

# PROCESOS EVALUATIVOS EN EL DESARROLLO DE COMPETENCIAS DE DISEÑO EN INGENIERÍA DE SISTEMAS.

Adriana Rocío Lizcano Dallos, Ricardo Vicente Jaime Vivas  
alizcano@udi.edu.co, ricardojaime@udi.edu.co  
Docentes de la Escuela de Ingeniería de Sistemas  
Corporación Universitaria de Investigación y Desarrollo

## Resumen

Este artículo presenta los elementos conceptuales, pedagógicos y metodológicos que se han implementado para superar las deficiencias que se presentan en los cursos del área de programación, bases de datos y sistemas de información del programa de Ingeniería de sistemas. La problemática se asume desde el planteamiento del diseño como competencia fundamental en la ingeniería, mostrando las manifestaciones de la problemática en cada una de las áreas, la descripción de las mejoras en los procesos de evaluación adoptadas, con el fin de incorporarlos efectivamente dentro del aprendizaje y algunos de los resultados observados en su implementación.

## 1 Dificultades inherentes al diseño en Ingeniería de Sistemas

La formación en ingeniería comprende el estudio de las matemáticas y las ciencias naturales, y con ellas diversas maneras de utilizar recursos en la resolución de problemas para el progreso social. La práctica de la ingeniería consiste en la apreciación de los problemas y los proyectos sociales, y la selección del método más conveniente para su abordaje, de acuerdo a condiciones específicas de alcance, tiempo, costo, disponibilidad de recursos, entre otras (Wright, 2004, pág. 23).

El enfoque de diseño es propio de la disciplina de la ingeniería, en cuanto permite la evaluación crítica de una solución propuesta, su comparación con otras, y la selección con sustentos cualitativos y cuantitativos de la que finalmente deba ser implementada, es decir, llevada a la realidad. Con este enfoque se busca dar un uso racional y eficiente a los recursos, que suelen ser limitados.

La expresión típica del diseño son los planos, que sirven a la vez como abstracción de la propuesta de solución, y como medio para calcular los costos, tiempos y alcances de la misma. Un diseño, entonces, es la organización de información con respecto a la comprensión del problema, el tipo de recursos a utilizar, y la forma en que estos serán incorporados a la solución.

En el caso de las de más larga tradición, como la Ingeniería Civil, la Ingeniería Mecánica, y la Ingeniería Eléctrica, el desenvolvimiento de un diseño implica operaciones de extracción de materiales, transporte y construcción, perceptibles para cualquier observador. Las obras tienen una existencia tangible, concreta, ocupan un lugar en el espacio. Aunque se requiere la formación en Ingeniería para estar en capacidad de diseñar, la expresión del diseño puede ser hasta cierto punto comprendida por personas sin dicha formación.

No sucede lo mismo en Ingeniería de Sistemas, pues sus proyectos son de alguna manera invisibles, en cuanto a que no tienen una existencia material perceptible, sino que siguen siendo información. Ningún observador nota progresos en su ejecución, y la interacción del usuario es a través de la interfaz, que ha sido implementada para facilitar el uso y ocultar el funcionamiento interno. Los planos en Ingeniería de Sistemas, son planos de algo que al convertirse en realidad seguirá siendo abstracto, y que los observadores sin formación no perciben.

## 2 Diseño en el currículo de Ingeniería de Sistemas

Los cursos relacionados con diseño de algoritmos, de bases de datos y de sistemas de información, han sido considerados tradicionalmente como los más difíciles de superar por parte de los estudiantes, lo que se puede corroborar al revisar las cifras de mortalidad académica, que son comparativamente más altas en las áreas de programación, algorítmica y bases de datos, con respecto a las demás áreas del programa de Ingeniería de Sistemas. De esta problemática dan cuenta algunas experiencias nacionales (Villalobos, Casallas y Marcos, 2005; Bohorquez y Amaya, 2008 ) e internacionales (White, 2006; Van Gorp y Grissom, 2001), que consideran que estas áreas constituyen los cursos más “duros” para los estudiantes.

Estas áreas tienen en común el enfoque hacia el diseño de soluciones que permitan resolver determinados requerimientos de procesamiento de la información. Estos procesos requieren de una serie de habilidades cognitivas complejas, además de la capacidad para representar y argumentar adecuadamente las soluciones propuestas, elementos fundamentales en la formación profesional, e igualmente en el ámbito laboral. Aún cuando el desempeño laboral no se oriente específicamente a la programación de computadores ni al desarrollo de bases de datos, la necesidad de formular diseños para resolver los problemas constituye un componente que se diría omnipresente en el desempeño laboral.

A partir de esta problemática se plantea la necesidad de estudiar los enfoques pedagógicos y reformular las formas de evaluación que se han venido aplicando tradicionalmente en los cursos de Ingeniería de Sistemas. Este artículo muestra algunas de las experiencias de evaluación implementadas en cursos de las áreas de Programación y Algorítmica, y Bases de Datos, con el fin de mejorar los procesos de aprendizaje llevados a cabo en el programa de Ingeniería de Sistemas de la Universitaria de Investigación y Desarrollo - UDI.

### **3 El enfoque constructivista en el aprendizaje del diseño en Ingeniería de Sistemas**

Las implicaciones específicas ya mencionadas del diseño en Ingeniería de Sistemas, generan los siguientes factores (Connolly & Begg, 2006):

- Los estudiantes pueden aprender a diseñar, pero esta habilidad no se puede enseñar de forma didáctica ni discursiva, sino que debe aprenderse a través de operaciones prácticas.
- La habilidad para el diseño es de carácter holístico, es decir, no puede ser dividida ni aprendida por partes, sino que se vive en la experiencia.
- La habilidad para el diseño depende de la capacidad para el reconocimiento cualitativo del mundo, que no puede ser descrito por alguien que enseña, sino aprendido por el estudiante en la acción.
- El diseño es un proceso creativo, en el que continuamente el diseñador cambia su manera de reconocer y hacer cosas.

Estos factores hacen que surja la necesidad de aplicar el enfoque constructivista para el logro de competencias en diseño, a través de modelos pedagógicos como el razonamiento basado en casos de estudio, aprendizaje basado en problemas, aprendizaje colaborativo, juego de roles, entre otros. Pero el enfoque constructivista en Ingeniería de Sistemas enfrenta obstáculos diversos en el contexto actual : por parte de los docentes; relativos a los recursos; relativos a la preparación para la educación superior; y obstáculos por parte de los estudiantes.

#### **3.1 Obstáculos por parte de los docentes**

Una cantidad importante de los docentes en ejercicio en Ingeniería de Sistemas, son a su vez ingenieros sin preparación formal en fundamentos pedagógicos, que hicieron sus estudios bajo un enfoque conductista, en ambientes en los que la actividad predominante era la clase magistral sobre aspectos teóricos, y en que ellos como estudiantes debieron, según sus propias apreciaciones, hacer un gran esfuerzo para aprender. Por eso, no es raro escuchar opiniones entre los docentes con respecto a las actuales generaciones de estudiantes, en

cuanto a que estos no parecen estar dispuestos a hacer un esfuerzo similar al de aquellos, y por el contrario, parecen querer todo a la mano. Y la consecuencia es una práctica equívoca del enfoque constructivista, que deposita la responsabilidad por la lectura previa del tema de clase al estudiante, con un docente que en algunos casos se limita a contestar preguntas; se descuida la ampliación en aquello sobre lo que el estudiante no pregunta, como mecanismo de presión para obligarlos a asumir la responsabilidad. Y la clase magistral, entendida como aquella en que el docente asume un marcado protagonismo, es criticada como arcaica.

### **3.2 Obstáculos relativos a los recursos**

Es de anotar que hay otras explicaciones a tal diferencia de actitud entre los estudiantes actuales y la que los docentes tuvieron en su oportunidad. El computador era un recurso muy escaso aún en la última década del siglo XX; finalizando la primera década del siglo XXI, no solo las instituciones educativas pueden ofrecer mayor cantidad de computadores por estudiante, sino que el uso de computadores portátiles es muy común incluso en la población estudiantil. Y en todo sistema, a la abundancia de recursos suele seguirla un menor cuidado en su explotación y aprovechamiento. Mientras antes había que preparar cuidadosamente (diseñar) lo que se iba a desarrollar en la sala de cómputo en una o dos horas semanales por estudiante, ahora, con el recurso disponible para uso casi individual, se tiende a improvisar directamente sobre el teclado, a programar o a crear tablas de bases de datos sobre la marcha, y a eludir el trabajo de diseño. Los productos de diseño no llegan a ser la expresión de una propuesta, sino la de un resultado, algo que no es coherente con la definición misma de ingeniería.

### **3.3 Obstáculos relativos a la preparación para la educación superior**

Según plantea White (2006) la investigación muestra que la programación procedimental y la programación orientada a objetos requieren de altos niveles de desarrollo cognitivo (White menciona a Cafolla, 1987; Fletcher, 1984; Gibbons, 1995; Ignatuk, 1986; Little, 1984; Monfort y otros, 1990; Ott, 1989; Sein & Bostro., 1989; Wu, 1993; White, 2002), de manera que los estudiantes con bajos niveles cognitivos tienen dificultades para el aprendizaje de la programación.

Las investigaciones también muestran que la programación procedimental involucra habilidades cognitivas altas, tales como las operaciones formales planteadas por Piaget (Dalbey & Linn, 1985; Hudak & Anderson, 1990), llegando a la conclusión que el nivel de desarrollo cognitivo (según el planteamiento de Piaget) alcanzado por un estudiante, es un predictor del desempeño en programación procedimental. Little (1984) y Hudak y Anderson (1990) encuentran que los estudiantes que eran pensadores en operaciones formales según Piaget, puntuaban más alto en programación y en pensamiento lógico, que los que se encontraban en las etapas concreta o en transición. Los estudiantes con un desempeño cognitivo alto tenían un estilo de aprendizaje más abstracto, que le permitía aprender a programar.

De igual manera, varios estudios han identificado el uso de la lógica en las ciencias computacionales (Galton, 1992; Gibbs & Tucker, 1986; Myer, 1990; Sperschneider & Antoniou, 1991). Entre algunos aspectos se encuentran la lógica proposicional, ya que las tablas de verdad se ven reflejadas en la programación de sentencias "if". Adicionalmente se ha mostrado que un test plantado por Piaget para medir las tareas de lógica (PLT- Propositional Logic Test) puede predecir el éxito en la programación procedimental y en la programación orientada a objetos (Stager-Snow, 1985; White, 2002; White & Sivitanides, 2005), así como en la programación visual (White & Ploeger, 2004). Estos estudios han servido en Estados Unidos para ajustar el uso de lenguajes de programación en diferentes etapas del sistema escolar.

En Colombia, la incorporación de lenguajes de programación en niveles previos al universitario es aún incipiente y el desempeño mostrado por los estudiantes de últimos grados de educación secundaria en cuanto a pensamiento abstracto sigue siendo entre medio y medio-bajo. Los bajos niveles en habilidades lectoescritoras y de autonomía, una deficiente motivación

intrínseca, sumados al cambio de contexto del colegio a la universidad, conforman un panorama que permite explicar la gran deserción en los primeros niveles de educación superior, especialmente en los programas de ingeniería.

### **3.4 Obstáculos por parte de los estudiantes**

Por parte de los estudiantes, las dificultades para la puesta en práctica del enfoque constructivista son, entre otras (Connolly & Begg, 2006) :

- A medida que se comprende el problema, la complejidad aumenta en lugar de disminuir.
- Se requieren habilidades metacognitivas por parte del estudiante, es decir, una comprensión de su propio proceso de aprendizaje.
- Se necesita un amplio bagaje, para ser efectivo en un ambiente de resolución de problemas.

Este último punto es apoyado por autores que afirman que el diseño es una actividad intensiva en conocimiento, para la cual el diseñador utiliza conocimiento interconectado, parte del cual no es relativo a la tecnología que utiliza sino a los campos en donde se aplica, conocimiento variado sobre el que se construyen modelos mentales para resolución de problemas, se refuerza la capacidad para hacer inferencias, por lo cual el proceso de diseño en sí mismo es también una actividad de aprendizaje, un acto social, creativo, e incluso emocional (Mishra, Zhao, & Tan, 2000).

Todo esto determina que el diseño está más relacionado con la síntesis que con el análisis, lo que no es muy común en nuestro contexto educativo. A lo que se debe agregar que tampoco son nuestros estudiantes muy dados a filosofar, y su nivel de cultura general no es bueno.

## **4 Manifestaciones de la Problemática**

### **4.1 Diseño en Programación**

La problemática planteada anteriormente se manifiesta en dificultades durante los cursos de programación, entre ellas, la manipulación de los datos como elementos abstractos, cuya representación computacional está muy alejada del ámbito físico al que está acostumbrado el estudiante en el pensamiento concreto. Esta dificultad se incrementa con el transcurrir de los diferentes cursos, ya que van apareciendo cada vez estructuras de datos más abstractas y complejas.

El uso continuo de reglas para la aplicación de cada una de las estructuras básicas en la construcción de algoritmos hace que el estudiante requiera procesos de memorización a los cuales ya se ha desacostumbrado. En los procesos educativos se ha generado el mito de que la memorización es inadecuada y anacrónica, pero en este caso particular, la memorización de estas reglas hace expedita una parte del camino hacia el logro de la solución.

La preocupación por la sintaxis del lenguaje más que por la solución planteada, hace que el estudiante se preocupe más porque el editor de programas no le señale errores en su código, sin importar si los cambios que está realizando tienen sentido en la solución de problema.

El interés del estudiante por desarrollar las habilidades para analizar problemas y diseñar soluciones, se ve opacado por el interés de aprender distintos lenguajes, debido a que el estudiante recibe a través de internet un sinnúmero de noticias sobre diferentes lenguajes y se cuestiona acerca de si será capaz de aprender a utilizarlos algún día.

La ambición de observar lo más rápidamente posible programas con ventanas complejas que respondan a un sinnúmero de acciones, los llevan a sentirse frustrados ante el énfasis en el diseño de la solución o de la estructura de datos, más que en la codificación de la misma.

## **4.2 Diseño de Bases de Datos**

El modelo relacional, propuesto originalmente en 1976, es el que domina el panorama actual de las bases de datos. En la actualidad, las bases de datos son el componente principal de los sistemas de información. Su diseño implica la conceptualización acerca de un fenómeno, y la creación del esquema en que esta información será almacenada, con el propósito de minimizar espacio de almacenamiento, evitar redundancias, y posibilitar la recuperación con diferentes propósitos.

En el proceso de aprendizaje, el bagaje que un estudiante tenga sobre un tema, hace posible que dicho tema sea utilizado como área de aplicación, y por tanto, como facilitador de la comprensión de las técnicas de diseño de bases de datos. Asimismo, una vez comprendidas estas técnicas, el proceso de diseño de una base de datos se convierte en facilitador del aprendizaje en otras áreas del conocimiento.

Una problemática notoria en los cursos de bases de datos, habitualmente entre el segundo y tercer año del plan de estudios, es el escaso interés y conocimiento por parte de los estudiantes hacia temas no directamente relacionados con la carrera. Esto dificulta la labor docente, por cuanto no se tiene tal vehículo para acercar al estudiante con las técnicas de diseño. Cada caso de estudio planteado por el docente, sirve a un grupo pequeño de estudiantes por coincidencia de interés, pero no significa nada para el resto, y por tanto tiene un efecto adverso, ya que genera la impresión de que no poder entender el tema de aplicación, equivale a no poder entender las técnicas.

Una forma de dar alguna seguridad al estudiante sobre lo que se le pide que haga, es darle como requerimiento para el desarrollo, un reporte impreso predeterminado, como por ejemplo, una factura para que diseñe una base de datos que permita la facturación. Pero así como un documento base le da información concreta al estudiante, también lo limita a cumplir únicamente los requerimientos de impresión de resultados, que no son los únicos de una base de datos. Esto en el futuro ejercicio profesional, se traduce en que los diseños de las bases de datos son altamente limitados por los modelos de formularios preestablecidos, y no se plantean en los diseños innovaciones organizacionales, con lo cual el Ingeniero de Sistemas no desarrolla tampoco un sentido de liderazgo que le permita influir en los procesos de toma de decisiones en las organizaciones.

Al igual que sucede en el área de programación, en bases de datos los estudiantes manifiestan una urgencia por “dominar” manejadores específicos de bases de datos (Oracle, SQL Server, MySQL, y otros), esperando en forma errada que la herramienta los conduzca a un progreso en el diseño de bases de datos. Sin embargo, esto es como esperar que el uso de un procesador de texto, capacite a quien lo hace para generar cientos de páginas de buen material escrito. Y se presentan altos niveles de frustración por cuanto, al estar los manejadores de datos más robustos orientados a garantizar la seguridad de los datos, son del todo inflexibles con los descuidos en su diseño, y el estudiante termina por enfrentar la dura realidad, de que si no diseña no logrará que el manejador de bases de datos le permita el más mínimo manejo de los mismos.

## **5 La evaluación como evento generador de aprendizaje**

Para enfrentar la problemática antes descrita, se propone un cambio en el proceso evaluativo, con el fin de que la evaluación pase de ser un momento traumático en el que el estudiante debe esforzarse por mostrar resultados cercanos a la “respuesta correcta”, y se convierta en un momento pedagógico en que estando a prueba, el estudiante puede afianzarse en lo que ha aprendido y tener argumentos para mejorar lo que aún no comprende.

Un primer cambio consiste en que la evaluación debe hacer parte del aprendizaje. Es decir, la concepción del momento de evaluación como un espacio adicional para compartir, preguntar y opinar acerca de las deficiencias que se han detectado en el desarrollo de un eje temático.

Es importante la realimentación y la motivación en la evaluación. Aunque los resultados pueden ser en algunos casos verdaderamente devastadores tanto para los estudiantes como para el docente, es fundamental mantener una actitud positiva hacia el proceso, proporcionando opciones para mejorar los resultados involucrando una mayor dedicación y compromiso por parte del estudiante, y si es necesario un acercamiento individual del docente para facilitar la superación de las brechas en el conocimiento y la consecución de las habilidades del estudiante. Los investigadores del aprendizaje han encontrado reiteradamente que motivar a los estudiantes es fundamental para lograr un aprendizaje profundo que permanezca para su aplicación en diferentes situaciones (Guzdial y Soloway, 2002).

Debe darse claridad acerca de los criterios de evaluación en cada una de las actividades, lo cual facilita al estudiante enfocar su atención en los puntos que son igualmente claves para el docente. Aunque esto exige al docente una mayor definición de sus actividades de clase, también le facilita posteriormente el proceso de calificación y constituye una guía importante para el estudiante.

Se requiere buscar un equilibrio entre el trabajo individual y el colaborativo. Si bien el trabajo en colaboración con sus pares permite a los estudiantes aportar activamente en la construcción de una solución, intercambiar opiniones y hasta fortalecer sus conocimientos al explicar a sus compañeros, las habilidades para definir y explicar un diseño son esencialmente un proceso que se debe desarrollar a nivel individual. Aunque se intente formular procesos metodológicos de análisis comunes, los estilos de aprendizaje, la motivación interna y los conocimientos culturales previos del individuo definen la forma como aborda la solución, por esto las experiencias individuales constituyen la oportunidad de afianzar las estrategias cognitivas propias que permitirán acometer exitosamente procesos de diseño.

También se deben incorporar al proceso evaluativo problemas amplios que permitan ejemplificar diferentes ejes temáticos, con acercamiento a situaciones del mundo real. El logro de la autonomía en el estudiante se visualiza como un proceso gradual, que va desde problemas completamente definidos y cerrados, hacia problemas cada vez más abiertos, donde el estudiante deba tomar decisiones acerca de lo que es relevante para el diseño. De igual manera, aunque la situación planteada no sea de su ámbito de conocimiento, reconozca las actividades que le permitirán aprender acerca de ella y aplicar los conocimientos adquiridos de forma contextualizada.

## **6 Cambios en el proceso evaluativo en el corto plazo**

### **6.1 Diseño en Programación**

En esta sección se hará referencia a la incorporación de algunas estrategias de evaluación en cursos de programación de computadores, correspondientes al primer, segundo y tercer semestre de ingeniería de sistemas.

#### **6.1.1 Simulacros de parcial**

Un primer aspecto involucrado desde hace 6 semestres es la incorporación de simulacros de parciales, en fecha anterior al día definido para la evaluación propiamente dicha. Los simulacros de parciales corresponden a problemas que se resuelven en su mayoría en pequeños grupos con asesoría del docente y que guardan el mismo tipo de preguntas y puntaje de lo que será la evaluación. La idea del planteamiento de simulacros nace al observar el grado de tensión e incertidumbre que genera el parcial y de la necesidad de incentivar la preparación del estudiante con la suficiente antelación. Los estudiantes planteaban que no sabían el tipo de preguntas o ejercicios que constituirían el parcial y esto les generaba mucha tensión. Por otra parte, se observaba que los estudiantes esperaban hasta el día anterior para repasar los ejercicios realizados en clases y mirar los apuntes de su cuaderno, lo cual es insuficiente para obtener el desarrollo de las habilidades, si no se ha hecho el esfuerzo continuo de plantear soluciones a diferentes problemas.

Aunque los puntajes obtenidos en el simulacro no presentan diferencias significativas con respecto a las calificaciones obtenidas en la evaluación, el trabajo en grupo, la solución de los ejercicios propuestos en el mismo y el acompañamiento que se realiza por parte del docente, ayuda a superar las tensiones e incertidumbre a los que se ven abocados los estudiantes en la evaluación.

### **6.1.2 Realimentación de resultados y motivación**

En las diferentes actividades de evaluación el estudiante requiere obtener información acerca de lo que hizo bien y los errores que cometió. Un símbolo de X sobre una respuesta no es suficiente para hacer patente el proceso o la acción que realizó de manera incorrecta, que es en últimas lo que se pretende evitar. Por esta razón se han involucrado mensajes de realimentación que le proporcionan mayor información al estudiante, además de hacer completamente necesaria la explicación acerca de la solución mínima esperada por el docente.

El momento de entrega de los resultados puede llegar a ser desalentador para el estudiante, que en algunos casos piensa que sin importar los esfuerzos que realice, no va a lograr alcanzar el nivel deseado, por esta razón es muy importante hacer énfasis al estudiante en sus aspectos motivacionales.

Sobre este aspecto, en una experiencia realizada durante el primer semestre académico de 2008, se aplicaron mensajes de motivación sistemáticos a estudiantes del primer curso de programación en Ingeniería Electrónica, resaltando la importancia de lograr las metas personales y de vida que se han planteado al ingresar a la universidad, haciendo énfasis en todas las posibilidades que se les proporcionan para superar los inconvenientes que se les presentaran y en la confianza que el docente tiene acerca de las capacidades de todos sus estudiantes. Los datos mostraron una disminución del nivel de deserción en ese primer semestre, que se había mantenido durante varios semestres alrededor del 20%, hacia niveles del 10%. Adicionalmente se observaron cambios a nivel actitudinal de los estudiantes, una mayor empatía y confianza con el docente, generando espacios de aprendizaje más amenos y productivos.

### **6.1.3 Trabajos de programación.**

Los trabajos de programación corresponden a la solución de problemas en algunos casos planteados por el docente y en otros casos seleccionados por el mismo estudiante. Estos trabajos se desarrollan en dos fases, una individual y una grupal. En la fase individual se espera que el estudiante intente por su cuenta asumir el diseño de la solución, una vez cada estudiante ha obtenido una aproximación, se asigna a un grupo de trabajo en el cual se discute y defienden las diferentes soluciones para encontrar aquella que proporciona mayor seguridad de su eficacia al grupo de trabajo. Al finalizar el desarrollo, se realiza la socialización de los resultados obtenidos, de manera que cada uno de los grupos de trabajo defiende el diseño obtenido, recibiendo realimentación del proceso, tanto de sus compañeros de curso, como del docente.

## **6.2 Diseño de Bases de Datos**

El diagrama relacional, como producto de diseño, es el objetivo central de la evaluación en el curso de Base de Datos durante la primera fase; la segunda tiene como objetivo adicional, la puesta a prueba de tal diseño, mediante la verificación de la posibilidad de hacer consultas útiles sobre el mismo.

Como se mencionó antes, la información a partir de la cual el estudiante en su evaluación abordaba el diseño para solucionar un problema, usualmente consistía en formatos predefinidos de algún sistema de información ya existente que el estudiante debía emular. Sin embargo, el hecho de intentar una emulación, también restringe las posibilidades creativas para el diseñador, ya que estando circunscrita la solución a una salida predeterminada, son pocas las variaciones que se pueden formular.

Con algunos estudiantes se implementó una prueba evaluativa diferente, en la que se les entregaba un planteamiento bastante amplio, que primero debían interpretar para establecer los requerimientos, lo que no solo permitía sino que obligaba al acto creativo. El planteamiento implicaba que fuera poco probable llegar a una solución completa en las dos horas de duración del parcial, y esto en principio fue protestado por los estudiantes, que asumían que esta situación desembocaría necesariamente en bajas notas. Pero en la primera ocasión se debatió con respecto a las implicaciones favorables que podían encontrar, como que al tratarse de una temática amplia, no había un único camino de aproximación a la solución, ni un único punto inicial de abordaje de la misma; en el conjunto de soluciones propuestas por los estudiantes, había diferencias no excluyentes sino complementarias, con lo que se construyeron conocimientos mejor sustentados y el tema de la evaluación siguió siendo de discusión entre ellos durante dos semanas.

Aunque los resultados obtenidos en las dos horas se tomaron como indicador de la comprensión que en ese momento tenía cada uno sobre el diseño de bases de datos, y constituían un punto de partida para la nota final, se extendió el plazo y la evaluación salió del aula, por un tiempo prudente en que cada uno perfeccionaba la solución que había iniciado, o encontraba las razones por las cuales su enfoque había sido equivocado.

Finalmente, el docente presentaba una solución, obviamente más completa que las de los estudiantes y que había sido preparada con antelación. Sobre esta propuesta, se procuraba identificar aquello que los estudiantes habían logrado representar. Del mismo modo se sustentaba con la ayuda de esta solución, el por qué no se consideraban convenientes algunas propuestas, a partir de criterios técnicos pero valorando el esfuerzo conceptual del estudiante.

Para esta prueba se partió de una experiencia previa con los cursos del área de Sistemas y Organizaciones, que tradicionalmente se enfocan al aprendizaje de conceptos teóricos y la construcción de modelos de simulación por modelamiento cuantitativo, pero que durante los últimos semestres había sido complementado con un enfoque de modelamiento cualitativo, en el que se resalta que en la Ingeniería de Sistemas, haciendo uso de lenguajes que provee la Dinámica de Sistemas, pueden desplegarse procesos permanentes de aprendizaje con consecuencias favorables en el uso innovador de la informática.

Por el tamaño del curso de Bases de Datos, esta no pasó de ser una prueba exploratoria que no puede tomarse como experimento formal; pero tuvo resultados satisfactorios que abrieron el camino a una propuesta de investigación que se presenta a continuación.

## **7 Investigación alrededor del proceso evaluativo en el mediano plazo**

Para el segundo semestre académico de 2008, se presenta en la UDI un mayor esfuerzo para fortalecer la investigación, que se constituyó en el marco propicio para desarrollar las ideas sobre cambios en el proceso de evaluación, a través de un proyecto de investigación.

Con este proyecto se aborda como problema el hecho de que el rendimiento actual en el curso Bases de Datos I en la UDI es deficiente. Para lograr buena calidad, el docente debe repetir temas y el programa académico se hace lento. Si no es suficiente, se da una mortalidad académica que genera repitencia, deserción, o desistimiento de la carrera. Algunos estudiantes que aprueban el curso con el mínimo de competencia, tienen problemas en los cursos de Bases de Datos II, Sistemas de Información I y II, y otros; en el mediano plazo en el trabajo de grado; y en el largo plazo en su desempeño profesional.

Las causas del problema son: metodología de enseñanza inadecuada al perfil estudiantil; bajo nivel de cultura general de los estudiantes, que dificulta al docente el hallazgo de temas de interés para generar casos de estudio; carencia de software para este fin; y metodologías inadecuadas de estudio de los estudiantes.

La investigación dará respuestas con validez metodológica y pedagógica a estas preguntas: ¿cómo se afecta el desempeño de los estudiantes en diseño de bases de datos, con el uso de

casos de estudio mediados por un software?; ¿qué diferencias de resultados hay entre estudiantes con estilo cognitivo dependiente e independiente de campo?

Se busca implementar estrategias pedagógicas para el mejoramiento del rendimiento académico en el curso de Bases de datos I de la UDI, utilizando el enfoque de razonamiento basado en casos y aprendizaje mediado por herramientas tecnológicas, cumpliendo con los siguientes objetivos específicos :

- Desarrollar un software que proporcione opciones para documentar problemas de diseño de bases de datos relacionales, generación de las tablas que intervienen en el diseño, una interfaz genérica para el ingreso de registros a las mismas y una interfaz para la construcción de consultas SQL, como apoyo al desarrollo de competencias en el curso de Bases de Datos I.
- Implementar una estrategia pedagógica que facilite la incorporación del software dentro del curso de Bases de datos I, fundamentado en el razonamiento basado en casos.
- Realizar un análisis comparativo del desempeño de los estudiantes en la solución de problemas de diseño de bases de datos, considerando los estilos cognitivos de dependencia-independencia de campo.

En el desarrollo institucional de la UDI este proyecto busca disminuir los niveles de repitencia y deserción en Ingeniería de Sistemas, y mejorar el nivel de la docencia desde tres perspectivas: innovación y desarrollo tecnológico con la producción de material didáctico; solución de un problema contextual institucional con la identificación de especificidades del perfil estudiantil en la carrera; y generación de conocimiento con el desarrollo experimental de una pedagogía a partir del razonamiento basado en casos.

En cuanto a proyección nacional, la UDI puede aportar elementos a la discusión que sustenta el desarrollo de las TIC, en el que habitualmente prevalecen aspectos técnicos de hardware y software. Los resultados de este proyecto agregarían consideraciones organizacionales, en este caso de la organización educativa, para conseguir competencias laborales del Ingeniero de Sistemas en identificación de problemas y comprensión del impacto organizacional de las TIC, potenciando la carrera como sector económico promisorio por la capacidad de sus profesionales para abordar problemas organizacionales complejos.

En varios estudios nacionales (Hederich y Camargo, 1993, 1995 y 1998) se aborda la caracterización de los individuos según el estilo cognitivo dependencia-independencia de campo (DIC), identificado en la mayoría de los casos mediante la prueba de figuras enmascaradas (EFT Embedded Figures Test). La dimensión DIC puede describirse a partir de tres elementos fundamentales (Hederich y Camargo, 1998): 1) el tipo de información relevante para la realización de una tarea, 2) la forma como se codifica la información para operarla en la memoria a corto plazo y 3) la eficacia en la manipulación de la información en la memoria.

Con respecto a (1) se tiene que los sujetos independientes de campo (IC) prefieren atender a claves internas provenientes de su propio conocimiento o experiencias previas, en cambio, los sujetos dependientes (DC) atienden claves de carácter social que dirigen la entrada de información al sistema. Con respecto a (2), mientras que los IC reciben la información y la reestructuran de acuerdo con el objetivo de la tarea, los DC reciben la información y la almacenan manteniendo la estructura original. En (3), dada la forma de almacenamiento, los IC proporcionan un mejor manejo de la memoria, eliminando información irrelevante, proceso que no realizan los DC.

El (Case-based reasoning) CBR es un enfoque para la solución de problemas que se basa en la recuperación y adaptación de casos, o descripciones de episodios de problemas y sus soluciones asociadas (Allen, 1994 p.40). Desde esta perspectiva, se asume que el conocimiento humano está almacenado y puede ser capturado en forma de historias (Schank, 1990). Cuando se pregunta por la solución de un problema, se busca en la memoria las experiencias pasadas que pueden ser reaplicadas en nuevas situaciones y descritas contando

una historia. La razón para reutilizar estas historias es crear nuevas soluciones usando el conocimiento y las soluciones usadas anteriormente (Kolodner, 1993).

La investigación se plantea con un diseño cuasiexperimental, realizando los siguientes procesos:

- Desarrollo del primer prototipo del software para documentación de casos de estudio.
- Realizar una prueba piloto del funcionamiento del software.
- Aplicar las correcciones definidas para el software de acuerdo con los resultados arrojados por la prueba piloto.
- Documentación de 2 casos de estudio utilizando el software
- Aplicación de la prueba EFT para identificar el estilo cognitivo de los estudiantes de bases de datos.
- Se seleccionarán dos grupos de estudiantes del curso de bases de datos (A y B), según el agrupamiento planteado por la UDI : grupo A que realizará el estudio de los casos mediante el uso del software; grupo B que realizará el estudio de los casos de la forma tradicional.
- Aplicación de una prueba individual que permita medir la competencia de los estudiantes para diseñar una base de datos.
- Análisis de los resultados mediante un análisis de varianza de 2 x 2
- Redacción de conclusiones y recomendaciones para el rediseño del software.

Con el desarrollo del proyecto se espera obtener:

- Un software que gestione casos de estudio y sirva de soporte a los procesos académicos de la UDI, en el área de Bases de datos.
- Una Estrategia pedagógica orientada por el procesamiento de casos de estudio, ajustada a la orientación del curso de Bases de datos.
- Una caracterización del desempeño de los estudiantes dependientes-independientes de campo, en el curso de Bases de datos.

Lineamientos para la formulación y gestión de casos de estudio, específicamente en el dominio de las bases de datos, con aplicación en nuevos procesos de evaluación de las competencias de diseño de los estudiantes.

## **Bibliografía**

Bohórquez, L.; Amaya, Y. (2008) Diseño de un ambiente instruccional apoyados en TIC para el aprendizaje de fundamentos de programación de computadores. [En línea] URL: [http://www.colombiaprende.edu.co/html/mediateca/1607/articles-74614\\_archivo.pdf](http://www.colombiaprende.edu.co/html/mediateca/1607/articles-74614_archivo.pdf).

Cafolla, R. (1987). "The Relationship of Piagetian formal Operations and other cognitive factors to computer programming ability (Development)". *Dissertations Abstracts*. Vol. A47, No. 7, p. 2506.

Connolly, T. M., & Begg, C. E. (2006). A Constructivist-Based Approach to Teaching Database Analysis and Design. *Journal of Informations Systems Education* , 17(1), 43-53.

Dalbey, J., & Linn, M. C. (1985). "The Demands and Requirements of Computer Programming: A Literature Review." *Journal of Educational Computing Research*. Vol. 1, No. 3, pp. 253-74.

Fletcher, S. H. (1984). "Cognitive Abilities and Computer Programming." EDRS(ED259700).

Galton, A. (1992). "Logic as a Formal Method." *The Computer Journal*. Vol. 35, No. 5, pp. 431-440.

Gibbons, P. (1995). "A cognitive processing account of individual differences in novice Logo programmers' conceptualization and use of recursion." *Journal of Educational Computing Research*. Vol. 13, No. 3, pp. 211-226.

Gibbs, H., & Tucker, A. B. (1986). "A Model Curriculum for a Liberal Arts degree in Computer Science." *Communications of the ACM*. Vol. 29, No. 3, pp. 202-210.

Guzdial, M; Soloway, E. (2002). Teaching the Nintendo generation to program. *Communications of the ACM*. Vol 45. No. 4. April 2002.

Hudak, M. A. & Andersen, D. E. (1990) "Formal Operations and Learning Style Predict Success in Statistics and Computer Science Courses." *Teaching of Psychology*. Vol. 4, pp. 231-234.

Ignatuk, N. (1986). "An Analysis of the Effects of Computer Programming on Analytical and Mathematical skills of high school students." *Dissertation Abstracts*. Vol. A47, No. 3, p. 854.

Little, L. F. (1984). "The Influence of Structured Programming, Gender, Cognitive Development and Engagement on the Computer Programming Achievement and Logical Thinking Skills of secondary Students." *Dissertation Abstracts*. Vol. A45, No. 6, p. 1708.

Mishra, P., Zhao, Y., & Tan, S. (2000). From Concept to Software : Developing a Framework to Understanding the Process of Software Design. *Journal of Research on Computing Education* , 32 (2), 220-238.

Monfort, M., Martin, S. A., & Frederickson, W. (1990). "Information-processing differences and laterality of students from different colleges and disciplines." *Perceptual & Motor Skills*. Vol. 70, No. 1, pp. 163-172.

Myers, J., J. P. (1990). "The central role of mathematical logic in computer science." *SIGCSE Bulletin*. Vol. 22, No. 1, pp. 22-26.

Ott, C. F. P. (1989). "Predicting achievement in computer science through selected academic, cognitive and demographic variables," *Dissertation Abstracts*. Vol. A49, No. 10, p. 2988.

Sein, M. K., & Bostrom, R. P. (1989). "Individual differences and conceptual models in training novices users." *Human-Computer Interaction*. Vol. 4, pp. 197-229.

Sperschneider, V., & Antoniou, G. (1991). *Logic: A foundation for Computer Science*. Reading, MA: Addison-Wesley.

Stager-Snow, D. B. (1985). "Analytical ability, logical reasoning and attitude as predictors of success in an introductory course in computer science for noncomputer science majors." *Dissertation Abstracts*. Vol. A45, No. 8, p. 2473.

Van Gorp, M.J.; Grissom, S. (2001). An Empirical Evaluation of Using Constructive Classroom Activities to Teach Introductory Programming. *Computer Science Education* 2001, Vol. 11, No. 3, pp. 247-260

Villalobos, J; Casallas, R; Marcos, K.(2005) "El Reto de Diseñar un Primer Curso de Programación de Computadores". XIII Congreso Iberoamericano de Educación Superior en Computación, Cali, Colombia, Octubre.

White, G. (Spring, 2002). "Cognitive Characteristics for Learning C-H-." *Journal of Computer Information Systems*. Vol. 42, No. 3, pp. 51-55.

White, G. and Ploeger, F. (Spring, 2004). "Cognitive Characteristics for Learning Visual Basic." *Journal of Computer Information Systems*, Vol. 44, No. 3, pp. 5866.

White, G. and Sivitanides, M. (2005). "Cognitive Differences between Procedural and Object Oriented Programming." *Journal of Information Technology and Management*. Vol. 6, No. 4, pp. 333-350.

White, G. (2006). Visual Basic Programming Impact on Cognitive Development of College Students. *Journal of Information Systems Education*. West Lafayette: Winter 2006. Tomo 17, N° 4; pg. 421, 7 pgs.

Wright, P. H. (2004). *Introducción a la Ingeniería* (3 ed.). (R. Arriola Juárez, Trad.) México DF, México: Limusa.

Wu, C.-C. (1993). "Conceptual Models and Individual Cognitive learning Styles in Teaching Recursion to Novices." Unpublished Dissertation, University of Texas, Austin.