mismos. Dichas potencialidades subyacen en la enseñanza caracterizada desde la internacionalización, el intercambio de conocimientos, la creación de modelos instruccionales interactivos y proyectos de investigación que nacen en el seno de cada programa, pero que toman connotaciones de mayor nivel e impacto.

Es pertinente resaltar que realizar procesos de autoevaluación permanentes en los programas de la Facultad de Informática es la carta de presentación para mejorar los procesos y dinamizar el aprendizaje en la estructura de cada uno de sus programas, proyectando la capacidad de acción para ofrecerlos en consonancia con los requerimientos del mundo en cuanto a la formación de programadores y/o desarrolladores de software, administradores de sistemas de información y tecnólogos en electrónica, que conocen todo el fundamento científico para analizar, describir y hacer mantenimiento a sistemas electrónicos de gran nivel. Es así como uno de los criterios de calidad de los programas de la Facultad de Informática consiste en su pertinencia con el contexto nacional e internacional, de ahí que los procesos de autoevaluación impliquen planes de acciones para desarrollar en un tiempo determinado, con un presupuesto y unos responsables de la gestión para desarrollar específicamente los procesos académicos con el fin de reelaborar una visión de los programas.

LISTA DE REFERENCIAS

Bernal, J.B. (1993). *La calidad: desafío que enfrenta la educación en el momento actual*. Proyecto UNESCO/Países Bajos 519/Cos/10 (SIMED), San José.

Consejo Nacional de Acreditación, CNA (2003). *Autoevaluación con fines de acreditación de programas de pregrado*. Bogotá: Corcas Editores Ltda.

Londoño Restrepo, Guillermo (1996). *Acreditación de instituciones y programas de Educación Superior*. Medellín.

ARTÍCULO VIII

LAS TECNOLOGÍAS DE LA INFORMACIÓN PLATAFORMA DE UNA CULTURA GLOBAL

Jaime Alberto Acosta Gómez
Docente de Tiempo Completo
Tecnológico de Antioquia
jacosta@tdea.edu.co

RESUMEN

Las tecnologías de la información y comunicación (TIC) cumplen un papel determinante en la producción y desarrollo social a nivel global. En países desarrollados se asumen como instrumentos que contribuyen al logro de los objetivos nacionales y prestan un soporte fundamental en la implementación de políticas nacionales y programas de desarrollo desde todo punto de vista. Las TIC son un factor determinante para afrontar problemas que surgen de la complejidad del mundo actual; esto ha llevado a cambiar radicalmente nuestra forma de vivir y ha creado una cultura de la información permanente.

Palabras clave: Comunicación, Comunidad global, Tecnología, Información.

El ser humano en su proceso de formación permanente desarrolla elementos críticos, actitudes y aptitudes enmarcadas en un esquema de sociedad de la información, que le exige en su quehacer cotidiano establecer permanentemente interacción con las tecnologías de la información,cuyo uso permite intercambiar conocimiento para adquirir relaciones de poder y competitividad.

Dichas relaciones se reflejan en la capacidad de conocer más de lo propio, con el fin de establecer nuevos mercados, y de esta forma, vender al mundo información e innovación y gestión, para introyectar nuevas posturas y referentes que hacen de la cultura un conjunto de informaciones, juicios, nuevas tradiciones e ideas que trascienden de lo local a lo global.

Las tecnologías de la información permiten compartir este recurso en todas las direcciones, así como conectarnos y comunicarnos con una comunidad global. De esto podremos obtener un progreso económico, fuerte y sostenido, democracias sustentadas en la participación, mejores soluciones a los problemas locales y mundiales. Obtendremos, además, un mayor sentido de integración en el planeta. Valga decirque las tecnologías de la información están generando sociedades y culturas competitivas, como es el caso de la India, que hoy por hoy es reconocida como uno los primeros países en el mundo en desarrollo de software;

no obstante, la infraestructura de la información global no se construye de un día para otro, sino durante muchos años de inversión en capital, conocimiento y mejoramiento continuo. Es aquí donde juegan un papel importante los esfuerzos que se realicen en investigación y desarrollo, para buscar y aplicar especializaciones específicas que soporten la infraestructura de la información globalmente accesible, que permitan transformar la sociedad y la cultura.

Es preocupante como los países atrasados o más pobres se ven afectados por estas relaciones de poder en cuanto al conocimiento, ya que aumentan las desigualdades, las crisis sociales y la violencia; es aquí entonces que la educación desempeña un papel preponderante, puesto que permite situar al ser humano en un contexto en el espacio y en el tiempo, con el fin de transformarlo en un ser universal que se desarrolla en una cultura global.

El concepto de rol de la información en nuestra sociedad debe ajustarse a una infraestructura del conocimiento para acercarse en algo a ser globales por naturaleza. Tendrá que ser usada conjuntamente con redes de telecomunicación que desarrollen una definición coherente de información global similar a la autopista de la información, que es vista como una red de computadores que facilita el acceso y recuperación de información para la toma de decisiones.

Por lo anterior, podemos pensar entonces que las exigencias tecnológicas que nos rodean en nuestro contexto en materia de acceso al recurso de la información llevan a que en la actualidad se establezcan nuevos modelos de cultura informatizada y global que permiten parámetros de desarrollo propios de cada región o país.

Es así como el ser humano requiere estar informado para poder analizar situaciones, hallar las soluciones a problemas administrativos y/o políticos, utilizar el juicio y la razón en el momento oportuno, para disminuir el grado de incertidumbre, para mejorar las decisiones que determinan mediante elecciones sucesivas la suerte de la organización, buscando siempre el posicionamiento y la competitividad.

El fuerte impulso de la tecnologías de la información ha transformado las realidades sociales, hay nuevas formas de ver el mundo y se han establecido nuevas categorías en cuanto a la visión del mismo. La nueva economía que se desarrolla en contextos nacionales o de nación pasó a ser informacional y global. La red de redes o la internet se ha convertido en pocos años en la columna de la comunicación ligada al conocimiento y a la creación de nuevas relaciones capital-trabajo.

A MANERA DE CONCLUSIÓN

La acentuación de la globalización tiene repercusiones económicas (unificación de los mercados), sociales (incrementos de las distancias y las exclusiones sociales), culturales (peligro para la identidad de los pueblos) y comunicacionales (penetración de tecnologías de la información y la comunicación).

sta diversidad de dimensiones de la globalización no hace más que poner en evidencia que este es un fenómeno muy complejo quesubraya la necesidad de que, con las TIC y las comunicaciones en general,se deben establecer nuevas formas de relación para ahondar en un proceso de globalización mediante el cual se introyecten espacios de cultura e identidad.

Finalmente, ha de entenderse con lo anterior que la globalización es un proceso paralelo a la identidad cultural, cruza todas las fronteras, no solo porque existen los instrumentos electrónicos y las tecnologías de la Información, sino porque permite esa instantaneidad de acceso a la información con mayor conocimiento.

LISTA DE REFERENCIAS

Toffler, A. (1982). *La tercera ola*. Barcelona: Plaza & Janés.

Gates, B. (1995). *Camino al futuro*. México: McGraw-Hill.

Castells, M. (1997).*La era de la información. Economía, sociedad y cultura*. Volumen 1, La sociedad red. Madrid:Alianza Editorial.

ARTÍCULO IX

METODOLOGÍA DE ARTICULACIÓN CON LA EDUCACIÓN MEDIA

Walter Gómez Torres
Coordinador Media Técnica
Tecnológico de Antioquia
waltergomeztorres@gmail.com

Fabio Alberto Vargas Agudelo
Magíster en Ingeniería de Sistemas
fvargas@tdea.edu.co

RESUMEN

La Facultad de Informática del Tecnológico de Antioquia ha implementado un modelo de articulación con la educación superior, que interviene los procesos académicos de las instituciones de educación media, favoreciendo la continuidad de los estudiantes en las cadenas de formación. El sistema consta de los subsistemas: contratación, acompañamiento, motivación, homologación y sistematización.

Palabras clave: Articulación, Cadena de formación, Educación media, Homologación.

INTRODUCCIÓN

El Tecnológico de Antioquia, fundamentado en la normatividad vigente y la experiencia en procesos de articulación, ha implementado un modelo de articulación de la Facultad de Informática con la educación superior; modelo que desde una concepción sistémica interviene los procesos académicos de las instituciones de educación media, generando dinámicas de calidad que permiten la homologación de asignaturas, tanto básicas y fundamentales como del área específica, favoreciendo la continuidad de los estudiantes en las cadenas de formación.

El modelo de "Articulación de la Media Técnica con la Facultad de Informática del Tecnológico de Antioquia" propende por la reflexión conjunta entre la educación superior y la educación media, de tal manera que se generen estrategias de implementación y seguimiento permanente a los procesos de enseñanza y aprendizaje en las instituciones educativas en busca de una articulación con calidad, pertinencia y coherencia.

La articulación de la media técnica con la educación superior como proceso de modernización de la educación, sirve como indicador de calidad y pertinencia a las necesidades que requiere el contexto, como también a la política del actual Gobierno denominada "Revolución Educativa".

La educación media técnica se ha desarrollado en el marco de la educación formal del nivel medio en los grados 10 y 11, subsiguientes a la educación básica secundaria. Permite a los estudiantes obtener una formación general y laboral que los ayude a adaptarse al cambio permanente de las necesidades

para ejercer e integrarse con éxito a las diferentes áreas de la actividad productiva y/o continuar estudios superiores. Se define como la etapa de formación del ser humano que sucede a la educación básica y precede a la educación superior. Tiene como propósito facilitar al joven una mejor comprensión de sí mismo, el desarrollo de competencias que le permitan enfrentarse a un mundo en constante cambio, acceder a la estructura productiva con una participación efectiva en una sociedad pluralista y democrática y continuar las cadenas de formación en la educación superior.

El modelo de articulación de la Facultad de Informática propende por una articulación entre la educación media y la educación superior con calidad y pertinencia; mediante el reconocimiento y homologación de saberes de áreas básicas y fundamentales como también de las áreas del conocimiento específico, reconocimiento que se logra a través de un acompañamiento directo del Tecnológico de Antioquia a las instituciones de educación en proceso de articulación, siempre respetando la autonomía de las instituciones y en procura de dejar capacidad instalada.

El proceso de acompañamiento supone acciones colectivas con el fin de dinamizar los componentes curriculares, metodológicos y didácticos para el logro de las competencias y salidas ocupacionales que requiere el estudiante de cada modalidad, para la inserción al mundo productivo y la continuación en la educación superior con el reconocimiento de sus saberes.

El modelo está concebido como un sistema con varios subsistemas que interactúan entre sí para el logro de una articulación con calidad y pertinencia, el sistema se ajusta tanto para estudiantes de grado décimo como para los de grado once, con incidencia en los demás grados de la institución desde el grado sexto.

SUBSISTEMAS DEL MODELO DE ARTICULACIÓN DE LA FACULTAD DE INFORMÁTICA

En general, la metodología del sistema de articulación se basa en el acompañamiento directoa las instituciones de educación; acompañamiento que realiza el recurso humano que hace parte de la estructura administrativa de la Facultad de Informática, en diferentes momentos e instancias.

El acercamiento a la comunidad educativa se realiza mediante contacto directo con directivos, docentes, estudiantes y padres de familia, con trabajo individual, grupal, y apoyado con material impreso y electrónico. El Sistema de Articulación consta de los siguientes subsistemas.

Subsistema de contratación

Este es el procedimiento de formalización del convenio de articulación entre el Tecnológico de Antioquia y la entidad contratante.

Subsistema de acompañamiento

Este proceso incluye la presencialidad de todo el recurso humano que se contrata por parte de la Facultad de Informática para desarrollar cada uno de los roles necesarios para la ejecución de las actividades. En general todas estas personas son docentes y comparten todo el proceso de formación en la institución educativa. Los roles son:docente articulador, docente de acompañamiento, docente de la institución educativa de la media, docente asesor área de matemáticas, docente asesor área de lenguaje, docente asesor proyecto pedagógico integrador y docente asesor área de tecnología.

Subsistema de motivación a la comunidad educativa

Es necesario que la comunidad educativa se apropie del proceso, que se identifique con él y lo sienta como suyo, por esa razón la importancia de involucrar en el proceso de la articulación de la media técnica a los diferentes actores en el proceso educativo, concienciando y contextualizando a todos de tal forma que se dé lugar al surgimiento de inquietudes que favorezcan el acercamiento cada vez mayor entre la institución educativa, el sector productivo y la institución de educación superior, en este caso el Tecnológico de Antioquia,para lograr el compromiso de todos los involucrados.

En los grados 10 y 11 se realiza una motivación constante a estudiantes y padres de familia, explicándoles las ventajas del proceso de articulación con la educación superior y la continuidad en las cadenas de formación. En este sentido, se llevan a cabo reuniones con los grupos y reuniones con los padres de familia en las cuales además de explicar en qué consiste la articulación, se les da a conocer las diferentes alternativas de crédito educativo proporcionadas por las entidades del Estado.

Es necesario que el Tecnológico de Antioquia se muestre como la primera alternativa de continuidad en la educación superior, por lo que se fortalecerá la participación institucional en las instituciones de la media técnica.Las acciones más relevantes para la motivación a la comunidad educativa son:

- Reunión con docentes
- Reunión con directivos docentes
- Mesa de trabajo institucional
- Reunión con estudiantes
- Reunión con padres de familia

En procura de que los estudiantes de noveno grado realicen una buena elección de la media técnica al momento de ingresar al grado 10 es necesario realizar un proceso constante de orientación vocacional durante todo el año, procurando la mayor claridad al momento de la elección de la especialidad en la media. Esto generará mayor identidad con la modalidad elegida, garantizará mejor rendimiento académico y menor nivel de deserción.

Este proceso incluye como aspectos más importantes la información a los estudiantes de las profesiones pertinentes en el contexto actual, los diferentes perfiles ocupacionales y la identificación de sus propias competencias. Este trabajo será apoyado por la cartilla número tres de Educación Pertinente de la Secretaría de Educación Municipal.Los entregables de este subsistema son:

- Actas de las reuniones con docentes y directivos docentes, con los compromisos.
- Presentaciones realizadas a estudiantes, docentes y padres de familia con listados de asistencia.

Subsistema de homologación

Junto con el aumento de la calidad académica y la formación de competencias en el estudiante, el objeto final del Modelo de Articulación de la Media Técnica de la Facultad de Informática con la Educación Superior es la consolidación de todo el esfuerzo realizado por la comunidad educativa de la institución de educación media y del equipo de la media técnica de la Facultad de Informática.

La Facultad de Informática, basada en el acompañamiento y control que realice a los procesos de enseñanza y aprendizaje en las instituciones educativas de media técnica, hará el reconocimiento y homologación de los módulos correspondientes al primer y segundo semestre del ciclo Técnico Profesional en Sistemasde la Facultad de Informática, sujeto a la aprobación del Comité Curricular y el Consejo de Facultad, basándose en la malla curricular.

Subsistema de sistematización

La documentación de la experiencia de una manera sistemática permitirá la construcción de un instrumento rico en evidencias que pueden servir como referencia a otros procesos similares, la extracción de categorías de análisis que susciten procesos investigativos a nivel institucional o de entidades interesadas en el tema.

La sistematización se realiza desde los insumos y productos obtenidos en las diferentes acciones articuladoras propuestas por el modelo, donde se deja evidencia de su ejecución y resultados. Es importante dejar registro de acciones como: reuniones, mesas de trabajo, cronogramas, planes de mejoramiento, diarios de campo, informes de gestión, agendas, evidencias fotográficas, entre otros, que a bien tenga cada institución.Los entregables en este subsistema son:

- Carpeta de evidencias en cada una de las instituciones del convenio con los entregables de cada uno de los subsistemas y del proceso de evaluación y seguimiento.
- Informe final de articulación.

BIBLIOGRAFÍA

ARGÜELLES, Antonio, comp. Competencia laboral y educación basada en normas de competencia. México: Limusa, 1996.

MOREIRA, M. A. Aprendizaje significativo: teoría y práctica. Madrid: Visor Dis., S. A., 2000.

TECNOLÓGICO DE ANTIOQUIA, Manual metodológico: "Diseño de mallas curriculares y microcurrículos para el desarrollo de competencias". Medellín: Vicerrectoría Académica, 2005.

TECNOLÓGICO DE ANTIOQUIA. Términos de referencia "Convocatoria Para Apoyar Proyectos de Transformación de la Formación Técnica y Tecnológica", Medellín, 2005.

TECNOLÓGICO DE ANTIOQUIA.Articulación de la Media Técnica con la Educación Superior-Modelo Técnico Pedagógico.Medellín. 2005

FIGURAS

TABLAS