Ministerio de Educación Superior
Universidad Central "Marta Abreu" de Las Villas

TESIS EN OPCIÓN AL TÍTULO DE MASTER EN COMPUTACIÓN APLICADA

Título: Organización del conocimiento de la asignatura Programación II para Ingeniería Informática basada en mapas conceptuales.

Autor: Lic. Yolanda Soler Pellicer.

Tutor: Dr. C. Mateo Gerónimo Lezcano Brito.

2007
"Año 49 de la Revolución"

Índice **Pág.**

Resumen

Se realiza una evaluación histórico-tendencial y psicopedagógica de la Tecnología Educativa y los ambientes de aprendizaje, como herramientas que permiten elevar la excelencia del proceso docente, destacando la importancia de los mapas conceptuales, su utilización en la enseñanza de la programación y las ventajas que brindan para organizar los contenidos de una determinada materia.

Los resultados de este trabajo incluyen un análisis crítico del sistema de conocimientos de la disciplina Técnicas de Programación de Computadoras de la carrera Ingeniería Informática, del nivel de precedencia de los contenidos y la necesidad de vincular en la asignatura Programación II los temas de diseño y análisis para obtener algoritmos eficientes. Se crea un mapa conceptual que muestra la relación entre los Tipos Abstractos, las estructuras de datos, los algoritmos que implementan sus operaciones, características, conceptos y aplicaciones, algunas resultantes de los Proyectos de Curso de la asignatura.

Introducción

La tecnología educativa hace uso de herramientas que ayudan a las instituciones docentes en la formación y preparación de los estudiantes, así como a mejorar los procesos de evaluación, aumentando también el rango de los estudios, al ser posible la educación remota, sin menoscabo de la calidad de la instrucción recibida. (Bravo, 2000)

La computadora como instrumento pedagógico ha demostrado su eficacia cuando se aplica para apoyar la enseñanza y el aprendizaje de procesos complejos, abstractos o difíciles de visualizar en su medio real.

Es, entonces, válido el planteamiento de Ortiz (2001), al considerar el aprendizaje, como un fenómeno subjetivo que posee un carácter complejo y susceptible de ser analizado desde diferentes puntos de vista o enfoques, ya que el hombre, por su propia flexibilidad, es capaz de incorporar nuevos conocimientos de múltiples y variadas formas.

En el proceso de enseñanza-aprendizaje de la asignatura Programación II de la carrera de Ingeniería Informática, se han enfrentado problemas con la asimilación, análisis y creación de habilidades en el alumno que les permitan el diseño adecuado, eficiente y óptimo, de las estructuras para representar la información y las operaciones básicas que con ellas se realizan. Existen deficiencias en el uso de la representación secuencial, ya estudiada en la asignatura que le precede, y su aplicación en el diseño de estructuras complejas. No se evidencia una vinculación adecuada entre los conceptos estudiados, lo que influye negativamente en la solución óptima de los problemas.

En la investigación realizada con el objetivo de encontrar herramientas que apoyaran el proceso de vinculación e integración de los contenidos de la asignatura Programación II, se encontraron sistemas como el propuesto por Almeida (2003), un sitio web sobre estructuras de datos con animación de sus operaciones básicas; en el servidor de la aplicación CmapTools, diseñada por

Cañas (2004) para la creación de mapas conceptuales, no se encontró ninguno que representara los contenidos de la asignatura; el mapa conceptual hipermedial Kellu, creado por Uviña y colaboradores (2005), muestra las definiciones básicas de trabajo con las estructuras, fundamentalmente, los nexos entre los conceptos, sin apoyarlos con informaciones y recursos; el sistema Xtango, desarrollado por Axoft (2005), presenta las operaciones con las estructuras y la representación gráfica de las acciones de forma estática.

Teniendo en cuenta el diseño curricular de la asignatura, se comienzan a impartir los contenidos relacionados con Tipos Abstractos de Datos y las listas enlazadas como estructura de datos, se realiza una analogía con una estructura lineal ya estudiada, los arreglos, y se muestra la diferencia entre la representación de memoria secuencial y enlazada. Desde este momento se evidencian grandes insuficiencias en la definición de arreglos, por lo que se necesita profundizar en el manejo de la representación secuencial dada la incidencia que tiene en el logro de los objetivos de la asignatura Programación II, afectándose, además, la selección eficiente de estructuras para representar la información.

La situación expuesta no sólo debe analizarse en el momento en que se aprende sino también cuando se desarrolla una aplicación que utiliza algoritmos que por su complejidad pueden dar por resultado que el producto obtenido no sea lo suficientemente eficiente.

De ahí, la importancia que el contenido desarrollado en la asignatura Programación II tiene desde diferentes puntos de vista; teórico y educacional, metodológico, técnico, económico, desde el punto de vista práctico y de aporte a la producción y la sociedad. Basándose en esta realidad se debe considerar si:

- ¿Las técnicas de Enseñanza Asistida por Computadoras (EAC) tendrán un impacto positivo en el Proceso Docente Educativo de la asignatura Programación II?

- ¿Existen teorías educativas que aporten el fundamento pedagógico a sistemas de ayuda al diseño de algoritmos y de las Estructuras de Datos usadas para representar la información?
- ¿Se han combinado elementos de las tendencias pedagógicas contemporáneas y las herramientas informáticas en la solución de este problema?
- ¿Es necesario analizar el currículo de la carrera de Informática y la disciplina de Técnicas de Programación para realizar una propuesta metodológica que organice el sistema de conocimientos de la asignatura Programación II?
- ¿Es significativa la organización del sistema de conocimientos de la asignatura en forma de Mapa Conceptual?

Luego de analizadas estas interrogantes se evidencia la necesidad de organizar los contenidos de la asignatura Programación II, aprovechando las ventajas que ofrecen las herramientas informáticas y los ambientes de aprendizaje constructivistas, especialmente la enseñanza significativa y los mapas conceptuales.

Problema:

¿Cómo organizar los contenidos de la asignatura Programación II a través de una herramienta didáctica apoyada en mapas conceptuales?

Estas aplicaciones muestran el interés por buscar vías que apoyen el proceso de enseñanza aprendizaje de los temas de estructuras de datos, es por ello que en el presente trabajo, el Proceso Docente Educativo de la asignatura Programación II de la carrera de Ingeniería Informática constituye el **objeto de la investigación** y el **campo de acción** es el sistema de conocimientos de la asignatura.

Los resultados obtenidos permiten cambiar viejos paradigmas de enseñanza tradicional y utilizar un medio que contribuye a organizar en forma de sistema los

contenidos de la asignatura, poniendo las técnicas de diseño y análisis de algoritmos al alcance de los estudiantes. En base a la experiencia acumulada y a las facilidades que brindan actualmente la Tecnología de la Información y las Comunicaciones y la Informática Educativa, se proponen los siguientes objetivos:

Objetivo General:

Diseñar un mapa conceptual que organice el sistema de conocimientos de la asignatura Programación II, mostrando las relaciones entre los distintos Tipos Abstractos, las estructuras de datos, los algoritmos que implementan sus operaciones, sus características, conceptos y aplicaciones.

Objetivos Específicos:

1. Definir el ambiente de aprendizaje a utilizar, valorando la influencia de la Enseñanza Asistida por Computadoras, y en especial de los mapas conceptuales, en la enseñanza de la Programación.
2. Analizar los contenidos de la disciplina Técnicas de Programación de Computadoras, la relación y precedencia de sus asignaturas determinando los aspectos que afectan el proceso docente como sistema.
3. Crear una herramienta que, partiendo de los conceptos conocidos y a través de las relaciones con nuevos conceptos, logre un nivel de integración de los contenidos relacionados con los Tipos Abstractos, las Estructuras y los Tipos de Datos, incorporando al sistema elementos multimediales, simulaciones, imágenes y facilidades para el trabajo colaborativo en red.

Estructura y contenido del trabajo

El cuerpo de la Tesis se compone de tres capítulos, cuyos contenidos se describen a continuación.

Capitulo I. “La Tecnología Educativa y los ambientes de aprendizaje. Herramientas para elevar la excelencia del Proceso Docente.”

En él se realiza una caracterización histórico-tendencial y psicopedagógica de la Tecnología Educativa, y de los ambientes de aprendizaje centrados en el alumno. Se hace énfasis en el uso de los Mapas Conceptuales en la enseñanza y

especialmente en el estudio de la programación. Se plantean las ventajas del CmapTools como herramienta de apoyo a la enseñanza significativa, las que incidieron en su elección para la creación del mapa conceptual de la asignatura Programación II.

Capitulo II. "Análisis de los contenidos de la disciplina de Técnicas de Programación para Ingeniería Informática."
En este capítulo se profundiza en el diseño de la carrera del Ingeniero Informático en el contexto internacional y nacional, especialmente en la disciplina Técnicas de Programación de Computadoras y la asignatura Programación II.
Se describe el necesario perfeccionamiento de los Planes de Estudio provocado por el desarrollo acelerado de la tecnología y por el surgimiento de nuevos medios y ambientes de apoyo al autoaprendizaje y se destacan los aspectos que aún son susceptibles de mejorar teniendo en cuenta las orientaciones para la elaboración del Plan de Estudios D en la carrera.

Capitulo III. "Propuesta metodológica para la organización del sistema de conocimientos en la asignatura Programación II de la carrera de Informática"
Se describe la propuesta de organización del sistema de conocimientos de la asignatura Programación II, a través de un mapa conceptual, que incluye opciones para el trabajo colaborativo, permite compartir y adicionar nuevos conocimientos y muestra la relación entre los diferentes conceptos que se manejan al realizar el diseño de las operaciones básicas de las estructuras de datos. Se detallan los mapas principales y los recursos que lo forman. Se realizan recomendaciones a profesores y estudiantes para el uso del mapa conceptual en el proceso de enseñanza-aprendizaje de la asignatura.

1 Capítulo I. La Tecnología Educativa y los ambientes de aprendizaje. Herramientas para elevar la excelencia del Proceso Docente.

1.1 Introducción

Desde los albores de la historia humana, aprender ha sido una característica propia de cada ser humano, que ha contribuido a fundamentar las bases del desarrollo humano, ya que cuando se aprende, se adquiere el conocimiento por medio del estudio, el ejercicio o la experiencia. Las exigencias de la vida moderna, dinámica, competitiva y llena de información, han dirigido a la sociedad en la búsqueda de nuevos modelos, técnicas y sistemas que permitan adquirir esos conocimientos de una manera eficaz y eficiente.

La educación como concepto no puede ser expresada por una terminología global. Los sistemas de enseñanza parten de varios métodos que pretenden formalizar una metodología adecuada a las necesidades de nuevos estudiantes, algunas de ellas definen a la educación como un proceso de construcción y transformación del ambiente en el que se desarrolla el educando. (Bravo, 2000)

En la actualidad el proceso de enseñanza aprendizaje se caracteriza por una mayor heterogeneidad y diversidad en el alumnado, una estrecha relación y complementación entre tecnología y enseñanza y un desarrollo vertiginoso de las Tecnologías de la Información y las Comunicaciones, de ahí la necesidad de buscar nuevos modelos que ayuden a proyectar la enseñanza bajo esta nueva perspectiva.
El desarrollo de medios tecnológicos está logrando no sólo cambiar los sistemas de relación ser humano-medio, sino también, instalarse como componente cultural, por lo que su utilización en la enseñanza es imprescindible.

En el presente capítulo se hace una caracterización de la Tecnología Educativa y los ambientes de aprendizaje, desde el punto de vista histórico y psicopedagógico, particularizando en el uso de los mapas conceptuales en la enseñaza de la Programación, como fundamentos y antecedentes del presente trabajo. Se

exponen las facilidades del CmapTools como herramienta para la creación de mapas conceptuales.

1.2 La tecnología educativa en el proceso de enseñanza-aprendizaje.

Las tecnologías de la información obligan a modificar la organización de la educación, porque crean entornos educativos que amplían considerablemente las posibilidades del sistema, no sólo de tipo organizativo, sino también de transmisión de conocimientos y desarrollo de destrezas, habilidades y actitudes. La clave está en transformar la información en conocimiento y, éste, en educación y aprendizaje significativo. (Ruiz, 1996)

Cuando el alumno puede asociar el contenido que recibe con el conocimiento acumulado y la experiencia anterior, encuentra significado a lo que aprende, la motivación puede estar dada, además, por el uso de medios que lo asistan en el proceso de aprendizaje.

En el análisis realizado por Lezcano (1998), se constata que el sistema educacional cubano es una institución que imparte una educación formal que ha comprendido la necesidad de incorporar a sus metodologías el uso de herramientas que faciliten el aprendizaje, y siendo más ambiciosos, el autoprendizaje. Coincidiendo con este autor, puede añadirse que en el proceso de enseñanza-aprendizaje de las técnicas de programación estas herramientas constituyen, a la vez, objeto de estudio e instrumento pedagógico.

1.2.1 Caracterización histórico-tendencial de las aplicaciones educativas.

En época tan temprana como la década del 50 se desarrollaron sistemas mecánicos y electromecánicos que permitían la presentación de programas lineales, basados en el principio de respuesta activa. (Chambers, 1983)
La programación lineal tuvo su principal limitación en el hecho de no permitir la ramificación del proceso de enseñanza de acuerdo a la validez de las respuestas,

lo que se resolvió, en parte, con la programación ramificada, la cual utiliza las respuestas del alumno para controlar el material que se mostrará a continuación.

Ruiz (1996), plantea que desde principios de la década del 60, las computadoras sirvieron como base a los sistemas de enseñanza automatizados. Los materiales de la denominada enseñanza programada que se desarrollaron consistían en secuencias de pasos pequeños para asegurar que las respuestas fueran correctas con el fin de que el aprendiz recibiera un conocimiento reforzado.

Esta constituyó la idea central en el desarrollo de software educativo, especialmente en los primeros programas asistidos por computador, progresivamente, con el paso del tiempo, en la medida que los educadores han adquirido mayor experiencia en el uso de las tecnologías de la información, y éstas han logrado incorporarse a los ambientes de aprendizaje, el diseño se ha ido haciendo más complejo y los principios pedagógicos que lo representan son más variados.

Barrios (1997), expone que a fines de los años 60 las investigaciones en esta esfera habían languidecido considerablemente, aunque existía la convicción de que la Enseñanza Asistida por Computadora debía proporcionar nuevos e importantes cambios a la enseñan; el mismo autor plantea que en 1971 la National Science Foundation (NSF) de los Estados Unidos de América decide invertir 10 millones de dólares durante cinco años en los proyectos Time Sharing Interactive Computer Controlled Information (TICCIT) y Programmed Logic for Automatic Teaching Operation (PLATO), la finalidad de estos proyectos era demostrar que la Enseñanza Asistida por Computadora podía proporcionar mejor enseñanza a menor costo.

En esa época las computadoras eran bastante costosas y de interfaces poco flexibles, por ese motivo las premisas que rigieron aquellos proyectos no son exactamente las reglas que se emplean actualmente para la producción de ambientes educativos asistidos por computadora.

El estancamiento en la producción de aplicaciones educativas se produce nuevamente a finales de los años 70 y hasta comienzos de los 80, debido principalmente a la falta de madurez del desarrollo tecnológico: los reducidos rendimientos y prestaciones de las computadoras en comparación con sus costos,

según plantea Lezcano (1998), y también debido al escaso desarrollo conceptual y metodológico. Un importante momento para los sistemas de Enseñanza Asistida por Computadoras en particular y para el mundo en general fue, sin lugar a dudas, el surgimiento del micro procesador, que rompió con las grandes limitaciones inherentes a grandes centros de cálculo asociados a costos muy elevados e inalcanzables para los sistemas de educación de cualquier país.

En los últimos años se han diseñado e implementado ambientes de aprendizaje poderosos, que se compenetran con las características de los procesos de aprendizaje en forma efectiva y que involucran una nueva concepción del aprendizaje, como los sistemas SESE, SEP y Progen, desarrollados por Lezcano (1998), en el Centro de Estudios Informáticos de la Universidad Central de Las Villas. Los dos primeros utilizan técnicas de Enseñanza Asistida por Computadoras para la enseñanza de tópicos básicos de la disciplina Inteligencia Artificial; el último es una herramienta que genera en forma automatizada sistemas expertos que utilizan el lenguaje Prolog como máquina de inferencia, la que, unida al método que la sustenta, permite establecer una relación importante entre las asignaturas de Programación Lógica e Inteligencia Artificial.

En la actualidad se abren las puertas al desarrollo de las técnicas de Inteligencia Artificial, Aprendizaje Reforzado (Reinforcement Learning) y agentes inteligentes que ayuden a la navegación por los sistemas de enseñaza automatizada. (Bello et al., 2000; Holland et al., 2000)

Diversos autores, como Bravo (2000), Julian y Botti (2000), consideran que los agentes constituyen el próximo avance más significativo en el desarrollo de sistemas y pueden ser considerados como la nueva revolución en el software.

Teniendo en cuenta las evaluaciones del conocimiento, los agentes deben tener acceso a bases de datos que permitan tomar decisiones a partir de una diversidad de situaciones y de acuerdo a la apreciación del estado actual del conocimiento del estudiante.

Un momento importante en la aplicación de la tecnología en los procesos docentes lo constituye el desarrollo de las redes de información que ofrecen herramientas como la World Wide Web, los motores de búsquedas, el correo electrónico, las herramientas para la discusión y la conferencia, para el trabajo en grupo y de

colaboración, así como las tecnologías de presentación multimedia, hacen posible el desarrollo de comunidades de aprendizaje a través de la comunicación entre personas ubicadas en diversas partes del mundo, de manera relativamente poco costosa. (Lucero, 2004)

Si la infraestructura de apoyo al proceso docente lo permite, se considera vital el uso de la interactividad y las comunidades de aprendizaje, mediante ellas se logran la retroalimentación y el enriquecimiento del proceso, para el alumno y el profesor.

1.2.2 Caracterización psico-pedagógica

Hodson (1996) y Holland et. al. (2000), reconocen que la Enseñanza Asistida por Computadoras no siempre ha disfrutado de una buena reputación, pero en los últimos años con el incremento de la potencia de las computadoras y del software multimedios, los estudiantes tienen la posibilidad de observar la simulación del comportamiento de los sistemas que están estudiando, a la vez que se les presenta un gran cúmulo de información bien dosificada de acuerdo a estrategias pedagógicas cuidadosamente diseñadas, no obstante, en cada caso, es necesario ser minucioso al elegir el tema que se tratará y la forma en que será abordado.

Gorga y Madoz (2000), asimismo, consideran que lo más importante es introducir la aplicación educativa siguiendo un método, los cambios en los medios implican necesariamente profundos cambios metodológicos.

Paralelo a la aparición de nuevos medios tecnológicos se han desarrollado estrategias y conceptualizaciones para fundamentar que las herramientas computacionales constituyen un método para descubrir, describir, organizar y cuantificar la información.

1.2.2.1 Los ambientes de aprendizaje potenciados por la tecnología.

Una vía necesaria para incorporar las Tecnologías de la Información y las Comunicaciones en el contexto académico, es crear ambientes que conviertan el aprendizaje en significativo, el que concibe las ideas expresadas simbólicamente y

relacionadas esencialmente con lo que el alumno conoce, produciéndose una modificación de la información recién adquirida. (Ausubel et al., 1997)

La construcción del conocimiento comienza con la observación y reconocimiento de eventos y objetos a través de conceptos que ya se poseen.

Otro elemento importante en la teoría de Ausubel (1997), es la posibilidad de incorporar información nueva a un sistema organizado de conocimientos previos en el que existen elementos que tienen alguna relación con los nuevos.

A juicio de varios autores, una estrategia pedagógica a emplear para la Enseñanza Asistida por Computadoras aplicada a las Ciencias de la Computación es la enseñanza significativa y el enfoque constructivista. (Lezcano, 1998; Chestlevar, 2001)

1.2.3 Técnicas de enseñanza constructivista asistidas por computadoras.

Para Vigotsky (1988), el alumno no descubre el conocimiento, sino que lo construye, en base a su maduración, experiencia física y social, es decir al contexto o medio ambiente. El punto crucial y definitorio del enfoque constructivista, según Piaget (1989) y Duffy (1991), se manifiesta en la posibilidad humana de abstraer en los objetos algunos pocos rasgos para construir criterios de agrupamiento de los objetos abstraídos.

Los primeros pasos en el uso de software educativo como material didáctico con el enfoque constructivista fueron dados por Papert (1999), el cual desarrolla una línea de software que corresponde a los lenguajes para el aprendizaje y de ella surge el lenguaje LOGO, que a partir de su desarrollo en el Instituto Tecnológico de Massachussets (MIT) fue y es utilizado en numerosas escuelas y universidades en un sentido constructivista del aprendizaje.

Existen técnicas de enseñanza que se enmarcan en el enfoque constructivista asistido por computadoras, como son:

- **Ambientes de modelación**. Permiten que el aprendiz construya modelos, por ejemplo LOGO y los ambientes basados en él. (Papert, 1999)

- **Ambientes hipermedios**. Presentan la información en forma no lineal permitiendo que los estudiantes pueden navegar por ella a través de un dominio con enlaces predefinidos entre pequeñas piezas de información. El aspecto constructivista está en el hecho de que el aprendiz determina la secuencia de aprendizaje, pero los conocimientos del dominio en sí se presentan en forma expositiva. (Bravo, 2000)
- **Simulación por computadoras**. Son programas que contienen modelos del mundo real. La acción básica del aprendiz es realizar cambios en las variables de entrada y observar las consecuencias sobre las variables de salida. (Hennessy, 2003)
- **Ambientes de mapas conceptuales**. Permiten formar conceptos relacionados entre sí. (Cañas y Novak, 2004)
- **Ambientes colaborativos.** Buscan propiciar espacios en los cuales se favorezca el desarrollo de habilidades individuales y grupales a partir de la discusión entre los estudiantes al momento de explorar nuevos conceptos. Lo innovador en los ambientes colaborativos es la introducción de la informática a estos espacios, sirviendo las redes virtuales de soporte, lo que da origen a los ambientes Computer-Support Collaborative Learning - Aprendizaje colaborativo asistido por computador (CSCL). (Lucero, 2004)

Podría definirse el aprendizaje colaborativo como el conjunto de métodos de instrucción y entrenamiento apoyados con tecnología, así como de estrategias para propiciar el desarrollo de habilidades mixtas, aprendizaje y desarrollo personal y social, donde cada miembro del grupo es responsable tanto de su aprendizaje como del de los restantes miembros del grupo. (Lucero, 2004)

Según Heao (2004), diseñar ambientes de aprendizaje que permitan integrar el potencial de las Tecnologías de la información y las Comunicaciones, a través de recursos didácticos en línea, supone proveer experiencias de aprendizaje auténtico que permiten al que aprende desarrollar conocimientos significativos, aplicables y facilitar actividades y oportunidades interactivas.

En ambientes colaborativos, se espera que la tecnología apoye el pensamiento creativo, autoaprendizaje, compromiso, responsabilidad, participación, organización, crecimiento individual y grupal.

1.2.3.1 Los medios y herramientas que apoyan el aprendizaje.

La enseñanza significativa se centra en procurar que el aprendiz llene los vacíos existentes en su estructura de memoria para lo cual toma en cuenta que los estudiantes no son receptores pasivos de conocimiento, sino por el contrario, participantes activos en la interpretación de los modelos (muchas veces analogías) que ellos mismos o el profesor les propone para que intenten aprender aquello que aún no saben. Ausubel (1997) es, probablemente, el autor que más desarrolla un modelo de aprendizaje que se apoya en los medios, considera que la materia queda mediada ante el alumno por un complejo entramado de recursos, siendo el papel del profesor el de un organizador de esos medios y del proceso.

Arana (2000), considera que en el proceso de comprensión e interacción del alumno con la realidad viene a mediar todo un conjunto de sistemas de representación que utilizan símbolos (verbales o no); Pérez (2001), añade que sin ellos tal comprensión e interacción se encuentra seriamente comprometida.

La clave de la utilización de los medios de enseñanza está en rentabilizar sus propiedades con el objetivo de aumentar el grado de relevancia de los conocimientos. (Álvarez, 2001)

Es por ello vital considerar que los aspectos más relevantes de los medios no son la información que transmiten ni su accesibilidad o disponibilidad para ser utilizados en el aula, sino, precisamente, qué símbolos utilizan, cómo los organizan y estructuran.
En investigaciones realizadas Estrada (2002) y Santillana (2002), corroboran que la tecnología, en sí misma, no afecta al aprendizaje, sino más bien, es el sistema de símbolos que utiliza y el modo en que se compatibiliza con los ya existentes, gestual, icónico, verbal, matemático, y que, en algunos casos, pudiera implicar alguna jerarquía.

Por tanto, una de las preocupaciones en la enseñanza es precisamente la de promover competencias (aprendizajes) en el alumnado que le faciliten comprender, explicar la realidad y resolver sus problemas. (Uviña et al., 2005)

La comprensión holística conlleva tanto a atender su heterogeneidad como a traducir los diferentes modos de representarla.

1.2.3.2 Los Mapas Conceptuales en la enseñanza

Basándose en el aprendizaje como procesamiento de información, Novak y Gowin (1988), introducen el Mapa Conceptual como una respuesta a la línea de Ausubel del aprendizaje significativo dentro del marco de un programa denominado "Aprender a Aprender". En ellos, el conocimiento está organizado y representado en todos los niveles de abstracción, situando los más generales e inclusivos en la parte superior y los más específicos y menos inclusivos en la parte inferior.

Novak (1991), ha descrito el acto de hacer mapas como una actividad creativa, en la cual el estudiante debe hacer un esfuerzo para aclarar significados, identificando los conceptos importantes, relaciones y estructura del contenido tratado. La creación de conocimiento requiere un alto nivel de aprendizaje significativo, los mapas conceptuales facilitan este proceso, por lo que resultan importantes en el aprendizaje, principalmente debido a que:

- Facilitan una rápida visualización de los contenidos de aprendizaje.
- Favorecen el recuerdo y el aprendizaje de manera jerárquica organizada
- Permiten una rápida detección de los conceptos claves de un tema, así como de las relaciones entre los mismos.
- Favorecen el desarrollo del pensamiento lógico.
- Los materiales elaborados utilizando Mapas Conceptuales facilitan el estudio independiente.
- Permiten que el alumno pueda explorar su conocimiento previo acerca de un nuevo tema, así como la integración de la nueva información que ha aprendido.

- Organizan los conocimientos a partir de las principales relaciones entre los conceptos.
- Favorecen el trabajo colaborativo.

Ontoria (1993), considera que los mapas conceptuales constituyen un recurso esquemático para representar un conjunto de significados conceptuales incluidos en una estructura de proposiciones. Estas pueden ser explícitas o implícitas. Los mapas conceptuales proporcionan un resumen esquemático de lo aprendido, ordenado de una manera jerárquica.

1.2.3.3 Mapas Conceptuales en el estudio de la programación.

La enseñanza de la programación, como parte fundamental de la formación profesional de las carreras de Ciencias de la Computación, ha sufrido cambios importantes que han estado aparejados al desarrollo de la tecnología y la enseñanza, muchos de ellos se han basado en el surgimiento de nuevos paradigmas de programación. (Castillo y Barberán, 2000)

Entre los principales obstáculos que aparecen para el aprendizaje y aplicación de un lenguaje de diseño de algoritmos pueden puntualizarse los siguientes:

- El alumno se ve necesitado de manejar un gran número de nuevos conceptos e integrarlos de manera significativa. En los algoritmos, las acciones complejas suelen definirse en términos de otras acciones más sencillas. Esto hace que la comprensión acabada de las acciones más simples redunde en beneficios para entender aquellas más complejas.
- Las acciones tienen dos aspectos que están estrechamente relacionados entre sí: una *sintaxis* (reglas de redacción de las acciones en un algoritmo), y una *semántica* (significado formal y preciso de una acción dada).
- El alumno se enfrenta a la necesidad de manejar un *lenguaje objeto* (para elaborar algoritmos) y un *metalenguaje* (para hablar acerca de cómo se comporta el lenguaje algorítmico).
- Existen conceptos relativamente complejos interrelacionados entre sí. (Chestlevar, 2001)

Escribir un programa de computadora utilizando un Lenguaje de Programación requiere del alumno varias competencias y habilidades, que involucran básicamente la capacidad de manipular un conjunto de abstracciones interrelacionadas para la resolución de problemas. En tal sentido, el proceso de enseñanza-aprendizaje de un Lenguaje de Programación es extremadamente complejo.

La tarea de aprender a manipular el conjunto de símbolos asociado a un lenguaje conforme a una *sintaxis,* relacionándolos con una *semántica,* demanda un esfuerzo considerable para los alumnos de los primeros años de la carrera. A esto se suma, en muchos casos, una formación deficiente que les dificulta organizar nuevos conceptos de una manera ordenada para construir taxonomías y diferenciar propiedades que permitan establecer pautas para razonar sobre ellas.

En este contexto, Stojanovic (2002), propone el uso de mapas conceptuales para la enseñanza de conceptos básicos de programación y desarrollo de algoritmos.

Respecto a las destrezas cognitivas, los Mapas Conceptuales desarrollan conexiones con ideas previas, la capacidad de inclusión, la diferenciación progresiva entre conceptos, la integración o asimilación de nuevas relaciones entre ellos. (Moreira, 2002)

Al revisar los textos tradicionales de enseñanza de programación en el ámbito universitario se comprueba que mayoritariamente no hacen uso de un lenguaje de diseño de algoritmos para enseñar a programar y, en su lugar, apelan directamente a un lenguaje de programación.
Ese acercamiento prescinde, muchas veces, de una clara identificación de cómo se interrelacionan distintos conceptos teóricos entre sí, el resultado es que muchos se presentan independientemente y sólo a través de la práctica el alumno llega a interrelacionarlos. Esto puede motivar la exploración de distintas técnicas didácticas que facilitan a los alumnos una mayor comprensión y vinculación de los temas presentados.

Los mapas conceptuales brindan una presentación integradora y ofrecen un recurso esquemático de lo aprendido, donde se muestran las relaciones jerárquicas y los niveles de abstracción. (Cañas y Novak, 2004)
En programación se da el caso particular de que todo concepto expresado a través de la *sintaxis* de un lenguaje tiene su correlación con un significado operacional (semántica), y dicho significado estará definido de manera composicional, en término del significado de otros conceptos más elementales.

El mapa conceptual, puede ser usado, entonces, como una herramienta de organización, asociación, validación, interrelación, discriminación, descripción y ejemplificación de contenidos, con un alto poder de visualización. La incidencia de los mapas conceptuales en la pedagogía moderna para definir y organizar planes de estudio, currículos, programas de asignaturas y para la acción directa en el proceso de aprendizaje ha trascendido las aspiraciones iniciales de su creador.

El trabajo con los Mapas Conceptuales permite que el proceso de enseñanza-aprendizaje se desarrolle centrado en el alumno y no en el profesor, atendiendo el desarrollo de destrezas, no conformándose sólo con la repetición memorística de la información por parte de alumno, pretendiendo un desarrollo armónico de todas las dimensiones de la persona, no solamente intelectuales.

1.2.3.4 Herramientas para la creación de mapas conceptuales.

Los mapas conceptuales son un aporte que en sus inicios, no estaba relacionado con las redes de cómputo o comunicación, pero al ser éstos un potente recurso educativo y la educación no ser más que una forma especial de comunicación, el uso de los mapas, empleando las redes de información se hace más eficiente tanto cuantitativa como cualitativamente. El desarrollo de las redes facilitó la aparición de aplicaciones o herramientas que permiten, con gran facilidad, compartir, crear y editar mapas conceptuales, en una institución e incluso a escala mundial. (Díaz y Leal, 2004)

En el desarrollo de esta investigación se evaluaron varias herramientas para desarrollar mapas conceptuales, algunas, como *Shared Space*, constituyen herramientas potentes, este último fue diseñado principalmente para la educación, ofrece facilidades para el trabajo colaborativo, las discusiones de temas, puede generar mapas compatibles con CMapTools, posee sofisticadas herramientas de navegación y búsqueda, apertura y creación rápida y fácil de nuevos espacios de información, eficiente ayuda en línea, pero sólo es compatible con arquitectura y sistema operativo Macintosh. (Copsey, 2005)

Otras son utilizadas con fines comerciales, como *MindMapper*, *MindGenius*, *ConceptDraw*, *VisualMind*; algunas se encuentran disponibles en Internet y de código abierto, como, *FreeMind*, escrito en Java, que constituye la primera aplicación libre para la creación de mapas conceptuales, aunque presenta limitaciones al añadir recursos multimediales al nodo o concepto del mapa y no permite el trabajo colaborativo. (GAelLimited, 2004; CSO, 2005; TechnM, 2005; BosleyGroup, 2006; MedioWiki, 2006)

1.2.3.4.1 Ventajas de CmapTools el diseño de mapas conceptuales.

Durante los últimos doce años, el “Institute for Human and Machine Cognition” (IHMC), ha estado desarrollando CmapTools, un software libre con ambiente cliente-servidor que permite, entre otras opciones, construir y compartir mapas conceptuales. El software se usa extensivamente por personas de diferentes edades y en una gran variedad de aplicaciones y se ha creado teniendo en cuenta la colaboración en ambientes educativos, principalmente para escuelas y universidades. (IHMC, 2005)

La herramienta CmapTools cumple con los requisitos indispensables para ser usada como parte de una estrategia de enseñanza significativa, enmarcada en la construcción del conocimiento, como es la propuesta realizada en este trabajo para la asignatura Programación II de la carrera de Ingeniería Informática.

La figura 1.1 muestra desde la ventana Vistas de la herramienta CmapTools toda la organización de las carpetas que contienen los mapas y recursos asociados a los conceptos, almacenados en el disco de la computadora local o a través del servicio remoto compartido en sitios con la comunidad del Cmap.

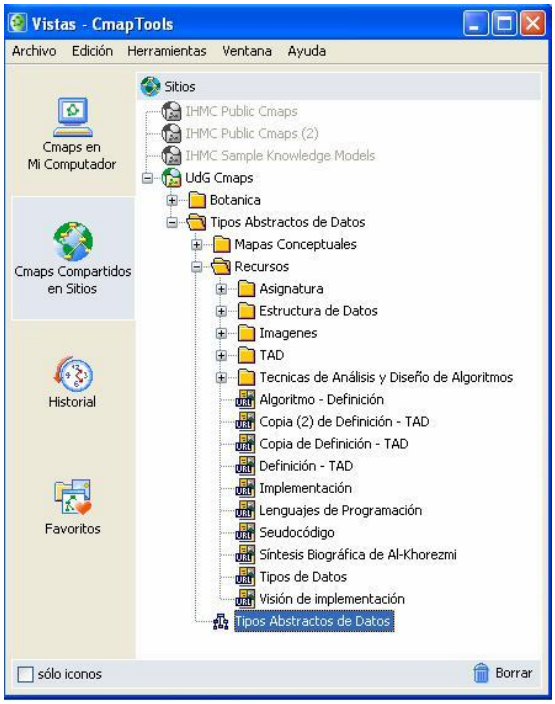

Figura 1.1. Ventana para visualizar la organización del CmapTools.

1.2.3.4.1.1 Recursos e informaciones en un mapa conceptual.

Una operación importante en la creación de un mapa conceptual es la de añadir información al mismo, a través de la inclusión de nuevos conceptos y relaciones hasta obtener un modelo de conocimiento. El CmapTools permite incluir un nuevo concepto, crear proposiciones o relaciones, crear carpetas, importar, adicionar y editar enlaces a recursos (Figura 1.2) y enlazar proposiciones entre varios mapas creados con CmapTools. (Cañas et al., 2003a)

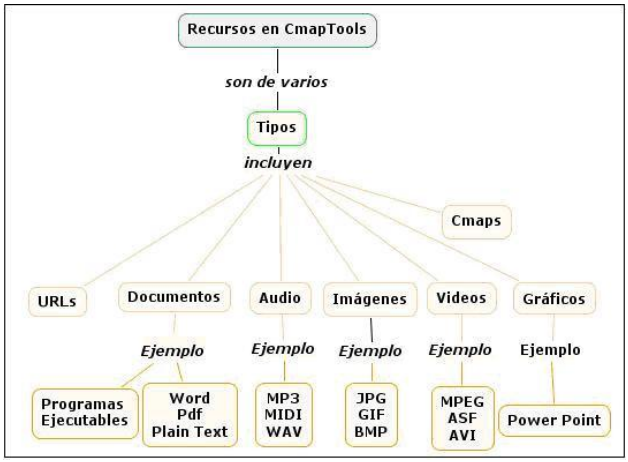

Figura 1.2. Recursos en CmapTools.

Esta opción es importante en el desarrollo de mapas conceptuales con fines docentes y para la construcción del conocimiento, ya que en su creación pueden encontrarse colaborando varios equipos de estudiantes, los cuales desarrollan de forma independiente mapas relacionados con diferentes temas que pueden unirse mediante relaciones entre conceptos afines, así se vinculan mapas de varias asignaturas o disciplinas.

1.2.3.4.1.2 Permisos y Control de Accesos

El CmapTools permite controlar la accesibilidad y permisos de una carpeta localizada en Cmaps en Mi Computador o en Sitios Compartidos. El administrador de un sitio autoriza a otros usuarios a crear, usar y acceder a los mapas, carpetas y recursos en ese sitio, también puede modificar los permisos ya existentes. (Cañas y Hill, 2004a)

Cuando no se ha especificado ningún administrador para una carpeta, la autoridad sobre la lista de permisos de la carpeta está determinada por el primer administrador que se encuentre, ya sea en el directorio padre, o en un subsiguiente directorio antecesor, o en la misma raíz. El directorio raíz puede estar dentro de Cmaps en Mi Computador o en un sitio de la red CmapTools.

1.2.3.4.1.3 Interacción de los actores en el proceso de autoaprendizaje y construcción del conocimiento.

La opción de controlar la accesibilidad permite personalizar el trabajo con los estudiantes en el proceso de enseñanza-aprendizaje del tema representado en el mapa conceptual, ya que el profesor/administrador concede los accesos y los permisos para modificar y añadir información a los alumnos/usuarios en dependencia al nivel de desarrollo alcanzado en relación con el tema que se trata. Uno de los propósitos en esta investigación, además de ofrecer una herramienta que organice los conceptos y los relacione creando proposiciones a través del mapa conceptual de la asignatura, es el enriquecimiento del mismo, a través de la inclusión de informaciones y recursos, resultado de la creación de los estudiantes.

1.2.3.4.1.3.1 Trabajo colaborativo.

La arquitectura cliente - servidor, junto con una colección de sitios Públicos (Servidores Cmap) en los que cualquier usuario de Internet puede crear una carpeta y construir, copiar o publicar sus mapas conceptuales, proporciona un ambiente adecuado para compartirlos y permite la colaboración durante la etapa de construcción (Figura 1.3). Los mapas guardados en estos servidores también generan automáticamente una página web y pueden ser observados por cualquier usuario solo utilizando un explorador de Internet. (Cañas et al., 2003a)

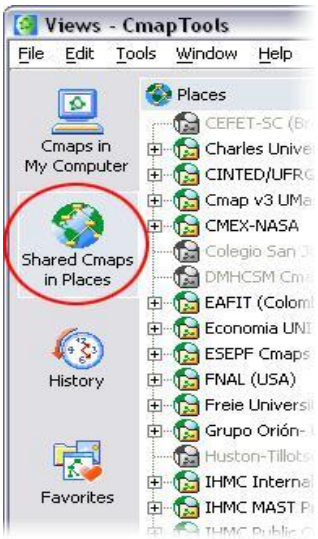

Figura 1.3. Sitios Públicos del CmapTools en Internet.

Las formas de colaborar son variadas, incluso existe una sesión de colaboración sincrónica donde los usuarios modifican el mapa concurrentemente, para lo cual se comunican a través de una ventana de Chat y por medio de Anotaciones (pos-it-notes) se facilita la revisión y colaboración por pares.
Los aspectos discutidos, se pueden agregar al mapa después de seleccionarlos y, a través de Hilos de Discusión, se adicionan a un nodo (concepto). El sistema apoya la construcción de modelos de conocimiento: grupos de mapas conceptuales y recursos asociados sobre un tema en particular.

CmapTools brinda la posibilidad de guardar el proceso de construcción del mapa conceptual, permitiendo al profesor evaluar y conducir su construcción y desarrollo. La reproducción también identifica cuál usuario llevó a cabo cada paso, y ese aspecto es esencial para apoyar el trabajo colaborativo. De hecho, la reproducción de los mapas conceptuales creados por un individuo revela los procesos por los cuales el aprendizaje significativo está ocurriendo.

A pesar del formato de estilo libre que pueden tomar, las características específicas de mapas conceptuales bien construidos (estructura, semántica, contexto, etc.) brindan una abundancia de información sobre la cual se pueden desarrollar

herramientas inteligentes que ayuden al usuario en el proceso de la construcción de los mapas. (Cañas y Hill, 2004b).

En la red Cmap comparten mapas conceptúales prestigiosas universidades del mundo como la de Extremadura, España; la de Los Andes, Colombia; de California Irving, de Massachussets y centros de investigación como el National Aeronautics and Space Administration, NASA en Estados Unidos; la Universidad Agrícola de Sydney; la Universidad Claude Bernard Lyon, Francia, la de Aizu; Japón y el Institute for Human and Machine Cognition, entre otros. (IHMC, 2005)

1.2.3.4.1.4 Uso de Listas de discusión.

Las listas de discusión en el mapa conceptual pueden añadirse a proposiciones (forman parte del modelo de conocimiento), conceptos y frases de enlace, los cuales son partes del mapa conceptual, el cual puede ser almacenado en un servidor Cmap. Tienen un creador, al cual se le solicita su ID+contraseña. A ellas acceden también otros usuarios a través de una participación anónima o mediante autentificación.

Tienen una fecha de vencimiento y asociado también un mensaje al usuario sobre la fecha en que expira. Permiten suministrar información a través del correo electrónico a los usuarios. (Cañas et al., 2003b)

Su uso aporta una relación constante de los estudiantes y el profesor y entre estudiantes que colaboran y aprenden sobre el tema propuesto con lo cual se llega a formar una comunidad en torno a un interés común.

En las listas de discusión creados para discutir un tema relacionado con un concepto del mapa se pudo comprobar que los estudiantes, además de plantear sus dudas, sugerencias y presentar incluso sus algoritmos al criterio del profesor y el resto de los miembros de la listas de discusión, se preocupan por la calidad de los textos tanto en contenido como en redacción, ortografía, gramática y otras características del lenguaje escrito.

El profesor como coordinador principal debe escoger los temas de mayor dificultad para el estudiante y en los que la orientación al trabajo independiente ha sido más

profunda. De ser posible es recomendable seleccionar a un estudiante que colabore como facilitador, esto ayuda a los usuarios a compartir información relevante a la temática, en discusión libre y amistosa.

1.2.3.4.1.5 El uso del Cmap en la modalidad de Educación a Distancia

La Educación a Distancia es una estrategia educativa basada en el uso intensivo de las tecnologías de la información y las comunicaciones, estructuras operativas flexibles y métodos pedagógicos altamente eficientes en el proceso enseñanza-aprendizaje, que permiten que las condiciones de tiempo, espacio, ocupación o edad de los estudiantes no sean factores limitantes o condicionantes para el aprendizaje. (Bravo, 2000)

Las Nuevas Tecnologías en la educación a distancia se practican en la actualidad, por numerosas instituciones, empleando redes de información globales, la tecnología computacional móvil, el desarrollo de la multimedia y los avances de la telecomunicación. Existen instituciones en el mundo, que trabajan exclusivamente a partir del modelo de educación a distancia, aprovechando los programas de las universidades tradicionales e incorporándolos, con diferentes modificaciones a este tipo de enseñanza. Otros han desarrollado Aulas Virtuales y han incorporado modernas aplicaciones interactivas para la comunicación. (Antúnez et al., 2004)

El mapa resultante de esta investigación se encuentra accesible desde todas las Universidades que pertenecen a la Red Nacional del Ministerio de Educación Superior, al utilizar la herramienta en red una vez que el mapa se ha guardado en el Servidor, automáticamente se almacena una versión que permite visualizarlo como una página web. Al realizar una búsqueda directamente en Internet, usando un buscador o metabuscador e incluyendo en el perfil de búsqueda las palabras claves necesarias aparecerá la página con la información que se encuentra en el servidor Cmap. En la figura 1.4 se muestra la información de la carpeta Tipos Abstractos de Datos del servidor Cmap de la Universidad de Granma.

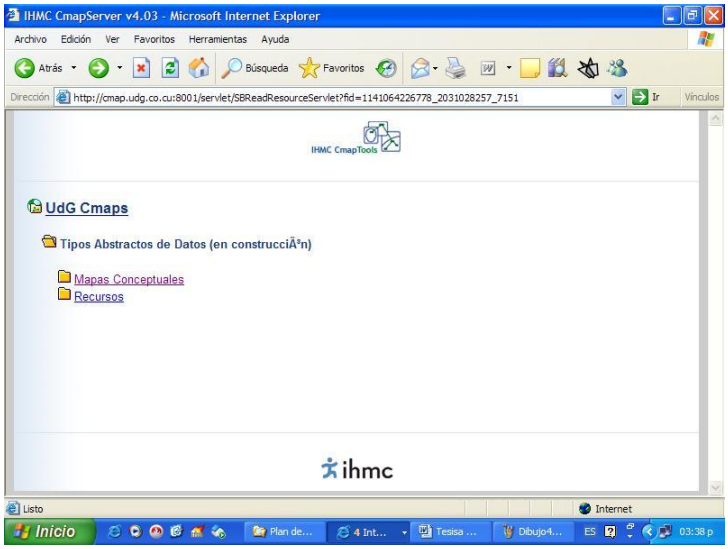

Figura 1.4. Página web que muestra la carpeta **Tipos Abstractos de Datos**

La herramienta CmapTools, permite al usuario realizar búsquedas en Internet de forma sencilla, introduciendo en una ventana el perfil de búsqueda, también posibilita realizarla en Servidores Cmap, incluyendo mapas conceptuales que están relacionados con la información buscada, puede trabajar automáticamente con los términos del mapa, de un concepto o frase de enlace, obteniéndose todos los recursos o mapas relacionados con este término. (Cañas y Hill, 2004a)

Este proceso tiene gran importancia en la creación del mapa y de la estructura y relación de los contenidos representados, ya que la información así obtenida puede usarse para mejorar el mapa propio.

La posibilidad de que el mapa diseñado como resultado del proceso de enseñanza, por parte del profesor, y enriquecido producto del autoaprendizaje en el caso de los alumnos se encuentre en Sitios Compartidos por el Cmap en Internet hace que el trabajo realizado sea útil a otros usuarios y sirva como retroalimentación al recibir sugerencias para mejorarlo.

También le permite al profesor comparar el proceso de construcción del conocimiento por parte de alumnos de diferentes centros de enseñanza, evaluar la calidad de las habilidades alcanzadas y rediseñar y enriquecer la estrategia de enseñanza utilizada.

El mapa conceptual propuesto para la asignatura Programación II puede ser utilizado, además, instalando la aplicación CmapTools (cliente), abriendo la carpeta UdG Cmap y seleccionando el mapa Tipos Abstractos de Datos. Estas ventajas de la herramienta CmapTools permiten usarla en la modalidad de educación a distancia, la cual da validez a los principios de educación para todos, aprender a aprender, la enseñanza-aprendizaje personalizada, la educación para toda la vida, es decir, la educación permanente o continua.

El módulo de comparación de mapas y generación automática de una versión HTML apoyan al usuario en el uso de los mapas en un ambiente educativo. Por los aspectos señalados se escoge esta herramienta para el desarrollo de la propuesta resultante de esta investigación.

1.2.4 Aplicaciones educativas en la enseñanza de la asignatura de Programación II.

En los últimos años se han intensificado los esfuerzos por diseñar aplicaciones de ayuda a la comprensión de los contenidos de la asignatura Programación II, entre ellas, la propuesta por Ruiz (1996), que muestra de forma gráfica las operaciones básicas con ciertas estructuras de datos, simulando el proceso que se ejecuta.

Señas y Moroni (2003), plantean la experiencia en el uso combinado de los llamados Esquemas de Ejecución Algorítmica (EEA), representaciones gráficas para visualizar la ejecución de un algoritmo y los Mapas Conceptuales Hipermediales (MCH). Los EEA ayudan a comprender la relación entre el algoritmo, como entidad estática, y el dinamismo de su ejecución, constituyen, además, una herramienta valiosa para la etapa de prueba y verificación. Por otra parte, el alumno crea el MCH del tema sobre el que trata el problema a resolver, lo que representa una ayuda significativa para la comprensión del mismo.

La herramienta EDApplets, una aplicación web orientada a la enseñanza-aprendizaje de los algoritmos y la programación en las ingenierías se basa en la tecnología de Applets Java, está orientada a la animación y visualización mediante trazas de algoritmos y estructuras de datos. La herramienta permite cubrir diversos

aspectos en una enseñanza que puede ser dirigida a distintos estilos aprendizaje. (Almeida et al., 2003)

Algunos ejemplos de herramientas informáticas para la creación de animaciones de algoritmos lo constituyen Balsa, Zeus, StarLite y Xtango, esta última ofrece una plataforma que soporta una serie de conceptos primitivos, útiles para la creación de animaciones, es fácilmente transportable, no requiere grandes recursos del sistema donde debe ser instalado, puede utilizarse en ambiente web y, finalmente, el modelo conceptual que utiliza es sencillo. (Axoft, 2005)

Estas aplicaciones resultan de mucha utilidad en la enseñanza de los temas de estructuras de datos, permitiendo que los estudiantes comprueben de forma gráfica el efecto (representación gráfica) que provoca la causa (algoritmo).

Sin embargo la lucha por lograr algoritmos eficientes va mucho más allá, se requiere en primer lugar seleccionar la estructura de datos adecuada para representar la información, diseñar el algoritmo y realizar el análisis de complejidad del mismo.

El análisis realizado permite corroborar que el enfoque constructivista se adapta a la enseñanza de la programación de computadoras y que en esta investigación se utilice para dar solución al problema planteado.

1.3 Resumen del capítulo.

La enseñanza es una tarea compleja que necesita de recursos que ayuden a hacerla más amena y productiva y el aprendizaje, al igual que cualquier evento social, es un fenómeno complejo, que se resiste a la redacción lineal y causal que a menudo se le impone; éste no puede ser interpretado como una simple secuencia de acciones, sino en términos de una danza estructurada, con una compleja coreografía de eventos.

Es por eso que teniendo como base los modelos teóricos presentados en este capítulo, hemos elegido, diferentes elementos del enfoque constructivista, de la enseñanza significativa y en especial del uso de los mapas conceptuales, incorporando a éste, el trabajo colaborativo en red y la simulación como medio de enseñanza. Es muy ambicioso pretender abarcar en una sola investigación todos

los temas que se tratan en la disciplina de Técnicas de Programación, por lo que se seleccionaron los de la asignatura Programación II, de modo que la experiencia lograda sirva para futuras generalizaciones hacia otros contenidos.

En el próximo capítulo se realizará un análisis del diseño de la carrera de Ingeniería Informática, fundamentalmente de la disciplina de Técnicas de Programación de Computadoras y la asignatura Programación II. Se evaluará la importancia de los temas de diseño y análisis de algoritmos que se tratan en la carrera de forma dispersa, lo que afecta considerablemente la integración de estos contenidos y su asimilación por parte de los estudiantes.

2 Capítulo II. Análisis de los contenidos de la disciplina Técnicas de Programación para Ingeniería Informática.

2.1 Introducción.

Desde 1960 comenzó el diseño a nivel mundial del currículo de las carreras de Computación. En 1968 la Association for Computing Machinery (ACM) publica el reporte Curriculum´68, el cual ofrece detalladas recomendaciones sobre los programas académicos. (ACM, 1968)

En nuestro país surge la carrera de Ingeniero en Sistemas Automatizados de Dirección en 1976. Desde su comienzo se mantiene la evaluación y perfeccionamiento de los Planes de Estudio, proceso continuo y labor ininterrumpida de la Educación Superior. Como consecuencia de ello, en determinados momentos, adquiere tal significación que se necesita modificar los planes de estudio. El ingeniero informático tiene como objeto de estudio el desarrollo de sistemas informáticos con inclusión o no de ayuda a la toma de decisiones, tanto en organizaciones productivas como de servicio, con el propósito de obtener un incremento en la eficacia y eficiencia de su funcionamiento con técnicas que le permiten delimitar los procesos computacionales, la información a procesar y las interrelaciones correspondientes; así como programar aplicaciones con alto nivel de profesionalidad. (MES, 1998a)

En el presente capítulo se realiza un esbozo de la carrera de Ingeniería Informática en el contexto nacional e internacional y su constante perfeccionamiento, debido, por una parte, al impacto que la tecnología ha tenido en la educación, y por otra a la incansable búsqueda de profesores e investigadores de nuevos ambientes de autoaprendizaje. Se hace énfasis en cómo la propuesta de éste trabajo se inserta en las orientaciones básicas para la elaboración del Plan de Estudios “D”, teniendo en cuenta la esencialidad del contenido, la semipresencialidad, el uso de las Tecnologías de la Información y las Comunicaciones en el Proceso de Docente Educativo y sobre todo su aporte a la nueva Estrategia Curricular de autoaprendizaje o “Aprender a Aprender”. Se analiza el diseño de la disciplina de

Técnicas de Programación y de la asignatura de Programación II, considerando la necesidad de incluir desde los primeros años los elementos de Diseño y Análisis de Algoritmos en la disciplina.

2.2 El diseño a nivel mundial de la carrera de Ingeniero en software.

En los últimos 14 años cuatro organizaciones han dirigido el desarrollo de los currículos de computación para los colegios y universidades del mundo: la Association for Computing Machinery (ACM), considerada como el centro de las investigaciones científicas que examinan nuevas estrategias y teorías que fundamentan la ciencia de la computación; la Association for Information System (AIS), fundada en 1994, apoya a los especialistas en Sistemas de Información; la Association for Information Technology Professionals (AITP), organizada en 1951, que congrega a los profesionales que usan la tecnología en los negocios y otras esferas; la Computer Society del Institute for Electronicaland Electronic Engineers (IEEE-CS) surgida en 1946 y ha suministrado recomendaciones a los currículos de ingenieros en computadoras, en software, tecnólogos en computación y de ciencias de la computación. Actualmente centra su trabajo en la especialidad del ingeniero en software. (ACM et al., 2004)

En 1990 los investigadores de la ACM y la IEEE-CS comienzan a vislumbrar que la computación se desarrolla ampliamente y en varias dimensiones, surgen nuevos programas de estudios y se hace difícil la elección de una u otra especialidad por parte de los estudiantes y la comunidad. Por estas razones emprenden la tarea de elaborar un reporte que proponga pautas orientadoras para el diseño de nuevas carreras en computación, que brinde una visión de los aspectos que identifican a cada especialidad y los temas que son comunes, con un detallado análisis de la incidencia en la carrera de los componentes organizacionales, de sistemas de información, aplicaciones tecnológicas, métodos y técnicas de software, infraestructura de sistemas, hardware y arquitectura de computadoras y el grado de teoría y práctica de cada uno. (ACM et al., 2004)

En 1998 se crea el Proyecto de Educación del Ingeniero en Software - Software Engineering Education Project (SWEEP) – que tiene como objetivo el rediseño curricular de la carrera, plasmado en el reporte SE2004, en el que participan varios países e instituciones interesadas en la calidad del graduado de este perfil. El reporte centró su atención en los conocimientos asociados al perfil del ingeniero en software y la pedagogía para impartirlos. (ACM et al., 2004)

En el reporte de la ACM y IEEE-CS (2005), se plantea que los avances tecnológicos influyen en la inclusión de nuevos tópicos en las carreras en esta etapa, debido, fundamentalmente, a la aparición y desarrollo de la World Wide Web y sus aplicaciones; la tecnología de redes, particularmente aquellas basadas sobre TCP/IP; los recursos multimedios; las Bases de Datos Relacionales; la Programación Orientada a Objetos; el uso de sofisticadas Interfaces de Programación (Application Programmer Interfaces, APIs); la seguridad y criptografía y los Dominios de Aplicación. Se definen nuevos contenidos, se proponen las disciplinas de cada carrera y las orientaciones metodológicas para que cada centro docente estructure el currículo propio según sus intereses y las demandas de la sociedad, teniendo en cuenta los temas esenciales.

La carrera de Ingeniero en Software tiene una base teórico-conceptual que incluye nociones de otros campos, como las matemáticas, las ingenierías, la administración de proyectos, las ciencias económicas y de otros diversos dominios de aplicación. En el Currículo del 2005 se define un área esencial del conocimiento, denominada Fundamentos de la Computación, que incluye cursos sobre Programación (control y datos, recursividad), Algoritmos, Estructuras de Datos/Representación (estáticas y dinámicas) y Complejidad, entre otras. En esta propuesta se analizan los temas comunes al resto de las especialidades de computación y los que le confieren identidad propia al Ingeniero en Software, como son, el análisis y modelación de problemas; diseño, verificación, validación, calidad y administración de software. (ACM y IEEE-CS, 2005)

2.3 Concepción y desarrollo de la carrera de Ingeniería Informática en Cuba.

A tono con el surgimiento y evolución de la carrera de Ingeniería en Software diseñada a nivel mundial, en Cuba se crea la especialidad de Ingeniero en Sistemas Automatizados de Dirección Técnico Económico (SAD-TE) con el objetivo de formar un especialista que comenzaba a ser necesario a la economía del país y debido a la cantidad de máquinas computadoras electrónicas, y otros medios técnicos de computación que se preveía fuesen introducidos paulatinamente en ministerios y empresas con el fin de hacer más eficiente la dirección y la gestión productiva y de servicios. Desde el principio se concibió al especialista con un perfil amplio de formación, que incluía la automatización de los sistemas de información y de toma de decisiones para la gestión y los procesos tecnológicos. (MES, 1998a)

La constante evolución de la informática obliga a que el Plan de Estudios de la carrera sea sumamente flexible para asimilar los cambios tecnológicos que tienen lugar, esto requiere que periódicamente se analicen y evalúen los resultados alcanzados y la calidad de los egresados, permitiendo perfeccionar los planes y programas, manteniendo la actualización técnica y añadiendo nuevos contenidos necesarios para la formación integral del Ingeniero.

El MES (1998a), representado en la Comisión Nacional de la carrera, luego de seis cursos de aplicación, valora integralmente el Plan de Estudios C y cataloga sus logros de muy satisfactorios teniendo en cuenta la calidad de la preparación del graduado y se constata la necesidad de introducir modificaciones que lo adapten a los nuevos requerimientos; no solo provenientes del avance de la ciencia y la tecnología de la computación y las comunicaciones, sino también de la sociedad y la economía cubanas en las que la informática ha continuado introduciéndose y desarrollándose vertiginosamente.

2.3.1 Modificaciones introducidas al Plan de Estudios C del Ingeniero Informático.

En el Plan de Estudios C Modificado, desarrollado por la carrera, MES (1998), se introducen variaciones, entre las que se encuentran:

- Otorgar mayor importancia a la transmisión de datos y redes de computadoras. Se tiene en cuenta que la integración de los medios de comunicación con los medios de cómputo ya es una realidad cotidiana.
- Se concibe la integración entre los proyectos de curso y la Informática Aplicada en la etapa de Práctica Profesional en todos los años de la carrera. Se refuerza la creación de habilidades de trabajo en grupo y la ejercitación de roles propios de un grupo de proyecto, para crear habilidades, tales como, programar lo especificado por otro, especificar para que otro programe y dirigir programadores.

Con estos cambios se logra una expresión más madura del Plan de Estudios original, lo que redunda en la formación de un graduado mejor preparado para dar respuesta a las necesidades y realidades nacionales; pero al mismo tiempo de un nivel equivalente al promedio internacional en la profesión. Se derivan las dos disciplinas integradoras, columna vertebral de la carrera, Técnicas de Programación de Computadoras e Ingeniería y gestión de software.

2.4 Sistema de conocimientos y propuesta de organización de la disciplina Técnicas de Programación de Computadoras.

Los elementos que componen la disciplina de Técnicas de Programación estuvieron presentes desde el primer Plan de Estudios “A”, elaborado para la especialidad de Ingeniería en Sistemas Automatizados de Dirección. (MES, 1998b) Al analizar los Planes de Estudios, desde el surgimiento de la carrera, se comprueba que el desarrollo de los contenidos abordados ha sido muy dinámico, especialmente en los últimos años, pues se ha tratado de impartir conocimientos y desarrollar habilidades acordes con el acelerado progreso de esta rama en el mundo.

El Ingeniero Informático requiere una profunda preparación en las técnicas modernas de modelación de datos y de programación, ya que ellas constituyen el instrumento más importante que le permite al egresado hacer un uso eficiente y verdaderamente profesional de las computadoras en cualquier actividad de las

más diversas esferas. Por todo ello, el objeto de estudio de esta disciplina está constituido por la modelación de los datos, tanto en memoria interna como externa; el diseño de algoritmos y la teoría de lenguajes.

La disciplina de Técnicas de Programación mantiene una estrecha relación con el resto de las disciplinas y es la responsable de transmitir los fundamentos de la programación, identificada al perfil terminal del graduado.

La mayor vinculación se logra a través de los Proyectos de Curso y las Prácticas Profesionales, las que tienen como objetivos fundamentales en primer año la solución de problemas simples usando la filosofía orientada a objetos; en segundo, el diseño de estructuras de datos y algoritmos complejos, desarrollando y poniendo a punto programas en lenguajes de alto nivel y documentar programas; en tercero, la solución de problemas de gestión mediante aplicaciones que hagan uso de los Sistemas de Gestión de Bases de Datos.

En el análisis del sistema de conocimientos de la disciplina Técnicas de Programación, realizado en esta investigación, se considera que existen aspectos que contribuyen al dominio de las técnicas de programación de computadoras, como son que los contenidos están dosificados de los más simples a los más complejos, las asignaturas de Práctica Profesional aportan el componente laboral al proceso docente y consolidan los contenidos impartidos, la distribución de las horas lectivas para cada asignatura es adecuada.

2.4.1 Consideraciones sobre el diseño de la disciplina.

Teniendo en cuenta los resultados docentes de la disciplina integradora Técnicas de Programación en los cuatro años de experiencia de la carrera en la Universidad de Granma, se determina que existen deficiencias en el diseño que influyen negativamente en el proceso de enseñanza-aprendizaje, fundamentalmente en el desarrollo de las habilidades en el diseño de algoritmos.

Las asignaturas Lógica Matemática, Matemática I, Algebra Lineal e Introducción a la Informática constituyen prerrequisitos para recibir las de Técnicas de

Programación. En la Figura 2.1 se muestra la organización de la disciplina, las asignaturas tienen un nivel de precedencia de acuerdo con el sistema de conocimientos y habilidades propuestos; se imparten Programación I, Programación II, Programación III, Sistemas de Bases de Datos; estas son, a su vez, precedentes de Programación IV y Proyecto de Programación. Práctica Profesional I se desarrolla al concluir la asignatura Programación I, Práctica Profesional II tiene como precedentes Programación II y Programación III y la Práctica Profesional III; Sistemas de Bases de Datos, Programación IV y Proyecto de Programación.

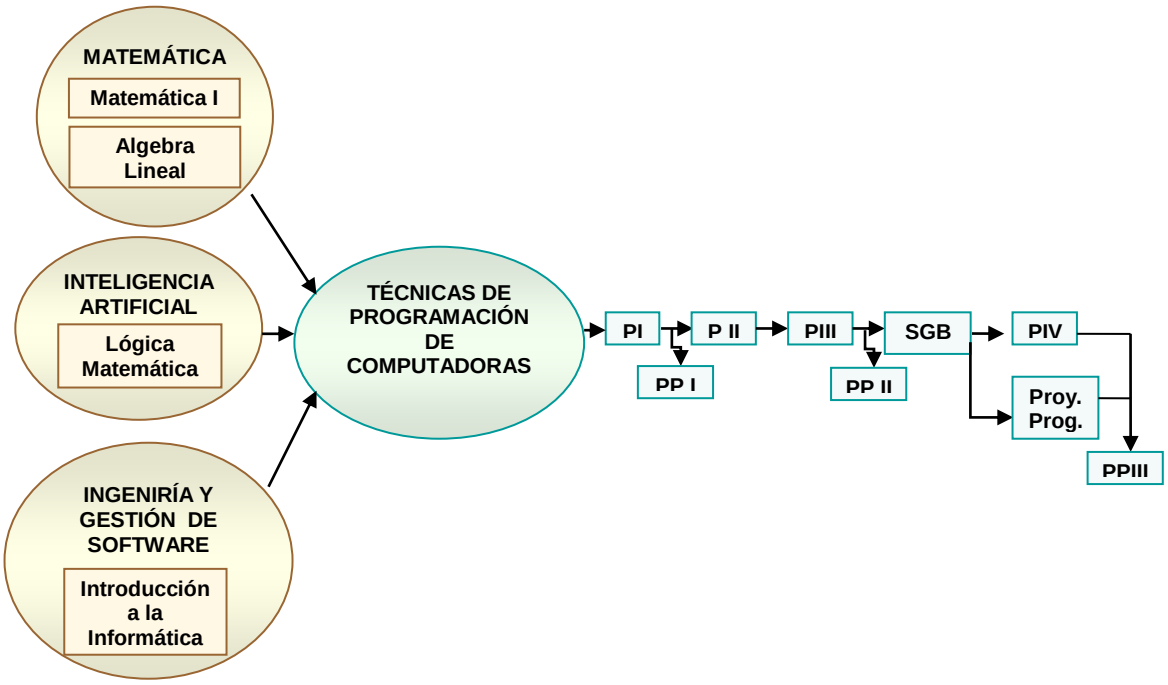

Figura 2.1. Precedencia de las asignaturas en la disciplina de Técnicas de Programación.

El sistema de conocimientos de la asignatura de Introducción a la Informática incluye elementos para describir algoritmos y programación en un lenguaje de alto nivel, profundizando más en los conceptos básicos para la utilización de máquinas computadoras, la historia de la informática (nacional e internacional), elementos de ética informática, software de uso general, aplicación de los principios del enfoque sistémico, definición de sistema, tipos, ejemplos, ciclo de vida de un proyecto de

Sistema Automatizado, Internet y sus posibilidades y el estado actual de los medios técnicos más utilizados.

La asignatura Programación I es la encargada de lograr que los alumnos diseñen algoritmos simples para resolver tareas de poca y mediana complejidad, expresando el algoritmo de solución de un problema mediante algún instrumento de descripción formal, aplicando los principios de la programación al diseño de un algoritmo, programándolo en un Lenguaje Orientado a Objetos, utilizando los arreglos en la solución de problemas de mediana complejidad. La asignatura tiene 96 horas y un Proyecto de Curso. (MES, 1998b)

En el análisis realizado se constata la alta carga de nuevos contenidos que debe asumir esta asignatura, por una parte los temas de diseño de algoritmos y por otro la implementación de los mismos en un Lenguaje de Programación Orientada a Objetos, filosofía que reciben por primera vez y que requiere capacidad de pensamiento abstracto y lógico que, en la mayoría de los casos están comenzando a desarrollar con la asignatura. (Guerrero, 2006)

Esta realidad afecta la calidad en el cumplimiento de los objetivos instructivos y del sistema de habilidades, por lo que los estudiantes de segundo año de la carrera presentan serias deficiencias en el diseño de algoritmos, incluso, de escasa complejidad.

2.4.1.1 Sistema de conocimientos de Programación II.

El estudio de las estructuras de datos es casi tan antiguo como el nacimiento de la programación, y se convirtió en un tema capital desde finales de la década de los 60. (Badía, 2004)

La asignatura Programación II del Plan de Estudios actual se desarrolla en el primer semestre del segundo año de la carrera, tiene como objetivos fundamentales diseñar las estructuras de datos en memoria interna más convenientes para la solución de un problema, definir los procesamientos sobre las estructuras de datos, en función de lograr una solución elegante y eficiente de un problema dado, diseñar algoritmos complejos utilizando técnicas avanzadas y

expresarlos con algún instrumento de descripción formal, definir la necesidad de realizar procesos de ordenamiento o búsqueda en memoria interna, eligiendo los métodos más adecuados, programar y poner a punto algoritmos complejos que incluyan el tratamiento de estructuras de datos, documentando los programas.

Se parte del presupuesto que los objetivos de la asignatura que la precede han sido cumplidos óptimamente, pero en un alto por ciento de estudiantes se mantienen insuficiencias, presentando, incluso, dificultades en la definición de arreglos y sus operaciones básicas. Ante esta realidad se considera necesario retomar la definición de esta tipo de datos, su modelo matemático y profundizar en la representación secuencial en memoria por la importancia que tiene para el diseño de estructuras más complejas, que constituyen objeto de estudio de Programación II. Es por ello que en la propuesta de este trabajo se incluye el mapa conceptual de arreglos dentro de las Estructuras de Datos lineales.

2.4.1.1.1 Propuesta de organización de los contenidos.

Los estudiantes de la carrera de Ingeniería Informática deben aprender a integrar teoría y práctica, reconocer la importancia de la abstracción y el modelado para adquirir un dominio eficiente del conocimiento que se requiere para apoyar el desarrollo de software en múltiples campos y para apreciar el valor de un buen diseño. Con la aparición de los lenguajes de programación estructurados, surge el concepto de tipo de datos, definido como un conjunto de valores que sirve de dominio de ciertas operaciones. En los lenguajes C, Pascal y similares, los tipos de datos se usan sobre todo para clasificar los objetos de los programas - variables, parámetros y constantes - y determinar qué valores pueden tomar y qué operaciones se les pueden aplicar.

2.4.1.1.1.1 Tipos Abstractos de Datos en el nivel primario de contenidos.

En la propuesta de diseño de la asignatura se parte del concepto de Tipos Abstractos de Datos y de forma sistémica se van complementando conceptos que no se habían logrado adquirir de forma sólida en las asignaturas precedentes, como algoritmos, abstracción, tipos de datos, arreglos, punteros.

Knuth (1973) y Lipschutz (1977) coincidieron en la necesidad de emplear una notación formal para describir el comportamiento de las operaciones con los datos, no sólo para impedir cualquier interpretación ambigua sino para identificar claramente el modelo matemático denotado por ellos. La definición de tipo abstracto de datos, abreviadamente, TAD, propuesto por Guttag y Liskov (1986), constituyó un aporte importante; análogamente a los procedimientos, los llamados tipos abstractos de datos constituyen un mecanismo que permite generalizar y encapsular los aspectos relevantes sobre la información (datos) que maneja el programa. Consideran un tipo de dato, no sólo como el conjunto de valores que lo caracteriza, sino también, como las operaciones que sobre él se pueden aplicar, conjuntamente con las diversas propiedades que determinan inequívocamente su comportamiento.

Se considera pertinente comenzar el estudio de la asignatura Programación II por la definición de abstracción, como mecanismo fundamental para la comprensión de fenómenos o situaciones que implican gran cantidad de detalles, es uno de los conceptos más potentes en el proceso de resolución de problemas, proporciona la capacidad de manejar un objeto como un concepto general, sin considerar la enorme cantidad de detalles que pueden estar asociados con dicho objeto. Sin abstracción no es posible manejar, ni siquiera entender, la gran complejidad de ciertos problemas.

El proceso de abstracción presenta dos aspectos complementarios, resaltar los aspectos relevantes del objeto y no tener en cuenta aspectos irrelevantes del mismo (la irrelevancia depende del nivel de abstracción, ya que si se pasa a niveles más concretos, es posible que ciertos aspectos pasen a ser relevantes).
El uso de la abstracción de datos durante el desarrollo del software permite al diseñador concentrase en cómo son usados los datos en el sistema para resolver el problema que le ocupa, sin tener que preocuparse por la representación y tratamiento de los datos en la memoria del computador, de ahí que se plantee que la verdadera utilidad de los Tipos Abstractos de Datos aparece en el diseño de nuevos tipos de datos. (Heileman, 2003)

A primera vista, puede parecer costoso, e incluso absurdo, dividir una aplicación en módulos y escribir procedimientos y funciones para controlar el acceso a la estructura que implementa un Tipo Abstracto de Datos, sin embargo esta metodología abunda en diversas propiedades interesantes:

- Abstracción. Los usuarios de un Tipo Abstracto de Datos no necesitan conocer detalles de implementación (tanto en lo que se refiere a la representación del tipo como a los algoritmos y a las técnicas de codificación de las operaciones), por lo que pueden trabajar en un grado muy alto de abstracción. Como resultado, la complejidad de un programa queda fraccionada entre sus diversos componentes.
- Corrección. Un TAD puede servir en las pruebas de programas, de manera que en una aplicación que conste de diversos tipos abstractos no tengan que probarse todos a la vez, sino que es factible y recomendable probarlos por separado e integrarlos más adelante. Evidentemente, es mucho más fácil detectar los errores de esta segunda manera, porque las entidades a probar son más pequeñas y las pruebas pueden ser más exhaustivas.
- Eficiencia. La separación clara entre un programa y los TAD que usa favorece la eficiencia, ya que la implementación de un tipo se retrasa hasta conocer las restricciones de eficiencia sobre sus operaciones, y así se pueden elegir los algoritmos óptimos.

2.4.1.1.1.2 Representación de la relación entre Tipos Abstractos, Estructuras y Tipos de Datos.

Con el objetivo de lograr una integración e inclusión de los nuevos conceptos con los ya conocidos desde la asignatura de Programación I se muestra, en la propuesta organizativa de los contenidos de Programación II, la relación entre los Tipos de Datos, las Estructuras y los Tipos Abstractos, ya que ellos constituyen diferentes niveles en el proceso de abstracción referida a los datos.

La implementación de un Tipo Abstracto de Datos supone una traducción de su especificación en la sintaxis de un lenguaje de programación particular. Esta traducción se compone de las declaraciones de variables apropiadas, que sean necesarias para definir los datos y un procedimiento o rutina de acceso que implemente cada una de las operaciones requeridas por el TAD. Entre el nivel de

los tipos de datos y el nivel de los tipos abstractos nos encontramos con las llamadas Estructura de Datos se refiere a una colección de variables que están relacionadas de alguna forma específica. Al nivel de las estructuras de datos, ya no tienen relevancia las operaciones que se puedan realizar sobre los componentes individuales de la misma, solamente la tienen las operaciones que implican a la estructura globalmente.

Según refiere Heileman (2003), los datos son las propiedades o atributos (cualidades o cantidades) sobre hechos u objetos que procesa el ordenador. El tipo de datos, en un lenguaje de programación, define el conjunto de valores que una determinada variable puede tomar, así como las operaciones básicas sobre dicho conjunto. Los tipos de datos pueden variar de un lenguaje a otro, tanto los tipos simples como los mecanismos para crear tipos compuestos.

En el nivel más alto de abstracción estarían los tipos abstractos de datos, estos se pueden ver como modelos matemáticos sobre los que se definen una serie de operaciones. El tipo abstracto de datos es una colección de valores y operaciones que se definen mediante una especificación que es independiente de cualquier representación. Para representar el modelo matemático básico de un TAD se emplean estructuras de datos y las operaciones definidas en el modelo se representan mediante procedimientos.

Un aspecto neurálgico en el análisis realizado con el objetivo de diseñar una estrategia que permita cumplir a cabalidad los objetivos de la asignatura, lo constituye el tema de la eficiencia en el diseño de algoritmos que solucionen problemas complejos.

2.4.1.1.1.3 Vínculo con los temas de diseño y análisis de algoritmos desde la asignatura de Programación II.

Al comparar los contenidos de las asignaturas de la disciplina con el sistema de habilidades que deben alcanzar los estudiantes en cada una, se revela que desde Programación I se exige eficacia en la selección de las estructuras para representar la información y la elaboración de programas que hagan un uso eficiente de los recursos de un sistema de cómputo. Sin embargo en el currículo de

la carrera de Ingeniería Informática no existe una asignatura encargada de los contenidos relacionados con las técnicas de diseño y análisis de algoritmos. Estos temas se han distribuido entre las asignaturas de la disciplina de Técnicas de Programación.

En la asignatura Programación IV, que se imparte en el segundo semestre de tercer año, se plantea explícitamente el tema de calidad en la programación, conjuntamente con fundamentos de lenguajes y compiladores y diseño de la interfaz de los sistemas, temas, en sí mismos, altamente complejos. La asignatura cuenta con 84 horas lectivas lo que dificulta la profundización y ejercitación adecuada en estos tópicos. (MES, 1998b)

En base al análisis del currículo de la disciplina, desarrollado en este trabajo, y a la calidad de las evaluaciones, tanto Pruebas Parciales como los Proyectos de Cursos y Exámenes Finales, se propone la inclusión de una asignatura en la disciplina que trate de forma exclusiva estos temas, contribuyendo al desarrollo del pensamiento lógico, algorítmico y la capacidad de análisis necesarios para el diseño de sistemas informáticos eficientes.

Esta modificación puede realizarse sin alterar las horas totales de la disciplina, ya que la asignatura Programación III cuenta con 80 horas y el contenido a impartir es relativamente sencillo, por lo que puede utilizarse un determinado por ciento de horas en la asignatura que se propone.

En el presente trabajo se incluye para cada estructura de datos, el análisis de complejidad de las operaciones que sobre ellas se realizan, vinculando, los algoritmos que usen técnicas de diseño con los nodos del mapa conceptual en que esta técnica se desarrolla, como una vía para profundizar en el diseño y análisis de los algoritmos desde los contenidos de la asignatura Programación II.

2.4.1.1.1.4 Paradigmas de la programación usados para implementar estructuras de datos.

En el mapa conceptual que organiza los contenidos de la asignatura se incluye información relacionada con los lenguajes y paradigmas de la programación que implementan las estructuras de datos.

El modelo general de las computadoras, desde que fue esbozado por von Neumann, no ha variado significativamente, mientras que la invención humana para proponerse nuevos problemas a resolver, usando la computadora, parece no tener límites. (Aspray, 2000)

En consecuencia, los lenguajes de programación tienen que adaptarse a éstas crecientes necesidades y aumentar la expresividad para poder resolver problemas muy diversos y cada vez más complejos. Además, tienen que ofrecer cierta eficiencia en la ejecución. La selección adecuada, tanto de las Estructuras de Datos como de los Lenguajes de programación para implementarlas constituye un reto y un logro difícil de alcanzar.

2.4.1.1.1.4.1 Programación funcional.

Tradicionalmente los matemáticos han resuelto problemas usando el concepto de función, la cual convierte ciertos datos en resultados. Si se conoce cómo evaluar una función usando la computadora, se pueden resolver automáticamente muchos problemas. Este análisis fue realizado por algunos matemáticos que consideraron la computadora una herramienta importante para su trabajo, creando los lenguajes de programación funcionales, los que aprovechan la posibilidad que tienen las funciones para manipular datos simbólicos, y no solamente numéricos, y la propiedad que les permite componer, creando de esta manera, la oportunidad para resolver problemas complejos a partir de las soluciones a otros más sencillos. También se incluyó la posibilidad de definir funciones recursivamente. Un lenguaje funcional ofrece conceptos que son muy entendibles y relativamente fáciles de manejar. El lenguaje funcional más antiguo, y seguramente el más usado hasta la fecha, es LISP, diseñado por McCarthy (1995), en la segunda mitad de los años 50. Su área de aplicación es principalmente la Inteligencia Artificial. En la década de los 80 se mostró nuevamente interés por los lenguajes funcionales, añadiendo la tipificación y algunos conceptos modernos de modularización y polimorfismo.

Programar en un lenguaje funcional implica construir funciones a partir de las ya existentes. Por lo tanto es importante conocer y comprender bien las funciones que conforman la base del lenguaje, así como las que ya fueron definidas previamente, de esta manera se pueden ir construyendo aplicaciones cada vez más complejas. (Koschmann, 2001)

El uso del LISP en la implementación de Listas, como Estructura de Datos, hace sencilla y elegante la programación de las operaciones básicas y las aplicaciones a problemas concretos, como el cálculo de la derivada de funciones y trabajo con polinomios.

2.4.1.1.1.4.2 Programación Lógica.

Otra forma de razonar para resolver problemas en matemáticas se fundamenta en la lógica de primer orden. El conocimiento básico de las matemáticas se puede representar en la lógica en forma de axiomas, a los cuales se añaden reglas formales para deducir aspectos verdaderos (teoremas) a partir de los axiomas. Gracias al trabajo de algunos matemáticos, de finales de siglo pasado y principios de éste, se encontró la manera de automatizar computacionalmente el razonamiento lógico -particularmente para un subconjunto significativo de la lógica de primer orden- que permitió que la lógica matemática diera origen a otro tipo de lenguajes de programación, conocidos como lenguajes lógicos. (Lezcano et al., 2000)

En la programación lógica el proceso de descripción de un dominio de aplicación se lleva a cabo mediante la conceptualización de los objetos y las relaciones que hipotéticamente existen, lo cual se expresa a través de un lenguaje formal apropiado. La noción de un objeto puede ser amplia. Dentro de los lenguajes orientados a la lógica tiene significativa importancia el *Prolog*, que ofrece la ventaja de que él mismo constituye un motor de inferencia para sistemas expertos, con lógica de primer orden en cuyo caso el programa es realmente la base de conocimiento más un conjunto de datos de partida. Es el primer lenguaje lógico y el más conocido y utilizado. Las aplicaciones a la Inteligencia Artificial mantienen el lenguaje vivo y útil.

2.4.1.1.1.4.3 Paradigma de la Programación Orientada a Objeto.

A mediados de los años 60 se empezó a vislumbrar el uso de las computadoras en la simulación de problemas del mundo real. Pero el mundo real está lleno de objetos, en la mayoría de los casos complejos, los cuales difícilmente son representados por los tipos de datos primitivos de los lenguajes imperativos.
Así es como, Dahl y Nygaard (1970), definen el concepto de *objeto* y sus colecciones, llamadas *clases de objetos*, que permitieron introducir abstracciones de datos a los lenguajes de programación. La posibilidad de reutilización del código y sus indispensables modificaciones, se reflejaron en la idea de las jerarquías de *herencia de clases*. Se introduce el concepto de polimorfismo. Todos estos conceptos fueron presentados en el lenguaje Simula 67. Aunque diseñado como lenguaje de propósito general, Simula tuvo su mayor éxito en las aplicaciones de simulación discreta, gracias a la clase SIMULATION que facilitaba considerablemente la programación.

2.4.1.1.1.4.4 Lenguajes de Programación Orientados a Objeto (LOO).

La concepción de Programación Orientada a Objeto ha originado nuevos lenguajes para expresarla, los Lenguajes Orientados a Objeto (LOO). Algunos de los más importantes son *SMALLTALK, Eiffel, Actor, Modular.* (Group, 2003; Meyer, 2005)

También han surgido extensiones de lenguajes tradicionales, como LISP, Pascal, C, que pretenden establecer un compromiso con sus viejos programadores y las nuevas ideas. (Zarko, 2006)

2.4.1.1.1.5 Los Tipos Abstractos de Datos y su implementación en un Lenguaje Orientado a Objeto.

En el diseño organizativo que se propone, para un TAD se especifican las propiedades funcionales de una estructura de datos y sus operaciones, y éstas se implementan en términos de las construcciones existentes en el lenguaje.

Este concepto tuvo su primera materialización en el concepto de clase del lenguaje SIMULA 67. (Dahl et al., 1970)

Junto con las clases aparecen en este lenguaje el concepto de herencia, mediante la cual las subclases heredan las propiedades declaradas en la interfaz de las

superclases y pueden definir operaciones y datos adicionales que especializan el comportamiento de la subclase. SIMULA tuvo un gran impacto en muchos de los lenguajes de programación modernos.

En la asignatura se brinda información a los estudiantes sobre diferentes Lenguajes de Programación Orientados a objetos y las especificidades de cada uno.

MODULA-2 ofrece un excelente mecanismo de abstracción de datos, el cual incluye un cierto grado de protección ya que no permite acceder directamente al estado interno del TAD. Presenta algunas limitaciones como que los nombres de los procedimientos usados en un módulo tienen que ser únicos y que los TAD pueden operar un solo tipo de dato. MODULA-2 es uno de los primeros lenguajes de programación ampliamente difundidos en usar la modularidad como un principio fundamental de estructuración. La última versión conocida como Modula-3 fue diseñada por Luca Cardelli. (Cardelli et al., 1989)

ADA usa la sobrecarga de operadores (operator overloading) y las unidades de programas genéricas (generic program unit). La primera se refiere a permitir a un programador usar múltiples operadores con el mismo nemotécnico. La distinción entre los operadores puede ser determinada en tiempo de compilación examinando los tipos y el número de parámetros. En general, la abstracción de datos en ADA se realiza usando tres herramientas principales. El package usado para encapsular un conjunto de definiciones relacionadas. El type determina el dominio de una variable (o estructura de datos) y como ésta puede ser manipulada. La definición de generic permite generar varias abstracciones similares desde un patrón. (ANTARESX, 2004)

El estado alcanzado en el proceso evolutivo de los tipos de datos en los lenguajes de programación, el cual se pretende sintetizar, presenta algunas limitaciones. En SIMULA falta el mecanismo de información oculta. En ADA los packages son objetos de segunda clase. La sobrecarga de operadores en ADA se resuelve en tiempo de compilación. En ADA no aparece el concepto de clase.

El desarrollo de estos lenguajes ha dado paso a la actual filosofía de Programación Orientada a Objeto. En ella los programas están basados en objetos (entidades que combinan las propiedades de los datos y de los procedimientos que actúan sobre ellos), los cuales son instancias de un tipo o clase de objeto (descripción intencional del dato). Por eso, un objeto puede verse y usarse en una forma completamente análoga a un TAD en los lenguajes de programación tradicionales.

Si los TAD son similares a los objetos, se pudiera considerar a la Programación Orientada a Objeto simplemente como un modelo de programación en el cual todos los datos son abstractos y toda la manipulación de la información se implementa a través de las operaciones con los TAD, pero esto no es posible, ante todo porque los objetos tienen un tipo (dado en la mayoría de los casos por su clase) lo cual permite que sean valores de primera clase (pueden ser manipulados como una estructura de datos dentro del lenguaje) y ser usados directamente en expresiones. Además a la POO la caracterizan otros conceptos como herencia, polimorfismo y enlace dinámico.

2.4.1.2 Vinculación de la propuesta organizativa con las orientaciones del Plan de Estudios D.

En el año 2003 el Ministerio de Educación Superior cubana emite las orientaciones básicas para la elaboración del Plan de Estudios D, teniendo en cuenta las transformaciones que han tenido lugar en el mundo en relación con la Tecnología, las ciencias de la educación y las exigencias de la sociedad. (MES, 2003)
En el documento se recogen los aspectos fundamentales con vistas al perfeccionamiento de los Planes de Estudio y se le confiere una importancia vital a la determinación de los contenidos esenciales y la aplicación de medios que apoyen la enseñanza semipresencial y el autoaprendizaje.

El Documento Base también refiere que:

- Se deben producir cambios importantes en la actividad presencial de clases de los estudiantes con una tendencia a la disminución desde los primeros a los últimos años, a partir de la introducción de nuevos métodos en el proceso de

formación, que centren su atención principal en el autoaprendizaje de los estudiantes, y entre los cuales ha de desempeñar un importante papel las tecnologías de la información y las comunicaciones (TIC).

- Las asignaturas y disciplinas deben evidenciar un mayor nivel de esencialidad, centrando su atención principal en aquellos elementos del contenido que son fundamentales para el logro de los objetivos previstos en la carrera y que aseguren una adecuada secuencia lógica y pedagógica de los contenidos.

Así, a partir del currículo base, se abre un espacio complementario en los Centros de Educación Superior, de carácter más táctico, en el cual los marcos curriculares de su potestad, se han de perfeccionar constantemente.

Por todo lo anterior se considera que la propuesta del mapa conceptual que estructura el sistema de conocimientos de la asignatura Programación II y que muestra la relación con contenidos de asignaturas que le preceden y que no se encuentran diseñadas en la carrera se inserta en las orientaciones para el perfeccionamiento del Plan de Estudios.

2.4.1.3 La propuesta como apoyo al Sistema Bibliográfico de la asignatura.

El análisis realizado en esta investigación, constata que, no obstante a las modificaciones introducidas al Plan de Estudios C, aún existen carencias en la concepción del Sistema Bibliográfico de la disciplina. Las propuestas, en su mayoría, lo constituyen libros de textos impresos (basados en Lenguajes de Programación como Pascal, Delphi o C). No forma parte de las orientaciones metodológicas de la disciplina el uso de otros medios que exploten las ventajas de trabajo con Plataformas Interactivas, con herramientas de trabajo colaborativo en Red, ni el uso de software libre o aplicaciones de código abierto.

Sin embargo, las Comisiones de Carrera y los colectivos de disciplinas, en los diferentes Centros de Educación Superior, han hecho gala de inventiva y creatividad y han utilizado diversos medios y herramientas tales como sitios web, donde se vinculan informaciones sobre determinados contenidos y listas de discusión, sitios ftp con libros y software actualizados relacionados con las diferentes disciplinas y asignaturas y se integran, además, variadas técnicas

mediante el uso de Plataformas Interactivas, aplicaciones multimedias, tutoriales, entre otros.

Al incluir en los nodos del mapa conceptual de la asignatura información relacionada con los conceptos, aplicaciones, simulaciones, libros y listas de discusión se organiza, además, el sistema bibliográfico de forma directa a cada concepto. En consecuencia, el sistema propicia una educación centrada en el alumno, el cual en la misma medida que estudia un tema puede acceder a la documentación necesaria para profundizar en el mismo.

2.5 Resumen del capítulo.

Teniendo en cuenta el análisis del sistema de conocimientos de la disciplina Técnicas de Programación y en especial de la asignatura Programación II, la evolución de los Planes de Estudio, las orientaciones del Documento que resume las bases conceptuales que sustentan el diseño de los Planes de Estudios “D”, se comprueba la necesidad de diseñar nuevos métodos y medios para apoyar el proceso de enseñanza-aprendizaje.

Las deficiencias detectadas en el diseño de estructuras adecuadas para representar la información y la forma dispersa en la que se tratan los temas de Análisis y Diseño de Algoritmos, constituyen una oportunidad para desarrollar una herramienta pedagógica basada en mapas conceptuales que organice el sistema de conocimientos de la asignatura Programación II, integre al mapa el sistema bibliográfico y favorezca el proceso de autoaprendizaje de los estudiantes.

3 Capítulo III. Propuesta metodológica para la organización del sistema de conocimientos en la asignatura Programación II de la carrera de Informática.

3.1 Introducción.

El carácter activo del aprendizaje y su papel determinante, con respecto al desarrollo, permiten a la didáctica orientar el enfoque metodológico de las asignaturas y la organización de su proceso de enseñanza. En el contexto de la Educación Superior Cubana, tiene importancia trascendental la proyección de las asignaturas en todo plan de estudio, su contribución a la educación, la instrucción y el desarrollo de las nuevas generaciones.

El perfeccionamiento de la enseñanza exige una selección adecuada del sistema de conocimientos de las ciencias que se estudian y su consecuente estructuración lógica y psicológica en los diferentes Programas y Planes de Estudios; de modo que, en el contenido de enseñanza y en su concepción metodológica se encuentre claramente expresada su contribución a la formación y desarrollo multilateral de la personalidad de los estudiantes.

En estas circunstancias, adquiere especial interés la búsqueda de estrategias metodológicas relacionadas con las formas de concebir la estructuración didáctica de la asignatura en general, y de su enseñanza en particular; de modo que el contenido y los métodos de enseñanza satisfagan las nuevas exigencias de la sociedad.

En el presente capítulo se propone un sistema que no sólo se dirige a la instrucción, sino también al proceso de educación, al propiciar el autoaprendizaje y el trabajo colaborativo, haciendo uso de la herramienta CmapTools con el fin de organizar el sistema de conocimientos de la asignatura Programación II. Se presenta la propuesta metodológica para insertar este medio en el currículo de la asignatura y se evaluarán los aportes de la misma en el Proceso Docente Educativo.

3.2 Mapa conceptual que organiza el sistema de conocimientos de la asignatura Programación II.

En el análisis realizado al Proceso Docente de la asignatura Programación II se detecta como uno de los problemas en la asimilación de los contenidos, el nivel de abstracción que tienen los conceptos que manejan; la complejidad que puede alcanzar el diseño de las estructuras de datos, en dependencia del problema a resolver; la integración de las diferentes estructuras estudiadas en la solución de un problema; el logro de algoritmos eficientes y las escasas habilidades logradas en las asignaturas que preceden a Programación II.

Teniendo presente la caracterización de la Tecnología Educativa, la evaluación de la herramienta CmapTools y las deficiencias encontradas, se propone un sistema que facilite el trabajo colaborativo basado en mapas conceptuales y organice e integre el sistema de conocimientos de la asignatura.

3.2.1 Mapa *Tipos Abstractos de Datos*.

En el nivel primario del mapa se muestra el concepto **Tipos Abstractos de Datos** como un **modelo matemático** y las **operaciones** que se ejecutan sobre él, para profundizar más en el tema al nodo se le añade una página web con la definición y ejemplos.

En el diseño del mapa los conceptos ya conocidos se han representado con rectángulos, un ejemplo de ello son los de **Modelo Matemático** y **Operaciones**. En este nivel de su formación los alumnos son capaces de definir, debido a los conocimientos adquiridos en las asignaturas de Álgebra y Matemática I, el modelo matemático de los números enteros, complejos o las matrices y las operaciones sobre ellos, en este mapa se especifica que la unión de estos dos conceptos conocidos, definen un Tipo Abstracto de Datos. (Figura 3.1)

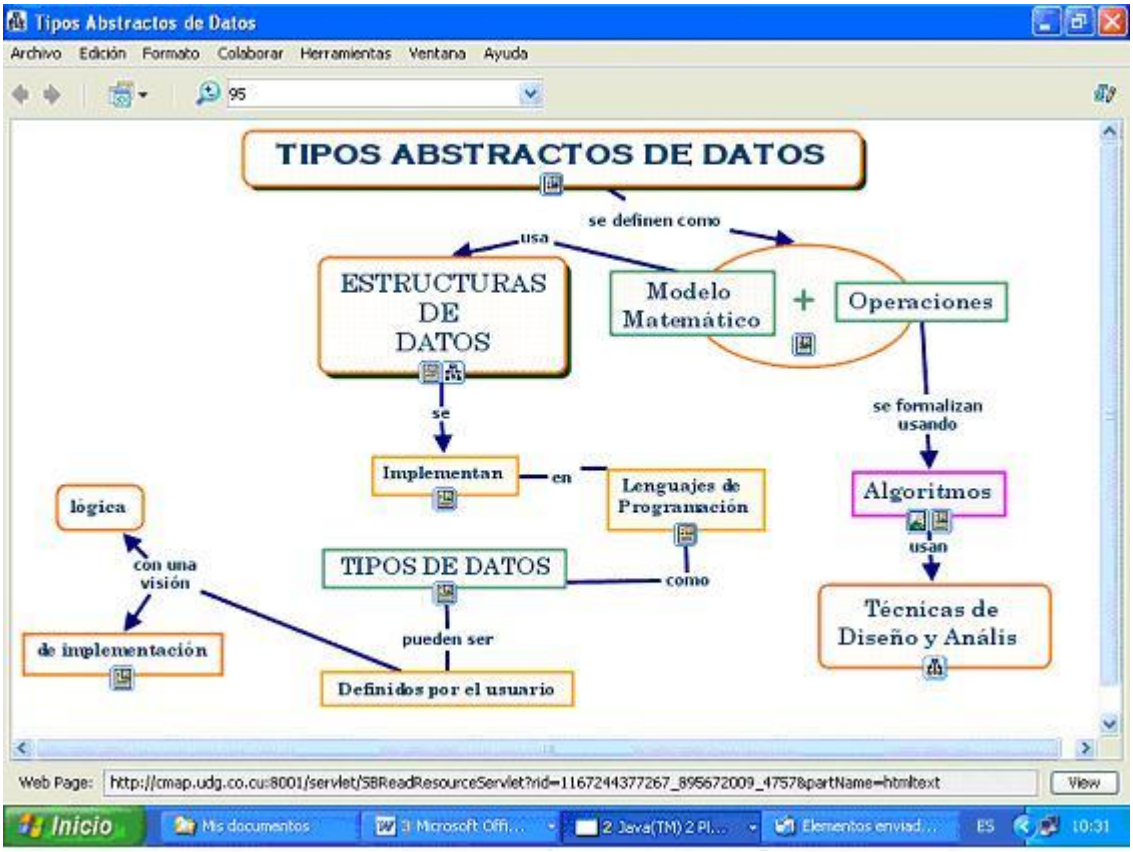

Figura 3.1. Mapa Conceptual Tipos de Datos Abstractos

De forma similar, los estudiantes se han familiarizado en la asignatura Programación I con los **Tipos de Datos** que pueden ser definidos por el usuario. En el nodo **Lenguajes de Programación** se brinda información sobre diversos paradigmas y Lenguajes, más detalladamente los Orientados a Objetos, sus semejanzas y diferencias.

De esta manera se pretende mostrar los diferentes niveles de abstracción en el diseño de los datos y su implementación.

3.2.1.1 Aporte a la sistematicidad e integración de conocimientos.

En el Proceso Docente Educativo de la asignatura Programación II se ha realizado un trabajo interdisciplinario que apoya la relación vertical y horizontal con la carrera. La asignatura Programación Descriptiva de la disciplina Inteligencia Artificial realizó un proyecto para la implementación de Listas usando el Lenguaje de Programación funcional LISP.

En el Proyecto de Curso de Programación II se ha buscado la integración con otras asignaturas del año, se logró el diseño e implementación de Colas usando el lenguaje Ensamblador que se estudia en Arquitectura de Computadoras y de diversos métodos matemáticos para el cálculo de raíces de polinomios. Estos resultados enriquecen el mapa conceptual y contribuyen a mostrar al estudiante los contenidos impartidos en la carrera como un sistema.

Es importante destacar el nivel de sistematicidad de los contenidos dentro de la propia asignatura, ya que lo aprendido vuelve a utilizarse en la solución de nuevos problemas y en el diseño de estructuras de datos más complejas, así, la representación secuencial es usada para diseñar pilas, colas, grafos, la diferencia entre ellas está dada, no por el modelo matemático, sino por las operaciones básicas que se efectúan sobre ellas, lo que permite simplificar el proceso de enseñanza relacionado con el modelo y hacer énfasis en el diseño eficiente de los algoritmos que implementan sus operaciones.

El establecimiento de nexos horizontales y verticales en la carrera permite guardar cierta secuencia temporal lógica y pedagógica en la presentación de temas interrelacionados o que se complementan, aunque pertenezcan a disciplinas distintas, asimismo se puede evitar la presentación de puntos de vista diferentes o reiteraciones, que no se sustenten en la adquisición de un nuevo aprendizaje o la transferencia a otro objeto de estudio.

Para establecer un nexo vertical se aborda el estudio de un mismo objeto basado en conocimientos previos, pero con una hondura y extensión mayor, se alude a la continuidad, en el momento de tratar los elementos esenciales del currículo, mientras que la secuencia se obtiene al avanzar en el aprendizaje con aspectos de mayor complejidad y abstracción.

La integración horizontal se establece mediante la presentación al alumno del objeto de estudio con las características esenciales de los sistemas de conocimientos científicos, y se le hace consciente que debe utilizar los

conocimientos de otra ciencia para su explicación, debido a la diversidad de objetos y a los vínculos y relaciones de interdependencia entre ellos.

3.2.1.2 Contribución al desarrollo de la cultura de la profesión.

Al adicionar información o recursos a los nodos del mapa conceptual se tuvo en cuenta que integrar los valores al aprendizaje de manera intencionada y consciente significa no sólo pensar en el contenido como conocimientos y habilidades, sino en la relación que ellos poseen con los valores.

El conocimiento posee un contenido valorativo y el valor un significado en la realidad, el que debe interpretarse y comprenderse adecuadamente a través de la cultura y por lo tanto del conocimiento científico y cotidiano, en ese sentido el valor, es sentimiento y afectividad en el individuo.

Así, el aprendizaje de un conocimiento profesional debe ser tratado en todas sus dimensiones: histórica, política, moral, es decir, subrayando la intencionalidad hacia la sociedad, donde se exprese la relación ciencia-tecnología y estén presentes los análisis cualitativos, los enfoques de procesos y la motivación, así la enseñanza-aprendizaje adquiere un nuevo contenido por su carácter integral.

La reflexión del profesor sobre el valor educativo de las acciones en el proceso, significa de igual modo, valorar el método de aprendizaje no como simple procedimiento sino como vía de comunicación, de lograr las relaciones interpersonales, es analizar el componente socio humanista de la ciencia que se enseña y de cómo hacerlo, lo que representa brindar un enfoque integral, complejo y dialéctico al aprendizaje, es reconocer que no existen dos culturas separadas, sino reflexionar sobre la totalidad de ésta, en su historia, en sus contradicciones, en su actualidad, en sus métodos, en sus consecuencias e impactos y, por supuesto en su ética.

En el mapa principal las operaciones de un Tipo Abstracto de Datos *se formalizan usando* Algoritmos, concepto ya estudiado en las asignaturas Programación I e Introducción a la Informática, a este nodo se asocia la definición; información sobre Pseudocódigos, usados para expresarlos independientes de cualquier lenguaje de programación; se aportan datos relacionados con la cultura de la profesión, la biográfica de Al-Khorezmi, imágenes de su monumento en Khiva; un sitio con

las contribuciones más relevantes realizadas a la ciencia de la Computación por diferentes investigadores como Blaise Pascal, Charles Babbage, George Boole, Augusta Ada Byron, John Von Neumann , Alan Mathison Turing , Donald Knuth, entre otros.

El nodo **Algoritmo** se asocia, a su vez, con el nodo **Técnicas de Diseño y Análisis de Algoritmos**, necesarias para lograr eficiencia en el diseño, el cual tiene como recurso un mapa que desarrolla estos contenidos.

Se muestra, además, la relación entre los conceptos, **Tipos Abstractos de Datos**, **Estructuras**, **Tipos de Datos** y los Lenguajes de Programación que los implementan. Desde este nivel del mapa conceptual se busca que los estudiantes asimilen la relación entre los conceptos nuevos y los ya conocidos, con el objetivo de que esta información tenga significado, sea lógica para él y de esta manera logre la inclusión de un concepto más completo y jerárquico en su representación del sistema de conocimientos del tema.

3.2.2 Mapa *Estructuras de Datos*.

En el mapa conceptual **Estructuras de Datos** se realiza la clasificación de las mismas según su asignación de memoria, en **estáticas** y **dinámicas**, y en **lineales** y **no lineales**, teniendo en cuenta su representación (Figura 3.2).

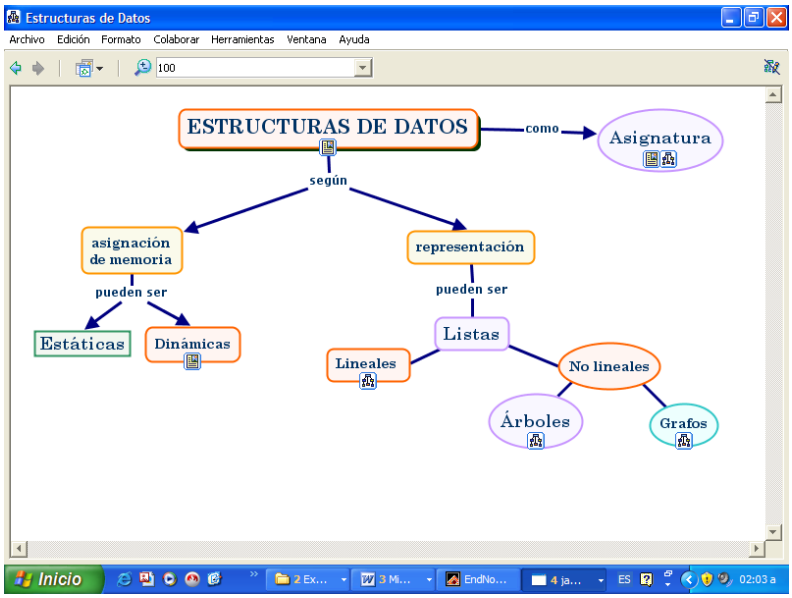

Figura 3.2. Mapa Conceptual de las Estructuras de Datos

En este mapa se relaciona el nodo de **Estructura de Datos** con el de la **asignatura** encargada de impartir este contenido en la carrera, al que se le agregan como recursos los Reportes de la ACM/IEEE, SE2004 y CC2005/2006, donde se detallan las propuestas de modificaciones al currículo de las diferentes carreras de perfil informático a nivel mundial, y un mapa que contribuye a organizar aspectos importantes relacionados con el Programa de la Asignatura Programación II en la carrera, como son, los objetivos, sistema de habilidades y valores, invariantes metodológicas, bibliografía de consulta, ejercicios resueltos y problemas propuestos.

3.2.2.1 *Estructuras de Datos lineales*.

En el mapa creado para organizar la información de las Estructuras de Datos con representación **lineal** se definen, los conjuntos, arreglos, pilas, listas enlazadas y colas. (Figura 3.3).

Esta concepción contribuye al desarrollo intelectual de los estudiantes y, en particular, del pensamiento teórico, pues posibilita que desde un primer momento estos comprendan la esencia del conocimiento que se desea formar, de esta manera el conocimiento generalizador precede y conduce, en lo posible, a la adquisición de conocimientos particulares.

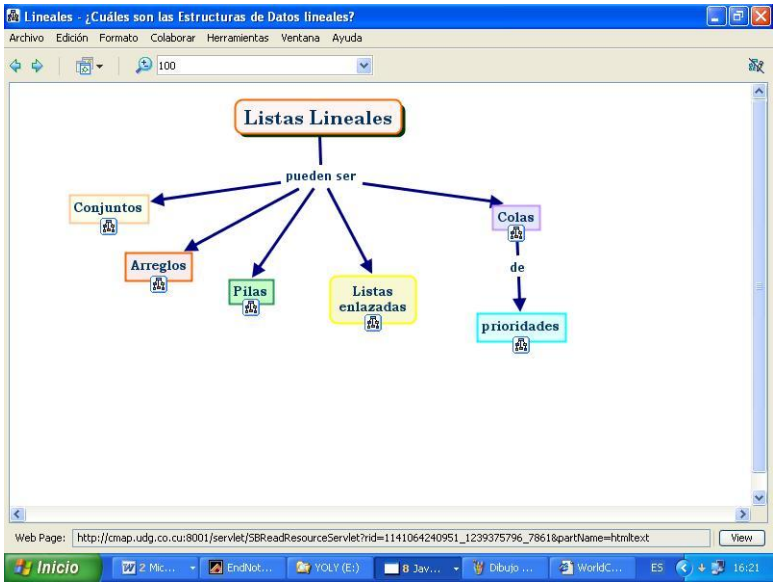

Figura 3.3. Mapa Estructura de Datos Lineales

Cada Estructura de Datos lineal es representada en un mapa conceptual que contiene su representación gráfica, estructura vacía, definición, ejemplos de aplicación, simulaciones, las operaciones básicas y sus algoritmos, los cuales

pueden hacerse más eficientes, en algunos casos, usando Técnicas de Diseño y Análisis.

3.2.2.1.1 Mapa conceptual *Arreglos.*

Los arreglos utilizan una representación secuencial en memoria, algunas de sus operaciones ya han constituido objeto de estudio en asignaturas anteriores, sin embargo, en Programación II se profundiza en algunos algoritmos como son el método de búsqueda binaria y los algoritmos de ordenamiento rápido o QuickSort, por inserción, mezcla y selección y las técnicas Divide y Vencerás, Ramificación y Poda usadas para obtener mayor eficiencia,

Al nodo **Arreglos** se asocia la definición, representación gráfica y algunos ejemplos de **aplicación** como la simulación de procesos del sistema operativo (Figura 3.4), donde la representación de la memoria y las operaciones se hacen a través de arreglos; en este sitio, añadido como recurso, se simula una secuencia de referencias a memoria sobre el espacio direccionable de varios procesos, permite seleccionar el tamaño de cada bloque de memoria de la lista de valores disponibles, o aleatorio si quiere que se determine al azar (el rango está entre los valores mínimo y máximo de la lista).

Del mismo modo se escoge el número de tareas que se estarán ejecutando simultáneamente en memoria y el tamaño en marcos de la memoria principal, y si se desea prepaginación o no, además de su grado; se elige la opción *prepaginar* siempre que se desea que las acciones se realicen **cada vez** que recupere el control de la CPU, de lo contrario sólo se realizará cuando no tenga ninguna página en memoria. Este proceso resulta difícil de entender sin un simulador al no ser visible por el estudiante. El sistema permite elegir el tamaño de las estructuras y se genera, además, una cadena de referencias aleatoria, se puede comprobar la cadena de referencias con otro algoritmo.

Al comenzar la simulación se muestran las tablas de páginas de los procesos y la tabla de marcos de memoria, a partir de este momento se puede apreciar la ejecución paso a paso, avanzando o retrocediendo en el algoritmo. Puede

observarse, también, la comparación del comportamiento de los algoritmos frente a una larga cadena de referencias, mostrando el número de fallos a medida que transcurre el tiempo.

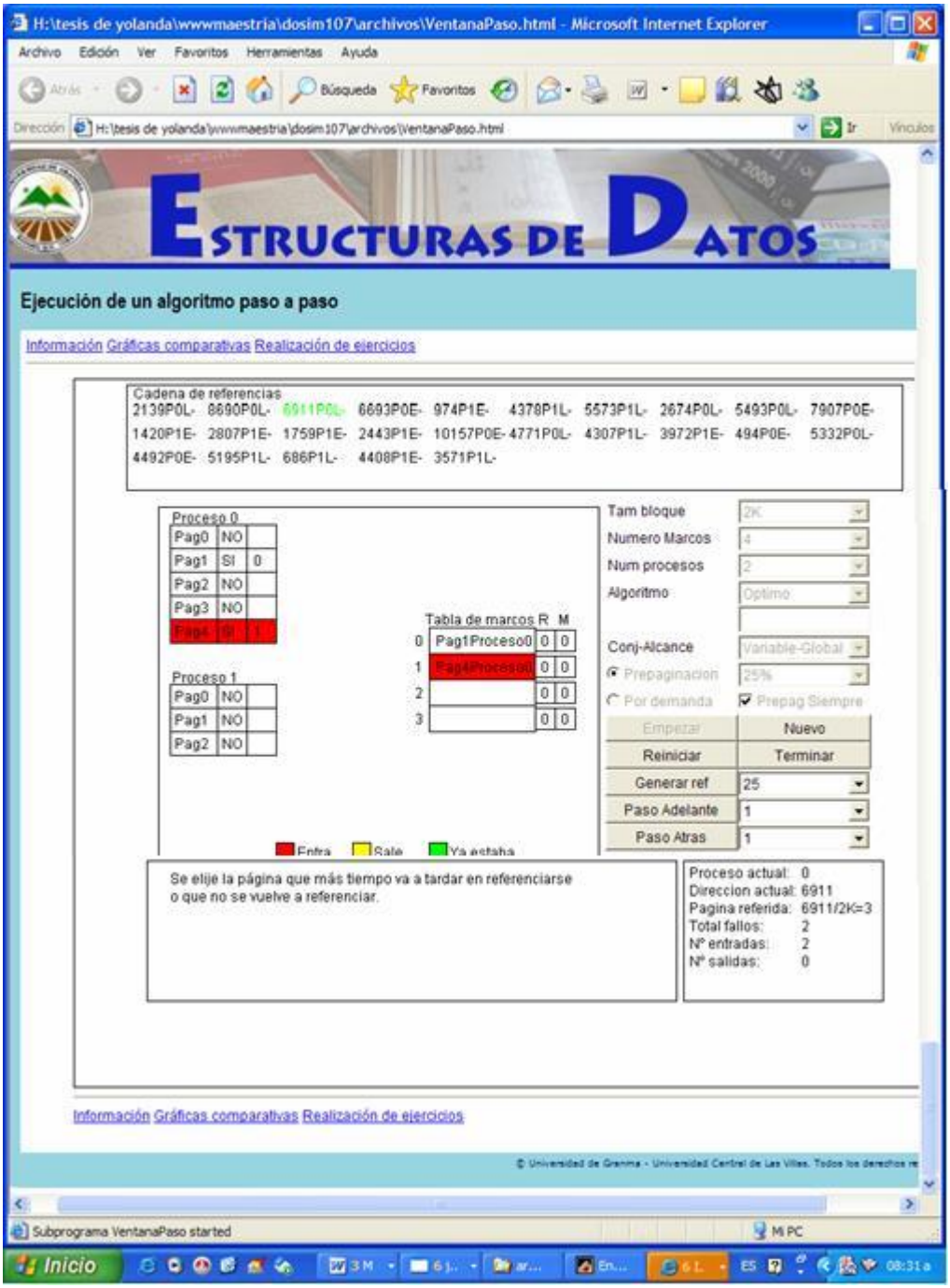

Figura 3.4. Simulador de Sistema Operativo.

3.2.2.1.1.1 Introducción al análisis de la complejidad en el estudio de los algoritmos.

A la operación de **Ordenamiento** se adicionan diferentes algoritmos o métodos, como puede observarse en la Figura 3.5. Para cada uno se ofrece un análisis de la complejidad teniendo en cuenta el mejor, peor y el caso medio, lo que permite a los estudiantes familiarizarse con el concepto de eficiencia.

Se parte para este análisis del algoritmo de ordenamiento Bubble, ya estudiado y fácil de entender, demostrando su complejidad cuadrática y por tanto su ineficiencia al compararlo con otros. Se adiciona una página web con información general sobre esta operación (Ordenamiento.htm) y en ella se introduce la notación ***O*** y se comparan los diferentes algoritmos teniendo en cuenta esta función.

El nodo cuenta, además, con problemas resueltos donde se utilizan algoritmos de ordenamiento, se mezclan algunos y en otros se propone introducir variantes a los estudiados para mejorar su eficiencia, en cada caso se realiza el análisis de complejidad y se compara con la de otros algoritmos similares.

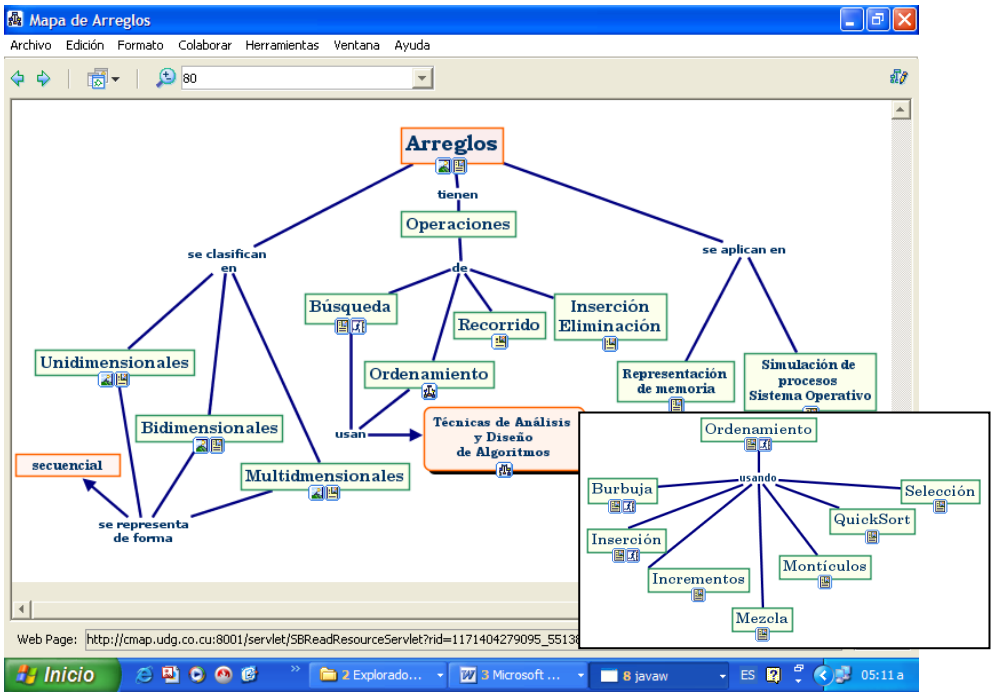

Figura 3.5. Mapa Estructura de Datos Lineal – Arreglo.

Como ya se ha planteado, una de las ventajas de la herramienta CMapTools es la posibilidad de adicionar a un nodo una lista de discusión, lo que facilita el trabajo colaborativo y la interactividad, alumnos-alumno y alumno-profesor. Para un mismo concepto pueden crearse varias listas de discusión.

Se añaden a los nodos **Ordenamiento** y **Búsqueda** aplicaciones que simulan estas operaciones, las cuales permiten visualizar el algoritmo y el efecto que cada acción provoca, explicando cada paso. Un ejemplo de ello se muestra en la figura 3.6 con dos aplicaciones que implementan los algoritmos de ordenamiento Bubble y por Inserción.

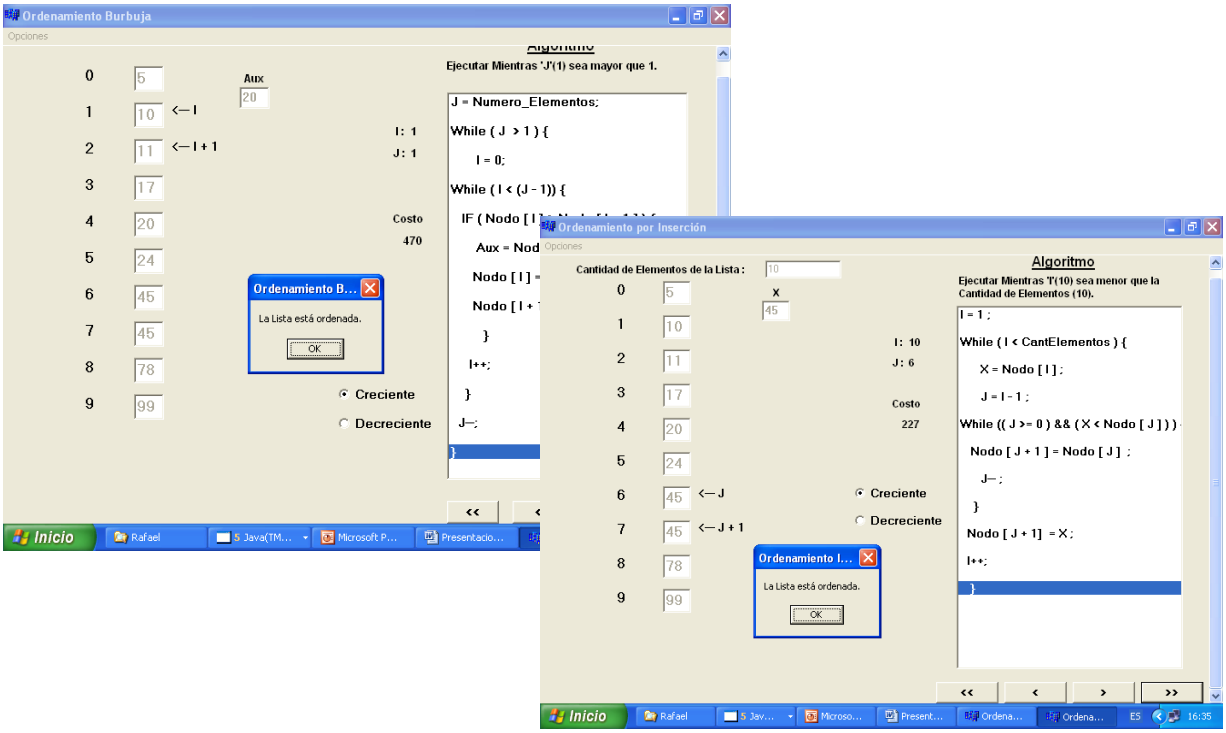

Figura 3.6 Aplicaciones que muestran los algoritmos de ordenamiento Bubble e Inserción.

Las operaciones de ordenamiento y búsqueda pueden ser más óptimas si se utilizan **Técnicas de Diseño y Análisis de Algoritmos**, es por ello que al orientar el estudio del método de Ordenamiento Rápido o Búsqueda Binaria se sugiere una visita al nodo **Divide y Vencerás** de este mapa.

3.2.2.1.2 Mapa conceptual *Listas.*

Aplicando la misma metodología de organización de la información usada en Arreglos se estructuran los conceptos del mapa **Listas**. (Figura 3.7)

Se introduce el concepto de puntero y se compara la representación enlazada con la secuencial, mostrando la conveniencia de utilizar una u otra en la solución de problemas. En los ejemplos de aplicación se incluye un programa para el cálculo de las raíces de un polinomio por el método de Bisección.

Las listas pueden clasificarse en simplemente enlazadas, doblemente enlazadas, múltiples, circulares, con nodo cabecera. Las operaciones básicas sobre ellas son las mismas, pero sus algoritmos se diferencian en el manejo de los punteros, se muestran las ventajas de usar cada una al solucionar problemas y la representación gráfica que ayuda a la comprensión de estos procesos.

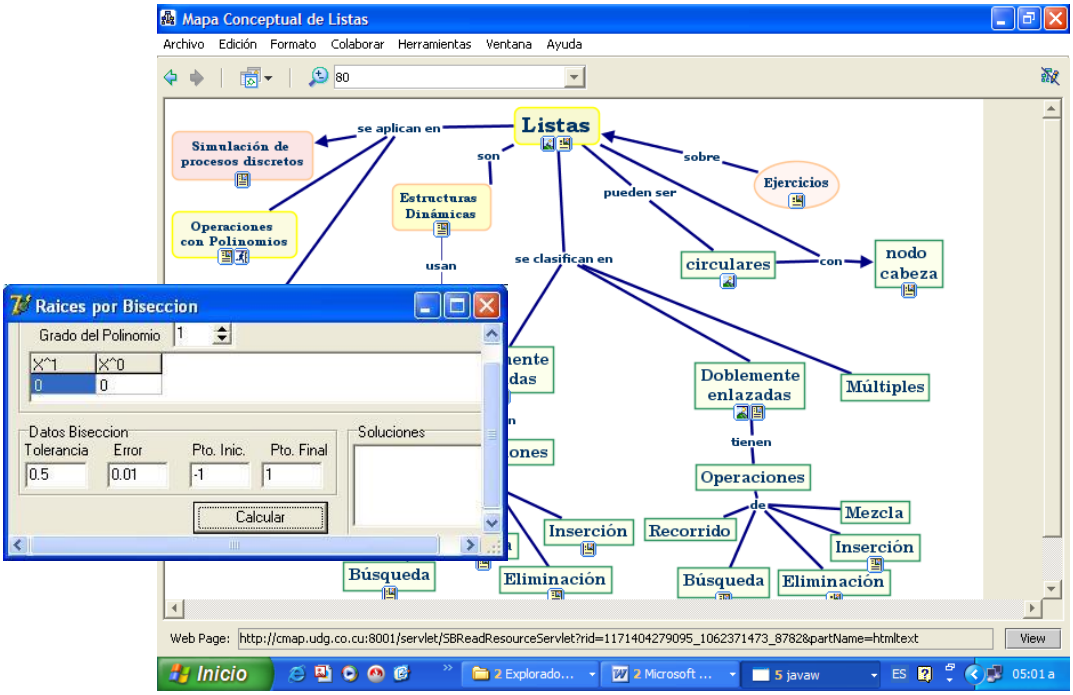

Figura 3.7 Mapa Conceptual Listas.

3.2.2.1.3 Mapa conceptual *Pilas.*

En el mapa conceptual de la estructura de datos **Pila** se muestra que puede ser representada de forma secuencial o enlazada, sus dos operaciones básicas PUSH y POP y sus aplicaciones. (Figura 3.8)

3.2.2.1.3.1 Aplicaciones de Pilas. Procesos Recursivos.

En el mapa se manejan conceptos relacionados con procesos difíciles de entender, como la recursividad, la que se presenta como herramienta para resolver problemas particulares que, en algunos casos, los estudiantes ya han resuelto utilizando algoritmos iterativos, se busca la motivación en la presentación de ejercicios. Para calcular el término n-ésimo de la secuencia de Fibonacci se explica que en 1202 Fibonacci propuso el problema en su libro Liber Abaci de la siguiente forma: ¿Cuántos pares de conejos se producen a partir de una única pareja en un año si cada mes cada par de conejos da nacimiento a un nuevo par, el que después del segundo mes se reproduce, y no hay muertes?, se explica que los números que aparecen en la solución de este problema se conocen como números de Fibonacci y que pueden ser definidos recursivamente. Se explica brevemente la biografía de Fibonacci.

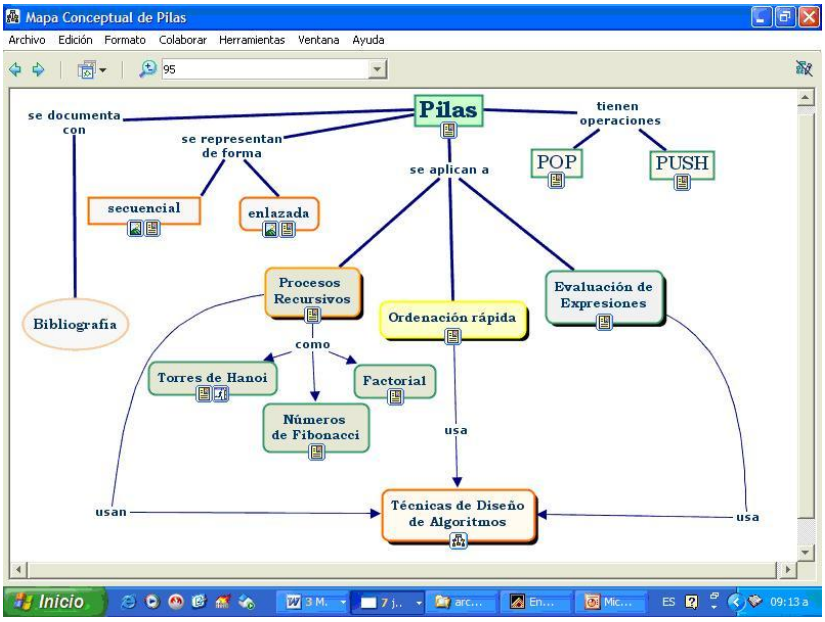

Figura 3.8 Mapa Conceptual Pilas.

Otro ejemplo es el conocido como Torres de Hanoi. Se presenta la leyenda que cuenta que en Benarés (India), el Dios creador Brahma entregó a los monjes tres vástagos sobre una base de bronce. Ensartó entonces 64 discos de oro, todos de dimensiones distintas, en una de las varillas, dispuestas de modo que el mayor estuviera en la base y los discos fueran decreciendo en tamaño. Y ordenó a

los monjes que moviesen toda la Torre de Brahma a otro de los vástagos, de modo que en cada traslado sólo fuese movido un disco dorado, y de manera tal que nunca un disco tuviera debajo otro de menor tamaño. Al final sentenció: “Cuando hayáis acabado la tarea, el mundo se vendrá abajo como montaña de polvo”. Si bien solucionar el juego no es muy difícil, el número de movimientos necesarios para hacerlo crece exponencialmente conforme aumenta la cantidad de discos.

Se ha comprobado que los estudiantes logran apropiarse del conocimiento que se imparte con más rapidez y eficacia si se acompaña de la historia relacionada con el fenómeno o proceso que se estudia, a la vez que se va desarrollando la cultura de la profesión y se contribuye al desarrollo integral del estudiante.

Se presenta la versión recursiva del algoritmo de ordenamiento rápido o QuickSort, que ya había sido estudiado de forma iterativa, enfatizando en su elegancia y eficiencia.

3.2.2.1.4 Mapa conceptual *Colas.*

En el nodo **Colas** se incluye información relacionada con la definición, se introduce la teoría de colas, el origen, el modelo de formación y los elementos que lo componen.
En el mapa se relacionan los conceptos de prioridades, las operaciones básicas de Inserción y Eliminación, las diferencias entre colas de prioridades y aquellas que no la tienen y se presentan ejemplos de aplicación.
Se muestran las formas de representar las colas en memoria, de forma secuencial, enlazada o combinando ambas. (Figura 3.9)
En correspondencia con la representación usada al diseñar la cola, se presentan los algoritmos de **Inserción** y **Eliminación**. Se exponen las ventajas de usar una u otra representación teniendo en cuenta el problema a resolver.

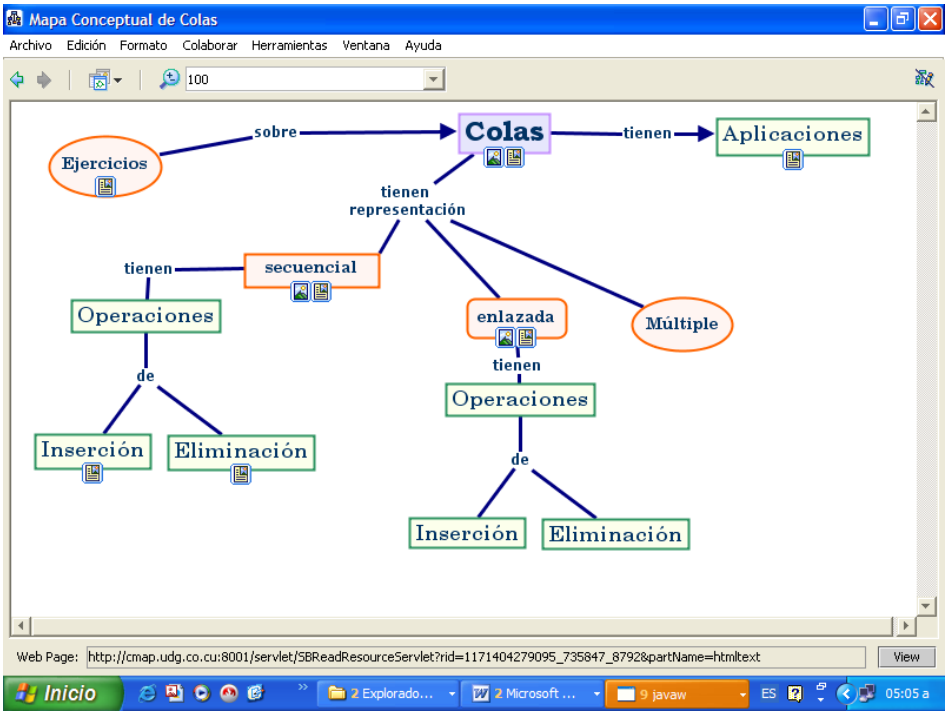

Figura 3.9 Mapa Conceptual Colas.

3.2.2.2 Estructuras de Datos no Lineales.

En la información contenida en los mapas de **Árboles** y **Grafos**, se muestran las diferencias de estas estructuras no lineales con las estudiadas anteriormente, se vinculas procesos conocidos, como la recursividad, el trabajo con punteros, la representación secuencial, enlazada, las operaciones con matrices, entre otras.

3.2.2.3 Mapa Técnicas de Diseño y Análisis de Algoritmos.

Es necesario enfatizar que en el caso de la asignatura Programación II se exige como parte del sistema de habilidades diseñar algoritmos complejos utilizando técnicas avanzadas y expresarlos con algún instrumento de descripción formal, sin embargo estas técnicas no se encuentran dentro del sistema de conocimiento de la asignatura, por otra parte se requiere que los algoritmos sean óptimos y eficientes pero no se trata dentro de la asignatura el tema de complejidad con la profundidad que este contenido requiere, como ya se ha destacado, la carrera de Informática no incluye en su diseño curricular una asignatura equivalente a la de Análisis y Diseño de Algoritmos de la carrera de Ciencias de la Computación.

El colectivo de profesores de la disciplina de Técnicas de Programación ha buscado variantes metodológicas, entre las que se encuentra el mapa conceptual Tipos Abstractos de Datos, basadas en las TICs, teniendo en cuenta los ambientes constructivistas y de enseñanza significativa, para darle solución a los vacíos cognoscitivos que se generan en el estudiante en el proceso docente educativo.

Las operaciones de las diferentes estructuras de datos que usan técnicas de diseño para lograr mayor eficiencia se vinculan a las **Técnicas de Diseño y Análisis de Algoritmos** como Divide y Vencerás, Algoritmos Voraces, Ramificación y Cotas y Vuelta Atrás (Figura 3.10). Para cada una de ellas se muestran ejemplos de algoritmos que usan estas técnicas y en cada caso se realiza el cálculo de la complejidad del mismo.

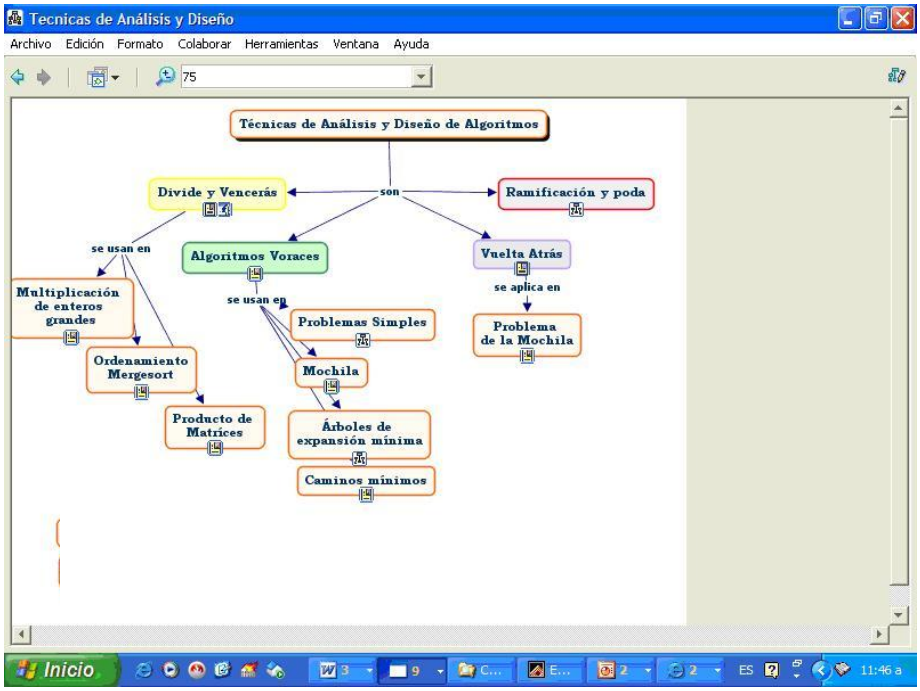

Figura 3.10 Mapa Conceptual Técnicas de Diseño y Análisis de Algoritmos.

3.2.3 Propuesta metodológica para el uso de esta herramienta.

Con este trabajo se pretende presentar un sistema bibliográfico novedoso y una herramienta de organización que muestre la esencialidad de cada

contenido y la interrelación entre ellos, entendiendo que este contenido parte de lo simple a lo complejo, teniendo en cuenta la inclusión del conocimiento.

Al proponer la utilización del Mapa Conceptual **Tipos Abstractos de Datos** en el proceso docente de la asignatura Programación II se considera pertinente formular algunas sugerencias a profesores y estudiantes.

3.2.3.1 Indicaciones para los docentes.

Se han usado regularidades para crear esquemas de organización, en cada estructura de datos se presenta la definición, el modelo matemático, las operaciones que sobre ellas se realizan, su representación gráfica, la estructura vacía, aplicaciones – que pueden incluir simulaciones o no – y, teniendo en cuenta el sentido curricular de los contenidos presentados, ejercicios y problemas resueltos. Esto permite la familiarización del estudiante con el contenido y hace que se desarrollen analogías que posibilitan, ante un nuevo problema, diseñar de forma eficiente nuevas estructuras.

Para usar de esta herramienta, el profesor puede seguir las siguientes recomendaciones:

- Imparte la clase teórico-práctica relacionada con un Tipo Abstracto de Datos, teniendo en cuenta su diseño, la representación en memoria que utiliza y las operaciones básicas, haciendo énfasis en los algoritmos básicos de las mismas y en el cálculo de la complejidad.
- Orienta la consulta del Mapa Conceptual para la estructura de datos, se les solicita que revisen los ejercicios resueltos y luego realicen los ejercicios propuestos.
- Los estudiantes tendrán permisos de acceso para adicionar en el nodo de ejercicios resueltos la solución propuesta por ellos, una vez que haya sido analizada por el grupo, lo que contribuye a enriquecer el mapa conceptual.
- El profesor abrirá espacios de discusión virtuales sobre las temáticas de mayor interés o complejidad, aprovechando las ventajas que ofrece el trabajo colaborativo. Este permiso también será concedido a estudiantes

que por su preparación en algunos temas estén capacitados para dirigir y encausar estos debates.

- Se orientarán temas a estudiar a los que no se les puede conceder el tiempo requerido teniendo en cuenta el diseño de la asignatura, como son las Técnicas de Diseño y Análisis de Algoritmos y el cálculo de la complejidad.
- Se sugerirá el análisis de complejidad de los algoritmos y la factibilidad de usar una determinada técnica de análisis y diseño para lograr eficiencia.
- Las aplicaciones resultantes de los Trabajos de Curso se incorporarán al Mapa Conceptual como simulaciones de procesos reales, es importante incluir el código fuente de los programas y el diseño del TAD, si es posible, exigir que incluyan el algoritmo en pseudocódigo.
- Se debe orientar al estudiante el uso del mapa conceptual teniendo en cuenta el nivel de conocimiento alcanzado y los objetivos vencidos.

3.2.3.2 Indicaciones para los estudiantes.

- Deben tener en cuenta el nivel de precedencia de los contenidos estudiados, al comenzar a estudiar algoritmos más complejos deben garantizar que los conceptos y procedimientos ya recibidos en asignaturas anteriores se dominen y puedan ser aplicados a los nuevos conceptos estudiados.
- Si se presentan deficiencias en la comprensión de conceptos que no constituyen objeto de estudio de la asignatura, consultar con el profesor para ubicarse dónde encontrar en el mapa la información y los ejercicios que le permitirán alcanzar la habilidad deseada.
- Al recibir un método o técnica de programación nueva, se le sugiere, revisar el mapa **Técnicas de Diseño y Análisis de Algoritmos**, el cual presenta información detallada de cada una y muestra algoritmos que las usan para resolver problemas.

- Para familiarizarse con el análisis de la complejidad de los algoritmos se les sugiere revisar la información sobre las operaciones básicas de las estructuras de datos, las cuales presentan el análisis correspondiente.
- Al diseñar una estructura que use una representación ya estudiada puede volver a analizar el mapa de la estructura que trata esta representación de forma explícita.
- Si necesita, por ejemplo, diseñar una cola de prioridades usando una representación secuencial, puede analizar los algoritmos de Inserción y Eliminación de las colas sin prioridades que usan esta representación, comprobará que los algoritmos son muy similares.
- Debe buscar la integración de los contenidos nuevos a los conocidos.

3.2.4 Aportes de la propuesta al desarrollo de la asignatura y la carrera.

La propuesta metodológica para la organización del sistema de conocimientos en la asignatura Programación II de la carrera de Informática, permite:

- Relacionar e integrar conocimientos que contribuyen a seleccionar la estructura de datos adecuada a la solución del problema.
- Llegar a conclusiones propias y construir nuevos conocimientos a partir del sistema didáctico desarrollado.
- Valorar los cambios de algoritmos, en el caso de ser necesarios, o justificar la permanencia del código y las estructuras de datos presentados, cuando consideren que no es necesario hacer modificaciones.
- Evaluar la calidad del proceso, no solo al final del mismo, sino en cada una de las etapas previas, logrando así una mayor calidad en el producto final.
- Proporcionar material bibliográfico y recursos que tratan el tema de complejidad y las Técnicas de Diseño y Análisis de Algoritmos.
- Apoyar el sistema de valores definidos por la asignatura al fomentar el criterio estético, que se logra mediante el trabajo de programación y en la presentación de los Proyectos de Curso; tomar conciencia de ahorro de recursos, de eficiencia y eficacia que se concreta en el diseño de las estructuras de datos necesarias para la solución de los problemas planteados y la elaboración de programas que hagan un uso eficiente de los recursos de un sistema de

cómputo; lo que constituye uno de los objetivos fundamentales de la asignatura Programación II de la carrera de Ingeniería Informática y el desarrollo de la responsabilidad individual y el colectivismo, que se logra al trabajar en equipo para la elaboración de los Proyectos de Curso.

3.3 Resumen del capítulo

Todos los modelos o teorías del aprendizaje dan entrada a los medios en su diseño instruccional. Por tanto, la postura del profesor ante ellos debe de ser analítica y crítica, no basta ubicarse en un paradigma explicativo del proceso instructivo que justifique la utilización de los medios, sino que deberá de investigarlos, aún cuando el marco teórico en el que se ubique condicione esa investigación.

El mapa conceptual de la asignatura Programación II integra recursos que apoyan la enseñanza significativa, incluye elementos importantes de educación en valores, contribuye a organizar el sistema de conocimientos y lo enriquece con aplicaciones y ejercicios, evidencia el vínculo horizontal y vertical de la asignatura con la carrera, permite realizar un proceso docente centrado en el alumno y en muchos casos coordinado por él, promueve el autoprendizaje y el sentido crítico, autocrítico y de eficiencia en el desarrollo de las aplicaciones resultantes del Proyecto de Curso de la asignatura, por lo que representa una herramienta de trabajo, a la vez que un sistema bibliográfico que se inserta en la organización de los contenidos tratados.

4 Conclusiones

1. El análisis del sistema de conocimientos de la disciplina Técnicas de Programación reveló los aspectos que afectan la integración de los contenidos y la creación de habilidades en el diseño de estructuras de datos y algoritmos.
2. La combinación de elementos de las tendencias pedagógicas contemporáneas y la Enseñanza Asistida por Computadoras aporta medios de enseñanza apoyados en las Nuevas Tecnologías de la Información y las Comunicaciones que contribuyen a elevar la calidad del Proceso Docente Educativo de la asignatura Programación II de la carrera de Ingeniería Informática.
3. Los temas de análisis y diseño de algoritmos son esenciales en la formación del Ingeniero Informático por su contribución al logro de habilidades de eficiencia y calidad.
4. La vinculación de componentes multimediales y de simulación de procesos contribuyó al desarrollo del mapa conceptual Tipos Abstractos de Datos que constituye un sistema bibliográfico novedoso y una herramienta de organización del sistema de conocimientos de la asignatura, que propicia la comprensión del contenido y la interrelación entre conceptos.

5 Recomendaciones

1. Continuar desarrollando el Mapa Conceptual de la asignatura Programación II hasta obtener el Modelo de Conocimiento.
2. Crear mapas conceptuales que representen los sistemas de conocimientos de cada una de las disciplinas de la carrera, mostrando la relación entre diferentes asignaturas a través de sus mapas conceptuales.

6 Bibliografía

1. ACM. (1968). Computing Curricula 1968. : The Overview Report covering undergraduate degree programs in Computer Science.
2. ACM, y IEEE-CS. (2005). Computing Curricula 2005. : The Overview Report covering undergraduate degree programs in Computer Engineering, Computer Science, Information Systems Information Technology, Software Engineering.
3. ACM, IEEE-CS, y AIS. (2004). Computing Curricula 2004. Overview Report The Association for Computing (ACM). The Association for Information Systems (AIS). The Computer Society (IEEE-CS).
4. Almeida, F., Blanco, V., y Moreno, L. (2003). EDApplets: Una Herramienta Web para la Enseñanza de Estructuras de datos y Técnicas Algorítmicas. Presentado en la Conferencia: X Jornadas de Enseñanza Universitaria de la Informática, Universidad de Laguna. Tenerife.
5. Álvarez, C. (2001). El diseño curricular en la escuela y análisis esencial del proceso curricular La Habana: Ed. Pueblo y Educación, 41 p.
6. ANTARESX. (2004). Lenguajes de Programación Orientados a Objeto: Addison-Wesley Company, 425 p.
7. Antúnez, G., Ramírez, W., Flores, A., Soler, Y., y Linares, M. (2004). Las NTI en la Educación a Distancia desde el Aula Virtual. Presentado en la Conferencia: Primer Congreso Virtual Latinoamericano de Educación a Distancia, Argentina del 23 de marzo al 4 de abril.
8. Arana, M., y Batista, N. (2000). La educación en valores: una propuesta pedagógica para la formación profesional. Pedagogía Universitaria, 4-3, 12.
9. Aspray, W. (2000). John Von Neumann and the Origins of Modern Computing. (Vol. 3) Massachusetts: Ed. The MIT Press, 24 p.
10. Ausubel, D., Novak, J. D., y Hainesian, H. (1997). Psicología Educativa. Un punto de vista cognocitivo México: Ed. Trillas, 85 p.
11. Axoft. (2005). XTango (Version 4.0). Argentina: Axoft S.A.
12. Badía, J. M. (2004). Estructuras de datos y de la información. La Habana: Ed. Pueblo y Educación, 45 p.

13. Barrios, A. M. (1997). Reflexiones epistemológicas y metodológicas en la Enseñanza de las ciencias para todos. Boletín 44, diciembre 1997 / Proyecto Principal de Educación. Disponible: http://www.unesco.cl/pdf/actyeven/ppe/boletin/artesp/44-4.pdf. [Accesado el 2/3/2002].
14. Bello, R., Gálvez, D., Lezcano, M., Lobato, A., y García, Z. (2000). Introducción a la Inteligencia Artificial. Guadalajara, México.: Ed. Pandora, 102 p.
15. BosleyGroup. (2006). MindMapper USA presents MindMapper, visual mind mapping software tool for visual thinking and writing. Priced for beginners, powered for professionals. (Version 3.10): BosleyGroup.
16. Bravo, J. (2000). Aprendizaje por descubrimiento en la enseñanza a distancia: Conceptos y un caso de estudio. La Mancha: Grupo de Informática Educativa. Departamento de Informática. Universidad de Castilla-La Mancha, 39 p.
17. Cañas, A., Granados, A., Pérez, C., Pérez, J., y Hill, G. (2003a). The network architecture of CmapTools. Pensacola, FL: Institute for Human and Machine Cognition. Technical Report. IHMC CmapTools
18. Cañas, A., y Hill, G. (2004a). Cmaptools: a knowledge modeling and sharing environment: Ed. Institute for Human & Machine Cognition. USA, 68 p.
19. Cañas, A., y Hill, G. (2004b). Cmaptools: a knowledge modeling and sharing environment Institute for Human & Machine Cognition, USA., p.
20. Cañas, A., Hill, G., y Lott, J. (2003b). Support for constructing knowledge models in CmapTools (Technical Report. HMC CmapTools). Pensacola, Fl.: Ed. Institute for Human and Machine Cognition.
21. Cañas, A., y Novak, J. (2004). Concept Maps: Theory, Methodology, Technology. Presentado en la Conferencia: First Int. Conference on Concept Mapping, Spain.
22. Cardelli, L., Donahue, J., Jordan, M., Kalsow, B., y Nelson., G. (1989). The Modula-3 type system. Presentado en la Conferencia: Sixteenth Annual ACM Symposium on Principles of Programming Languages.

23. Castillo, J., y Barberán, O. (2000). Mapas Conceptuales en Programación. Disponible: http://www.cip.es/netdidactica/articulos/mapas.htm [Accesado el 21/03/2006].
24. Copsey, B. (2005). Shared Space 2.0 wins TWO REALbasic Design Awards. Disponible: http://www.shared-space.net/ [Accesado el 11/06/2006].
25. CSO. (2005). ConceptDraw MINDMAP 4. Mind Mapping, Brainstorming and Project Planning. Disponible: http://www.conceptdraw.com/en/products/mindmap/main.php [Accesado el 23/03/2006].
26. Chambers, J. (1983). Computer-Assisted Instruction. Its Use in the classroom (Vol. 1): Ed. Prentice - Hall, 114 p.
27. Chestlevar, C. I. (2001). Reporte. Utilización de Mapas Conceptuales en la enseñanza de la programación. Bahía Blanca - Argentina: Departamento de Ciencias de la Computación. Universidad Nacional del Sur.
28. Dahl, O., Myhrhaug, B., y Nygaard, K. (1970). Simula 67 Common Base Language (2da ed.): Ed. Norwegian Computer Center, 114 p.
29. Díaz, J., y Leal, P. (2004). Ambiente Web de Apoyo al Proceso de enseñanza-Aprendizaje a través de la Representación Gráfica de Significados a modo de Mapas Conceptuales. Barcelona, 44 p.
30. Duffy, T. (1991). Constructivism: New implications for instructional technology. Educational Technology Research & Development, II, 23.
31. Estrada, V., y Febles, J. (2002). Mapas Conceptuales (Vol. III) México, 98 p.
32. GAelLimited. (2004). MindGenius - Te productivity acceleratos. Disponible: http://www.mindgenius.com [Accesado el 10/05/2006].
33. Group, O. M. (2003). Round-trip engineering for Delphi, Smalltalk, Eiffel, Java, C++, C#, Corba, VB.NET. Diagram export, report generator, multi-user: Ed. Group Object Model, 56 p.
34. Guerrero, R. (2006). Informe docente de la carrera de Informática de la Universidad de Granma. Cuarto año de experiencia. Bayamo: Universidad de Granma.
35. Guttag, J., y Liskov, B. (1986). Abstraction and Specification in Program Development. Massachusetts: Ed. The MIT Press, 23 p.

36. Heao, M. (2004). Experiencia con el uso de mapas conceptuales como estrategia de enseñanza en un curso de ingeniería del conocimiento. Sistemas Informáticos, 5 (4), 45.
37. Heileman, G. L. (2003). Estructuras de Datos, algoritmos y programación orientada a objetos. La Habana: Ed. Félix Varela, 310 p.
38. Hennessy, S. (2003). Learner perceptions of realism and magic in computer simulations. British Journal of Educational Technology, 24, 12.
39. Hodson, D. (1996). Laboratory works as scientific method: three decades of confusion and distortion. JCS Journal of curriculum studies, 28-22.
40. Holland, J., Holyoak, K., Nisbett, R., y Thagard, P. (2000). Processes of Inference, Learning, and Discovery. The Bactra Review, 57.
41. IHMC. (2005). Institute of Human and Machine Cognition. Disponible: http://cmap.ihmc.us/download/dl_CmapServer.php [Accesado el 12/2005].
42. Julian, V., y Botti, V. (2000). Agentes Inteligentes: el siguiente paso en la Inteligencia Artificial.: Ed. Addison-Wesley Company, 44 p.
43. Knuth, D. E. (1973). The art of computer programing (2da ed. Vol. 1) Stanford: Ed. Addison-Wesley Company, 634 p.
44. Koschmann, T. (2001). "The Common Lisp Companion". ISBN: 0-471-503-8-8: Ed. John Wiley & Sons., 401 p.
45. Lezcano, M. (1998). "Ambientes de aprendizaje por descubrimiento para la disciplina Inteligencia Artificial". Tesis en opción al grado de Doctor en Ciencias, Las Villas, Santa Clara.
46. Lezcano, M., Lobato, A., y Carralero, L. (2000). Prolog y los Sistemas Expertos (2da ed. Vol. 1) Guadalajara. México: Ed. Pandora., 247 p.
47. Lipschutz, S. (1977). Estructura de Datos La Habana: Ed. Revolucionaria, 390 p.
48. Lucero, M. (2004). Entre el trabajo colaborativo y el aprendizaje colaborativo. Revista Iberoamericana de Educación, XI, 23.
49. Mcarthy, J. (1995). LISP "Programming Manual Cambridge" (5ta ed.) Cambridge: Ed. MIT Press, 201 p.
50. MedioWiki. (2006). FreeMind - free mind mapping software. Disponible: http://freemind.sourceforge.net/wiki/index.php/Main_Page [Accesado el 05/2005].

51. MES. (1998a). Plan de Estudios de la Carrera Ingeniería Informática. La Habana: Ministerio de Educación Superior.
52. MES. (1998b). Programa de la disciplina de Técnicas de Programación. Comisión Nacional de la Carrera de Informática. La Habana: Ministerio de Educación Superior.
53. MES. (2003). Documento base para la elaboración de los planes de estudio (PLANES "D"). Dirección de Formación de Profesionales. La Habana: Ministerio de Educación Superior.
54. Meyer, B. (2005). Eiffel Analysis, Design and Programming Language. Presentado en la Conferencia: ECMA International. As approved as International Standard.
55. Moreira, M. A. (2002). Mapas conceptuales y aprendizaje significativo. Disponible: http://www.if.ufrgs.br/~moreira/mapasesp.pdf [Accesado el 01/2006].
56. Novak, J. (1991). Reporte. Ayudar a los alumnos a aprender cómo aprender. La opinión de un profesor-investigador en Enseñanza de las Ciencias.
57. Novak, J., y Gowin, D. (1988). Aprendiendo a aprender Barcelona: Ed. Martínez Roca, 201 p.
58. Ontoria, A. (1993). Mapas conceptuaIes: una técnica para aprender Javeriana: Ed. Narcea S.A., 78 p.
59. Ortiz, E. (2001). Concepciones teóricas y metodológicas sobre el aprendizaje. Holguín. Cuba: Ed. Instituto Superior Pedagógico "José de la Luz y Caballero", 35 p.
60. Papert, S. (1999). ¿Qué es Logo? ¿Quién lo necesita? Logo Philosophy and Implementation: Ed. Logo Computer Systems Inc. LCSI, 24 p.
61. Pérez, R. (2001). Reporte. Educación, enseñanza y curriculum. Madrid.
62. Piaget, J. (1989). La construcción de lo real en el niño. (4ta ed.) Grijalbo: Ed. Crítica, 58 p.
63. Ruiz, F. (1996). Nuevas herramientas tecnológicas para la realización de cursos por computador. Revista de Enseñanza y Tecnología, 5, 21 - 31.
64. Santillana. (2002). El mapa de conceptos y los esquemas en el aula. Disponible:

http://www.indexnet.santillana.es/mapasConceptos/mapasConceptos.htm [Accesado el 02/2006].

65. Señas, P., y Moroni, N. (2003). Reporte. Herramientas no convencionales para la enseñanza de la programación. Bahía Blanca-Argentina: Departamento de Ciencias de la Computación Instituto de Ciencias e Ingeniería de Computación. Universidad Nacional del Sur.
66. Stojanovic, L. (2002). El paradigma constructivista en el diseño de actividades y productos informáticos para ambientes de aprendizaje "on-line". Pededagogía, 23, 66.
67. TechnM. (2005). Visual Mind - eficciency througth smart thinking. Disponible: http://www.visual-mind.com/ [Accesado el 12/05/2006].
68. Uviña, P. R., Bertolami, M. A., Centeno, M. E., y Oriana, G. C. (2005). Mapas Conceptuales: una herramienta para el aprendizaje de Estructuras de Datos. Presentado en la Conferencia: JEITICS 2005 - Primeras Jornadas de Educación en Informática y TICS en Argentina, Dto. Informática-Facultad de Ingeniería-UNPSJB.
69. Vigostky, L. (1988). La Imaginación y el Arte en la Infancia. Madrid: Ed. Akal, 125 p.
70. Zarko, G. (2006). Delphi History - from Pascal to Borland Developer Studio 2006, 307 p.