Mapa de Aprendizajes Significativos como Modelo de Representación de Conocimientos Previos en el proceso de Caracterización de Problemas de Aprendizaje

Elizabeth Jiménez Rey, Darío Rodriguez, Ramón García-Martínez
Programa de Maestría en Tecnología Informática Aplicada en Educación. Facultad de Informática. UNLP
Laboratorio de Sistemas Operativos y Base de Datos. Facultad de Ingeniería. UBA
Área Ingeniería del Software. Licenciatura en Sistemas. Departamento de Desarrollo Productivo y Tecnológico. UNLa
ejimenezrey@yahoo.com.ar, djhr_1977@yahoo.com.ar, rgm1960@@yahoo.com

Resumen

Se presenta un avance en la indagación de posibles causas de problemas de aprendizaje de los alumnos de dos cursos de programación básica de computadoras. Se ha ensayado en trabajos previos el uso de herramientas de explotación de información para identificar incomprensiones de los estudiantes.
En base a la experiencia docente se construye el mapa de aprendizajes significativos. Se identifican los aprendizajes considerados claves, sus relaciones y precedencias y algunas estrategias pedagógicas desplegadas con la intención educativa de lograr el aprendizaje genuino de los alumnos.
Se explorará en trabajos futuros el grado de incidencia del diseño de la estructura de aprendizajes previos por parte del docente en el logro de soluciones algorítmicas de calidad en el proceso creativo de programas por parte de los alumnos.

Palabras claves: Mapa de Aprendizajes Significativos, Núcleos de Aprendizajes Significativos, Procesos Cognitivos Involucrados, Aprendizajes Significativos, Explotación de Información basada en Sistemas Inteligentes.

1. Introducción

En investigaciones recientes, interesa explorar aquello que los profesores creen que es importante enseñar en cursos introductorios de programación, aquello que realmente enseñan y aquello en que los estudiantes encuentran las mayores dificultades (según sus profesores), qué tópicos son enseñados y cuál es el rol que los novatos encuentran más difícil de desempeñar en cursos introductorios de programación [Schulte y Bennedsen, 2006].
La Explotación de Información ha sido señalada como una manera efectiva de descubrir nuevo conocimiento desde conjuntos de datos de procesos educacionales, datos generados por sistemas de aprendizaje o experimentos de aprendizaje, así como la información descubierta puede ser usada para probar la personalización y la adaptación [Beck et al, 2007; Britos, 2008]. Entre los problemas interesantes que la explotación de información puede ayudar a resolver: determinación de cuáles son los estilos o estrategias de aprendizaje comunes [Salgueiro et al, 2006; Britos et al, 2008a], predicción del conocimiento e intereses de un usuario basado en su comportamiento previo, particionamiento de un grupo heterogéneo de usuarios en clusters homogéneos o detección de conceptos no comprendidos en procesos de aprendizaje.
Una de las técnicas más comunes de explotación de información son los árboles de decisión (TDIDT), usados para descubrir conocimiento en forma de reglas, las cuales constituyen un modelo que representa el dominio del conocimiento subyacente a los ejemplos disponibles sobre el mismo. Una red bayesiana es un gráfico acíclico direccionado en el cual cada nodo representa una variable y cada arco representa una dependencia probabilística la cual especifica la probabilidad

condicional de cada variable dada a sus padres; la variable a la cual el arco apunta es dependiente (causa – efecto) de la variable en el orígen de ésta [Felgaer *et al*, 2006; Britos *et al*, 2008 b].

Para descubrir concepciones erróneas de aprendizaje comunes de aprendices, en el trabajo de investigación que se describe [Chen y Hsieh, 2005], se empleará la regla de asociación de Explotación de Información al perfil del aprendiz para diagnosticar conceptos no comprendidos comunes a los aprendices durante los procesos de aprendizaje. Las reglas de asociación que la ocurrencia del concepto no comprendido A implica la ocurrencia del concepto no comprendido B, pueden ser descubiertas utilizando la aproximación diagnóstica de aprendizaje de la regla de asociación propuesta.

Se estudia los casos correspondientes a dos cursos de la asignatura Computación de carreras no informáticas de la Facultad de Ingeniería de la Universidad de Buenos Aires.

Se ha venido realizando trabajos experimentales que tuvieron en cuenta qué medir y por qué a través de la identificación de una lista de componentes (errores más frecuentes) de conceptos no comprendidos [Britos *et al*, 2008c] por el estudiante en el proceso creativo de soluciones algorítmicas con la computadora (programación con tipos de datos simples y estructuras de control básicas) y sobre observación y registro de una población de 88 evaluaciones individuales en papel de dos cursos de computación en la FIUBA, estableciéndose [Jiménez Rey *et al*, 2008]:

- *Taxonomía de los componentes*
  Valoración metodología de desarrollo, funcionalidad del programa, calidad del diseño (Figura 1).
- *Descripción de los componentes*
  Qué medir
- *Fundamentación de la elección*
  Por qué medir
- *Determinación del peso*
  Cómo valorar

Para descubrir conceptos mal comprendidos de programación de los estudiantes, se focalizó la investigación en el uso de herramientas de bases de datos de sistemas inteligentes: inducción de reglas por algoritmos TDIDT y redes bayesianas. Estas reglas se utilizaron en un proceso de descubrimiento del conocimiento en tres pasos:

- Construcción de una base de datos sobre una caracterización estándar de cada estudiante y su estilo y conceptos mal aprendidos de programación.
- Descubrimiento de reglas (mediante algoritmos TDIDT) las cuales establecieron una relación entre conceptos no comprendidos de programación y sus posibles causas. Las reglas obtenidas se aplicaron para revisar la estructura propuesta para el curso y las hipótesis del profesor. Los resultados experimentales indican que aplicando este paso, el profesor puede descubrir posibles causas de los conceptos mal aprendidos por los estudiantes.
- Descubrimiento del peso que cada causa tiene sobre cada concepto mal aprendido (mediante redes bayesianas). Esto permitió establecer un rango de importancia de las causas de conceptos no comprendidos para proponer estrategias de enseñanza remediales.

Metodología de Desarrollo	**Funcionalidad del Programa**	**Calidad del Diseño**
¿Aplica método de refinamientos sucesivos? ¿Describe el significado de variables? ¿Tiene estilo de escritura? ¿Usa enunciados de documentación interna?	¿Comprende el objetivo del problema? ¿Descubre la naturaleza del problema? ¿Descubre el algoritmo? ¿Generaliza la solución? ¿Logra funcionamiento del programa? ¿Realiza prueba de escritorio? ¿Comete errores de sintaxis? ¿Usa delimitadores begin/end correctamente? ¿Asigna correctamente?	¿Usa conectores lógicos en forma correcta? ¿Controla condición fin de ciclo repetitivo? ¿Obtiene una solución lógica? ¿Desarrolla ciclo infinito? ¿Inicializa variables de condición antes del While? ¿Optimiza uso de recursos? ¿Evita repetición de código? ¿Usa indistintamente While y Repeat? ¿Usa símbolo de asignación en condición?

Figura 1. Taxonomía de los Componentes.

El interés principal radica en la exploración de la posibilidad de construir un Proceso que permita al profesor, por una parte, revisar la precedencia de los aprendizajes significativos de los estudiantes considerados en el diseño del currículo y, por otra, reflexionar sobre las estrategias de enseñanza de su práctica docente para favorecer en los estudiantes el desarrollo de la cognición en su más amplio sentido: la comprensión y la capacidad para pensar y conocer cada vez mejor [Litwin, 2008].
En particular, se estudia un caso específico del área tecnológica, la asignatura Computación desarrollada en la Facultad de Ingeniería de la Universidad de Buenos Aires, con el propósito de mejorar el proceso de enseñanza y aprendizaje de la misma.
Y en general, a partir del caso en cuestión y mediante la aplicación de técnicas de Minería de Datos basada en Sistemas Inteligentes como herramienta de diagnóstico, se busca auxiliar a los docentes de cualquier disciplina educativa, en la aplicación de un Proceso de Descubrimiento del Conocimiento para detectar y predecir las dificultades de aprendizaje de los alumnos.

2. Dominio del Problema

2.1. Fundamentación

Al igual que Bruner, Ausubel piensa que es posible enseñar a los alumnos los conceptos estructurales de cada disciplina de manera que funcione para ellos como un sistema de procesamiento de la información (estructura cognitiva). Es decir, un mapa intelectual que pueden usar para el análisis de dominios específicos y para resolver problemas dentro de esos dominios [Joyce y Weil, 2002].
Ausubel describe a la mente como un sistema de procesamiento y almacenamiento de la información comparable a la estructura conceptual de una disciplina académica. A semejanza de las disciplinas, la mente es un conjunto jerárquicamente organizado de ideas que proporcionan el anclaje para la información y las ideas, así como un lugar donde almacenarlas.
Cuando este sistema de procesamiento adquiere nueva información y nuevas ideas, se reorganiza a sí mismo con el propósito de incorporarlas. El sistema se halla entonces en un estado de cambio continuo. Ausubel postula que las nuevas ideas pueden aprenderse y retenerse provechosamente siempre y cuando se relacionen con los conceptos o proposiciones ya existentes que proporcionan los anclajes intelectuales.

2.2. Implicación

Las ideas de Ausubel acerca de los contenidos y la estructura cognitiva tienen consecuencias directas e importantes tanto en la organización del currículo como en los métodos didácticos. El autor utiliza dos principios, la *diferenciación progresiva* y la *reconciliación integradora,* como guías para organizar el contenido de las asignaturas.
La *diferenciación progresiva* significa que primero se presentan las ideas más generales de la disciplina y luego se sigue con un incremento gradual de los detalles y la especificidad. La *reconciliación integradora* significa que las nuevas ideas deben relacionarse concientemente con el contenido que ya se aprendió.
La secuencia curricular debe organizarse de manera que cada aprendizaje sucesivo esté cuidadosamente relacionado con las exposiciones previas.

3. Solución Metodológica Propuesta

La solución propuesta es una segmentación de los procesos intuitivos del docente [Jiménez Rey *et al,* 2009] (de construcción artesanal y fuertemente ligados a su experiencia) que se estructura en un proceso de varias etapas:

Etapa 1: Construcción del Mapa de Aprendizajes Significativos, que se articulará en concepto-atributo-valor [García Martínez y Britos, 2004] a determinar su existencia en los instrumentos de evaluación.

Etapa 2: Explotación de Información sobre la base de evaluación.

Etapa 3: Ratificación o Rectificación del Mapa de Aprendizajes Significativos.

Para construir el Mapa de Aprendizajes Significativos en la primera etapa del proceso, se aplica una metodología de desarrollo en tres pasos:

Paso1. Identificación de conceptos relevantes.

Paso2. Establecimiento de precedencia de conceptos.

Paso3. Formulación de hipótesis sobre relaciones causales de conocimiento.

En el Paso 1, el docente decide qué quiere o pretende enseñar, qué desea que aprendan los alumnos. Se diseña la estructura del contenido de la asignatura y se identifican los conceptos relevantes a ser enseñados. Se privilegian temas generadores caracterizados por su centralidad, accesibilidad y riqueza en el campo disciplinar para introducirlos en la trama de la asignatura.
En el Paso 2, el docente establece la relación entre los conceptos para organizar el contenido disciplinar. Se determinan los vínculos entre conceptos para definir cuáles deben ser aprendidos previamente por el alumno antes de intentar enseñar un concepto nuevo (precedencia de aprendizajes). Se ordena el material de aprendizaje por secuencias para presentarlo de manera que permita proveer de anclas conceptuales.
En el Paso 3, el docente realiza un análisis exhaustivo sobre el diseño de las precedencias de aprendizajes para formular hipótesis que validen las relaciones entre los conceptos propios del campo disciplinar y la manera de presentar el conocimiento a los alumnos para promover un aprendizaje genuino. Se reflexiona sobre el conocimiento nuevo a introducir en la estructura cognitiva del alumno, se considera cuidadosamente las conexiones con el conocimiento previo, se concilian las diferencias o discrepancias y se advierten las similitudes con la información ya existente.

4. Caso de Estudio

4.1. Descripción

Se identifican los aprendizajes claves que los alumnos de los cursos de Computación deben adquirir para una comprensión genuina de la programación de computadoras.
Los aprendizajes claves constituirán los nodos del Mapa de Aprendizajes Significativos y se denominarán Núcleos de Aprendizajes Previos (NAP).
Se establecen las conexiones entre los aprendizajes claves proporcionadas por la necesidad de los aprendizajes previos para que los alumnos puedan retener y apropiarse de los aprendizajes nuevos.
Las conexiones entre los aprendizajes claves constituirán los enlaces del Mapa de Aprendizaje Significativos y se denominarán Procesos Cognitivos Involucrados (PCI).
Desde el punto de vista didáctico:

- Los NAP son momentos de enseñanza y aprendizaje de los contenidos de la asignatura. Representan el *programa* para la enseñanza y el aprendizaje de la asignatura. Conforman la *planificación*, la decisión docente sobre **qué** enseñar.
- Los PCI son caminos de enseñanza y aprendizaje de los contenidos que permiten avanzar de un NAP a otro NAP para incrementar el aprendizaje con conocimiento nuevo y, a veces, también retroceder a otro/s NAP para reforzar el aprendizaje o recuperar el conocimiento ya adquirido. Representan la *estrategia* para la enseñanza y el aprendizaje de la asignatura. Conforman la *implementación*, la decisión docente sobre **cómo** enseñar.

Los NAP son cruzados por los PCI.

4.2. Desarrollo del Caso de Estudio

4.2.1. NAP

COMPUTACIÓN y COMPUTADORA.

Se intenta que los alumnos incorporen a su estructura cognitiva los conceptos de

Computación como una disciplina que busca establecer las bases científicas para resolver problemas con la computadora y de computadora como máquina ejecutora de algoritmos representados en forma de programa.

4.2.2 PCI

Se propone a los alumnos trabajar en grupo frente a la computadora durante 15 minutos. Deben utilizarla en clase de alguna de las maneras en que la usan en su vida cotidiana y redactar un informe que describa en forma concisa qué hicieron (la situación elegida) y cómo lo hicieron (el programa involucrado), identificando las entradas (datos) y las salidas (resultado).
Se analizan y comparan los informes de los distintos grupos mediante una puesta en común. Se aprovecha la experiencia para entender la versatilidad de la computadora (solución a diferentes situaciones problemáticas de la vida real), para incorporar conceptos de hardware (parte física), de software (parte lógica), automatización, comunicación, almacenamiento y procesamiento de información. Se clarifica el objetivo de la asignatura (resolución de problemas con la computadora) contextualizándolo dentro del alcance de las ciencias de la computación. *Relaciones concretas-abstractas, explícitas-implícitas. Distinción de los aspectos prácticos del nuevo conocimiento. Uso del objeto concreto para una relación completamente funcional del conocimiento.* Los significados de Computación y computadora que traen los alumnos se resignifican desde el campo disciplinar.
Surge la necesidad de comprender en qué consiste y cómo se resuelve un problema y en qué consiste una computadora.

4.2.3. NAP

RESOLUCIÓN GENERAL de PROBLEMAS
Se intenta que los alumnos incorporen a su estructura cognitiva el proceso científico de resolución de problemas, en este caso, el Modelo de Polya de Cuatro Fases [Nickerson *et al*, 1987], como metodología evolutiva para la resolución de un problema cualquiera.

4.2.4. PCI

Se propone a los alumnos la resolución grupal de un problema matemático a través de un enunciado. Deben pensar y discutir en grupo la solución del problema e informar el resultado al cabo de 5 minutos. *Se promueve el trabajo activo y cooperativo dentro del aula para provocar el aprendizaje reflexivo.* Se invita a todos los estudiantes a pensar sobre la forma en que pensaron para resolver el problema planteado. *Pensar sobre el pensamiento (metacognición).* Se interactúa con los alumnos a través de la formulación de preguntas para concluir que siguieron inconcientemente un proceso de cuatro fases para arribar a la solución. *Se proporciona a los alumnos oportunidades para pensar, para que se interroguen sobre los procesos por los que aprenden, haciéndolos concientes de los mismos.* Se representa el Modelo de Polya de las Cuatro Fases mediante un mapa conceptual [Jiménez Rey y Perichinsky, 2006]. Se redesignan las fases del modelo en el contexto de la resolución de problemas con la computadora (construcción de programas). Surge la necesidad de aplicar el Modelo de Polya para crear un programa.

4.2.5. NAP

ALGORITMO y PROGRAMA.

Se intenta que los alumnos incorporen a su estructura cognitiva los conceptos de algoritmo como procedimiento de resolución de un problema o de realización de una tarea y de programa como expresión de un algoritmo en un lenguaje de programación para que pueda ser entendido y ejecutado por la computadora.

4.2.6. PCI

Se propone a los alumnos el análisis grupal de una situación problemática de la vida real (creación de una receta de cocina por un chef para su publicación). Se presenta el texto de la

receta como la solución del problema y como un caso de aplicación del Modelo de Polya de Cuatro Fases. Los alumnos leen el enunciado del problema y la receta solución creada por el chef y deliberan para responder un cuestionario durante 10 minutos.
Se diseña el material didáctico para que la actividad genere el arribo a conclusiones que proporcionen el anclaje necesario para introducir los conceptos de algoritmo y programa. *La receta de cocina como organizador previo del concepto de algoritmo.*
Se verifican las respuestas al cuestionario de los distintos grupos mediante una puesta en común analizando la solución dada al problema por el chef y se estructura el texto de la receta de cocina en objetivo, ingredientes, elaboración. *Procesamiento de abajo-arriba.*
Se introduce el concepto de programa de computación como un caso particular de algoritmo. Surge la necesidad de conocer cómo se escribe el texto de un programa en lenguaje Pascal para solucionar un problema algorítmico.

4.2.7 NAP

ESTRUCTURA de un PROGRAMA Pascal.
Se intenta que los alumnos incorporen a su estructura cognitiva la organización de un programa Pascal como modelo de programa tipo, de manera a contar con un instrumento que les guíe en el proceso de modelización de una situación problemática de la vida real para su resolución por medio de la computadora.

4.2.8. PCI

Se parte de la estructura de una receta de cocina como organizador previo, se establece el paralelismo entre receta y programa como casos particulares de algoritmo, resaltando las similitudes y las diferencias y se redesignan los componentes en el contexto de la construcción de programas. *Diferenciación progresiva y reconciliación integradora*.
Se estructura el texto de un programa Pascal en dos secciones (declarativa y algorítmica). Cada sección se compone a su vez de subsecciones. La sección declarativa se descompone en una definición de objetivo y una definición de recursos. La sección algorítmica se descompone en subproblemas a resolver en distintos niveles de refinamiento hasta un último nivel de descomposición. Se instala un modelo de programa tipo para la implementación de soluciones a problemas con la computadora. *Procesamiento de arriba-abajo.*
Surge la necesidad de conocer el lenguaje que las computadoras pueden entender para comprender cómo se expresan en lenguaje Pascal el objetivo del programa, los recursos a clasificar y las sentencias (del último nivel de refinamiento) a secuenciar.

4.2.9. NAP

CONSTRUCCIÓN de una COMPUTADORA.

Se intenta que los alumnos incorporen a su estructura cognitiva el conocimiento sobre la construcción de la computadora a partir de dispositivos de conmutación de estado binario capaces de representar un bit (uno o cero) de información, los cuales pueden ser agrupados en patrones de bits (byte) para representar datos (recursos) e instrucciones (sentencias) como lenguaje de máquina que la computadora puede entender.

4.2.10. PCI

Se propone a los alumnos la interpretación de un patrón compuesto por dígitos decimales (conocimiento previo) y se relaciona con la decodificación de un patrón binario (conocimiento nuevo) para incorporar el concepto de byte (combinación de unos y ceros) como representación de la información.
Se realiza la analogía entre el concepto de byte almacenado y las posiciones posibles (“sí/no”) de las llaves comunes de pared para encender la luz para el logro de la comprensión del concepto de memoria (almacenamiento de la información). *Las metáforas otorgan una fuerza comprensiva diferente y son favorecedoras de nuevas conceptualizaciones.*
Surge la necesidad de conocer cómo la computadora ejecuta algoritmos escritos en un lenguaje de programación.

4.2.11. NAP

COMPILADOR
Se intenta que los alumnos incorporen a su estructura cognitiva el conocimiento de los compiladores como traductores de un programa en lenguaje de alto nivel a un lenguaje de máquina.

4.2.12. PCI

Para introducir el concepto de compilador se establece la analogía entre los traductores de programas fuente a programa objeto (conocimiento nuevo) y los libros traducidos de un idioma a otro por seres humanos (conocimiento existente). Surge la necesidad de conocer la organización básica de una computadora y las acciones elementales que puede ejecutar para resolver problemas que tienen soluciones algorítmicas.

4.2.13. NAP

CAPACIDADES y ORGANIZACIÓN de una COMPUTADORA.
Se intenta que los alumnos incorporen a su estructura cognitiva el conocimiento sobre las capacidades que debe tener una máquina para poder ejecutar algoritmos y las asocien con los dispositivos específicos (organización básica) de la computadora (hardware) y con los enunciados ejecutables (de entrada, salida y asignación) del lenguaje de programación Pascal (software) para solucionar un problema (programa) con la computadora.

4.2.14. PCI

Se propone a los alumnos el análisis en torno a los organizadores previos utilizados con anterioridad (resolución de problema matemático, creación de receta de cocina) para reflexionar sobre la necesidad de contar con un soporte físico y un soporte lógico para alcanzar la solución de una situación problemática. Se resalta que el proceso de solución de problemas o realización de tareas (manual, biológico, industrial, etc.) necesita de recursos (entrada) que son transformados por un procedimiento (proceso) para obtener un resultado o producto (salida). Se establece la analogía con el esquema funcional de una computadora.

4.2.15. NAP

CONSTRUCCIÓN de un PROGRAMA Pascal.
Se intenta que los alumnos incorporen a su estructura cognitiva el conocimiento sobre el proceso creativo de construcción de un programa Pascal como un proceso evolutivo de Cuatro Fases: análisis, diseño, codificación y evaluación. Las fases deben completarse mediante sucesivos avances y retrocesos hasta la obtención de una solución.

4.2.16. PCI

Se plantea la resolución de un problema con la computadora mediante la escritura de un programa Pascal en la primera clase. En el caso en estudio, la creación del programa requiere la utilización de tipos de datos simples y sentencias secuenciales. Como método estratégico de enseñanza y aprendizaje se usa un Mapa Conceptual para la Creación de Programas [Jiménez Rey, 2005] que se muestra en la Figura 2. Se realiza con los alumnos el tránsito cognitivo y evolutivo desde el enunciado-problema hasta el hallazgo del programa-solución. En cada una de las siguientes clases, se plantea la resolución de un nuevo problema con la computadora mediante la escritura de un programa Pascal. En cada uno de los casos en estudio, la creación de cada programa requiere la utilización de una nueva herramienta de programación (sentencias selectivas, repetitivas, de transferencia/retorno, tipos de datos compuestos) que se van integrando a las herramientas ya conocidas para solucionar problemas de mayor complejidad como se puede ver en la Figura 3.
En cada una de las siguientes clases, el mapa conceptual previo se recupera y se reestructura con la inclusión de cada nueva herramienta de

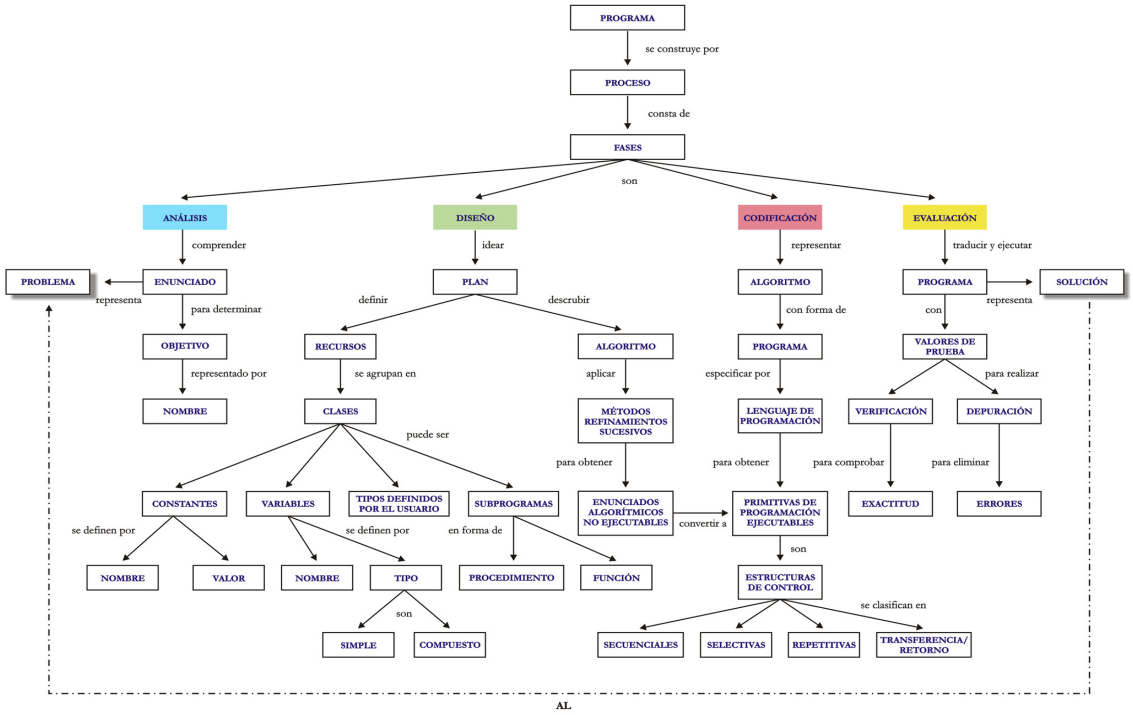

Fig. 2. Mapa Conceptual para la Creación de Programas.

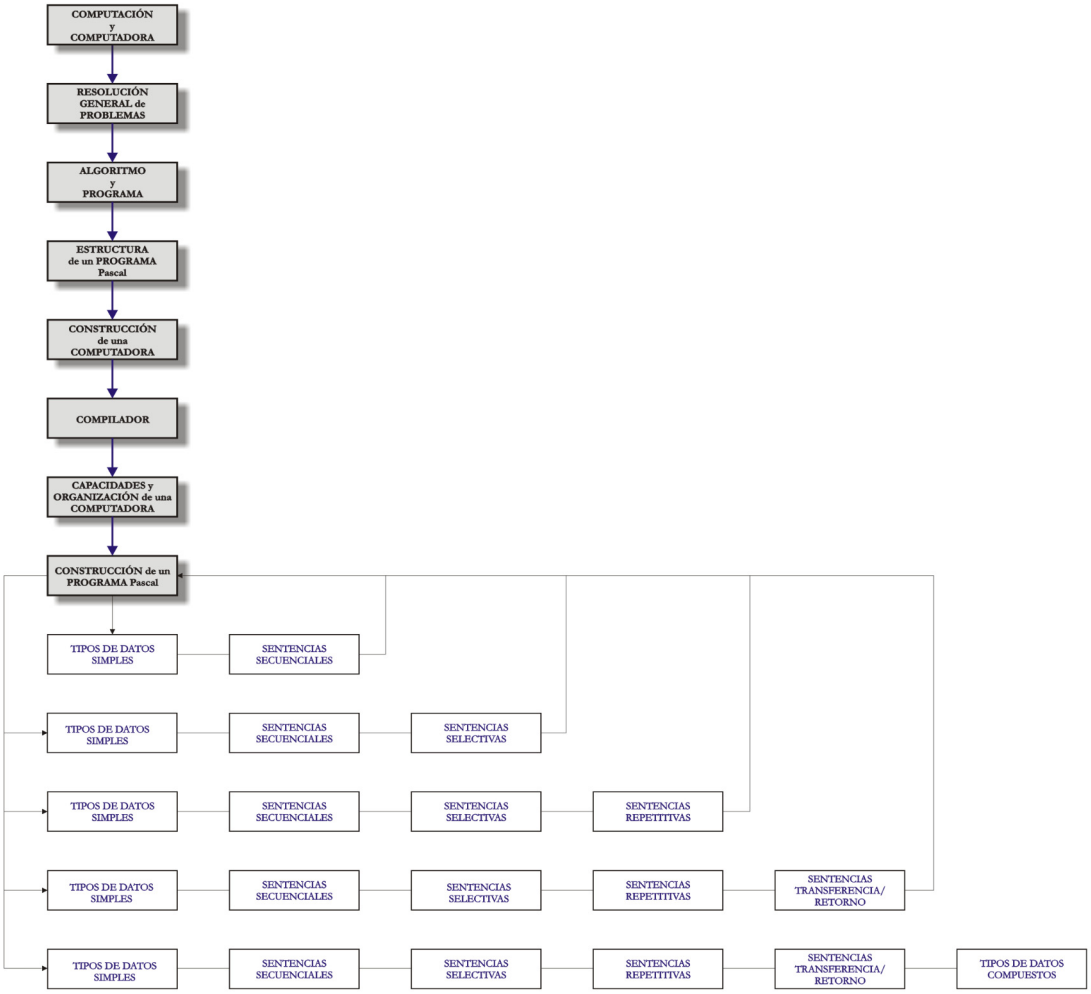

Fig. 3. Mapa de Aprendizajes Significativos.

programación, que lo amplía y complejiza. *Almacenamiento y recuperación memoria a largo plazo*.

5. Conclusiones y Futuras Líneas de Investigación

Se presentó un diseño preliminar de los conceptos considerados claves en la enseñanza de un curso de introducción a la programación básica de computadoras a estudiantes de ingenierías no informáticas para el logro de un aprendizaje significativo.

En próximos avances los modelos de representación de precedencia de aprendizajes significativos serán revisados para la ratificación o rectificación del Mapa de Aprendizajes Significativos.

Se explorará en trabajos futuros el grado de incidencia del diseño de la estructura de aprendizajes previos por parte del docente en el logro de soluciones algorítmicas de calidad en el proceso creativo de programas por parte de los alumnos.

Se prevé la identificación y codificación tabular de conceptos a aprender, la sistematización de los resultados de las evaluaciones y la aplicación de procesos de Explotación de Información sobre una población integrada por 300 estudiantes de los cursos de Computación.

6. Bibliografía

Beck, J. , Calders, T., Pechenizkiy, M., Viola, S. (2007). *Workshop on Educational Data Mining*. ICALT'05: 933-934.

Britos, P. (2008). *Procesos de Explotación de Información Basados en Sistemas Inteligentes*. Tesis de Doctorado en Ciencias Informáticas. Facultad de Informática. UNLP.

Britos, P., Cataldi, Z., Sierra, E., García-Martínez, R. (2008a). *Pedagogical Protocols Selection Automatic Assistance*. LNAI, 5027: 331-336.

Britos, P., Felgaer, P., Garcia-Martinez, R. (2008b). *Bayesian Networks Optimization Based on Induction Learning Techniques*. IFIP Series, 276: 439-443.

Britos, P., Jiménez Rey, E., Rodríguez, D., García-Martínez, R. (2008c). *Work in Progress: Programming Misunderstandings Discovering Process Based On Intelligent Data Mining Tools*. Proceedings 38th ASEE/IEEE Frontiers in Education Conference. Session F4H: Assessing and Understanding Student Learning. ISBN 978-1-4244-1970-8.

Chen , C. , Hsieh, Y.(2005). *Mining Learner Profile Utilizing Association Rule for Common Learning Misconception Diagnosis*. ICALT'05: 588-592.

Felgaer, P., Britos, P. and García-Martínez, R. (2006). *Prediction inHealth Domain Using Bayesian Network Optimization Based on Induction Learning Techniques*. Internacional Journal of Modern Physics C 17(3): 447-455.

García Martínez, R., Britos, P. (2004). Ingeniería de Sistemas Expertos. Editorial Nueva Librería. ISBN 987-1104-15-4.

Jiménez Rey, E. (2005). *Un Enfoque Procedimental para la Enseñanza de Computación en Carreras de Ingeniería*. Proceedings I JEITICs 2005: 35-39.

Jiménez Rey, E., Perichinsky, G. (2006). *El Mapa Conceptual como Representación del Modelo de Polya para la Creación de Programas*. Proceedings VIII WICC 2006: 581-585. ISBN 978-950-9474-35-2. ISSN 950-9474-35-5.

Jiménez Rey, E., Rodríguez, D., Britos, P. y García-Martínez, R. (2009*). Caracterización de Problemas de Aprendizaje basada en Explotación de Información*. Proceedings del XI WICC 2009. Artículo 2390. ISBN 978-950-8631015.

Jiménez Rey, E., Rodríguez, D., Britos, P., García-Martínez, R. (2008). *Identificación de Problemas de Aprendizaje de Programación con Explotación de Información*. Proceedings del XIV CACIC 2008. Artículo 1881. ISBN 978-987-24611-0-2.

Joyce, B., Weil, M. (2002). Modelos de Enseñanza. Gedisa. ISBN 84-7432-780-6.

Litwin, E. (2008). *El Oficio de Enseñar, Condiciones y Contextos*. Paidós. ISBN 978-950-12-1513-7.

Nickerson, R., Perkins, D., Smith, E. (1987). *Enseñar a pensar. Aspectos de la aptitud intelectual*. Paidós. ISBN 84-7509-452-X.

Salgueiro, F., Cataldi, Z., Britos, P., Sierra, E. y García Martínez, R. (2006). *Selecting Pedagogical Protocols using SOM*. Research in Computing Science Journal, 21: 205-214.

Schulte, C., Bennedsen, J. (2006). *What do teachers teach inintroductory programming?*. ICERW´06: 17-28.