UN ASISTENTE PROFESIONAL EN METODOLOGÍAS PARA LA MODERNA INGENIERÍA DE SOFTWARE

ALFONSO PÉREZ GAMA*
VICTOR HUGO MEDINA GARCÍA**
HADA JESSICA PÉREZ GUTIÉRREZ***

RESUMEN

Como resultado de un proceso de investigación iniciado en la Universidad Nacional de Colombia, continuado en la Escuela de Administración de Negocios EAN y en la Universidad Distrital F.J.C., se presenta un asistente o tutor inteligente para el aprendizaje del ciclo innovado de la metodología de ingeniería de software, a partir del modelo propuesto por James Martín, apoyado por bases de conocimientos, que están integradas así:

- Un método de planeación estratégica para el desarrollo de la informática.
- Un método de ingeniería de requerimientos y de procesos empresariales, incluyendo los métodos formales (matemática de ingeniería de software).
- Un método de ingeniería de prototipos para el diseño de software.
- Un método de ingeniería de software orientado por objetos para análisis y diseño.
- Un método de reingeniería de procesos.
- Un método de reingeniería de software que incorpora capas de conocimientos para sistemas sistemas de información gerencial.
- Un método de retroingeniería de software.

La arquitectura del sistema incluye un Asistente Virtual Inteligente, para su aprendizaje con una base de conocimientos evaluativos, un modelo de estudiante y otras herramientas.

* Ingeniero Electrónico, Universidad Distrital F.J.C.
Magister en Ingeniería de Sistemas, Universidad Nacional de Colombia.
Profesor Titular de la Universidad Nacional de Colombia, Univerisdad Distrital F.J.C. y Escuela de Administración de Negocios EAN.
E-mail: japerez@ieee.org
**Ingeniero de Sistemas, Universidad Distrital F.J.C.
Magister en Informática y Candidato a Doctorado en Lenguajes, Sistemas Informáticos en Ingeniería de Software, Universidad Politécnica de Madrid.
Director Posgrado en Ingeniería de Software, Universidad Distrital F.J.C.
Docente, Escuela de Administración de Negocios EAN.
E-Mail:
*** Núcleo EIDOS de Investigación, Universidad Nacional de Colombia.

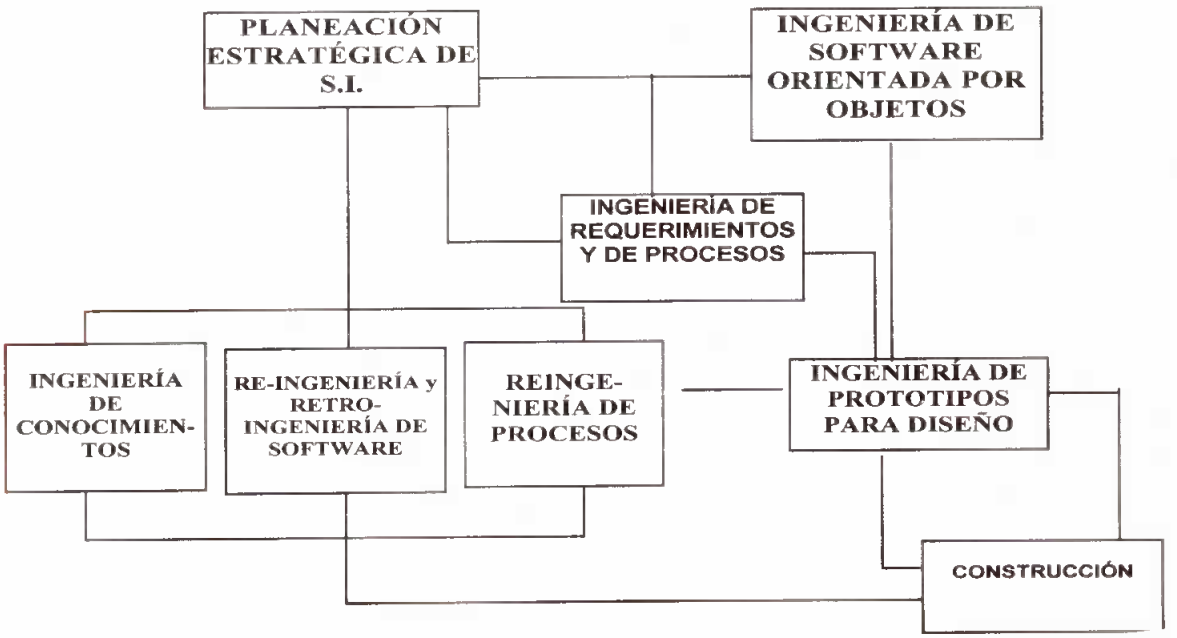
INTRODUCCIÓN

La ingeniería de software ha logrado su plena madurez: tiene su propio cuerpo de conocimientos, sus métodos de verificación y validación, su matemática en auge, tiene sus propias herramientas para construir software, sus propios estándares, ya es considerada como una profesión independiente con sus propios programas académicos en organizaciones internacionales como IEEE y la ACM quienes adoptaron un código de ética como un instrumento que orienta a sus practicantes a ejercer esta profesión con responsabilidad civil y social ante la comunidad. Al cabo de un tercio de siglo, cuando comenzó a acuñarse el término de ingeniería de software, la IS se apartó de las ciencias de la computación: La ingeniería de software ya no son simples programas de computador. Los ingenieros de software ya no son buenos programadores simplemente. La comunidad académica ya había entendido que la metodología de la ingeniería de software estaba evolucionando y que los métodos de las ciencias de la computación resultaban insuficientes. Es más, se ha observado desde décadas anteriores la convergencia de la ingeniería de software con la Inteligencia artificial o mejor aún la mismísima ingeniería de conocimientos.

Antecedentes

Nuestra investigación se inició en el Núcleo EIDOS de la Universidad Nacional a finales de los 80's, apoyado por Colciencias y RIBIE a instancias de la organización del V Centenario del Gobierno. Construimos varios prototipos para lograr el know how en cómo hacer reingeniería de software, cómo era una base de datos orientada por objetos, una base inteligente de datos, entre muchos otros. Posteriormente en la Universidad Militar y en la Escuela de Administración de Negocios EAN (Centro de Investigaciones) se trabajó en el diseño de un tutor virtual para la verificación de conceptos, en la vía de obtener un asistente profesional en la moderna ingeniería de software. Hoy continuamos en el Centro de Estudios e Investigación de la ACCIO y en la Universidad Nacional contamos con una base de conocimientos que incluyen entre otros los componentes metodológicos exhibidos en la Gráfica No. 1. Recientemente presentamos al Ministerio de Comunicaciones y a Colciencias un proyecto en grande escala para construir, implementar e instrumentar herramientas para aprendizaje autónomo en el contexto de educación virtual, entre ellas un asistente profesional inteligente en ingeniería de software. A este respecto hemos desarrollado varios prototipos.

Gráfica No.1
Ciclo Innovado de la Ingeniería de Software



Oferta Insuficiente de Profesionales de Software

Algo que plantea serios interrogantes a la comunidad académica es que el número de profesionales de ingeniería de software, son y serán insuficientes para abastecer la demanda en países avanzados. Es así que con motivo de Y2K y del lanzamiento de Euro el Doctor Casper Jones (1999); estimó en cerca de 700.000 profesionales que hicieron falta para estos proyectos cruciales en el año anterior. Recientemente J. Edwards (2000), basado en un estudio del IDC amplifica las estadísticas un déficit progresivo que puede superar el 1'000.000. Lo anterior debe alarmar a la comunidad académica, puesto que además se ha puesto en evidencia que algunas multinacionales de la informática están creando sus propios programas académicos de posgrado y re-entrenamiento.

EL PARADIGMA DE ORIENTACIÓN POR OBJETOS

Hace más de tres décadas los grandes del software con N. Wirth, D. Parnas y T. Hoare, entre muchos otros habían establecido la necesidad de definir el modelo matemático de un dato: los tipos abstractos de datos y los objetos abstractos. los primeros establecían <valor, operaciones> para el dato: qué clase de valores puede almacenar y además que operaciones les era permitido, lo cual sin duda alguna determinaba la validación de los datos a tiempo de compilación, con lenguajes como MODULA 2. El objeto además de aglutinar los tipos en un todo, con un identificador, predefinía la conducta de los datos: con qué objetos se puede comunicar, qué servicios puede ofrecer y además que servicios puede requerir de otros objetos. Lo anterior marcó un hito muy importante en términos de reducir la complejidad en la construcción de software donde millares de líneas de código se reducían a algunas decenas; además el concepto de objeto es muy intuitivo para la persona humana. Por otra parte el posibilitar el manejo del polimorfismo, herencia de propiedades y extensibilidades le da a la ingeniería un poder muy alto de lograr las metas de software cero defectos y de predecir ex ante la calidad del software a lograr.

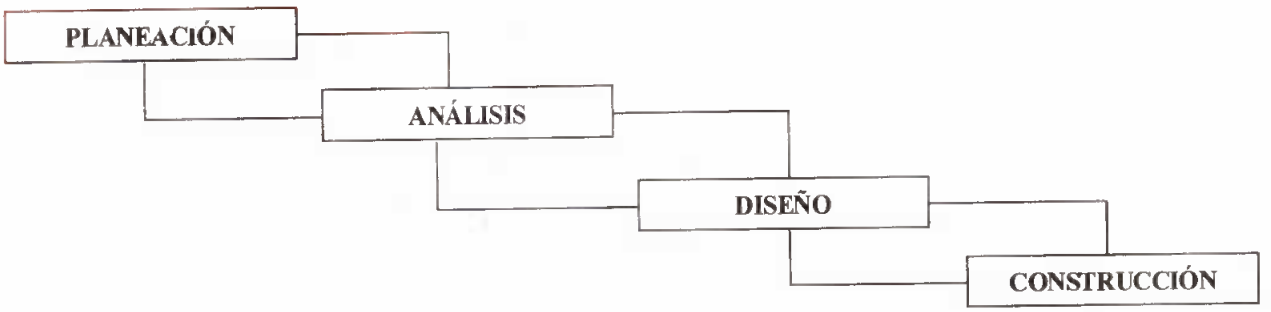
EL CICLO MULTIDIRECCIONAL Y DINÁMICO DE LA IS

La evolución hacia la moderna ingeniería de software distingue cuatro etapas fundamentales: Planeación, análisis, diseño y construcción, propuestos en el modelo de ingeniería de información de J. Martín.

El concepto de PLANEACIÓN no se debe confundir con la concepción simplista y mecanicista de hacer un diagrama de Gantt ni un diagrama de PERT o CPM. Planeación es la sinónima de conducir el cambio en la organización; es una forma de gerenciar un sistema, no es una mecánica, ni un conjunto de técnicas para formular programas de trabajo. La planeación estratégica del sistema o PESI, como concepto es el análisis de la problemática empresarial desde el punto de vista netamente de la actividad de la institución en la cual se está desarrollando el sistema de software. Su fin primordial es generar un plan de sistemas -solución -, orientados por estrategias particulares que estén debidamente alineadas con el plan corporativo y que constituya una solución empresarial. El PESI exige un conocimiento importante de la empresa, en términos de una misión, visión, procesos, datos, información, conocimiento, decisiones, comunicaciones, problemas, personal y especialmente el ubicar a la empresa en el tiempo y espacio tanto interno como externo lo cual nos dice de las potencialidades y peligros sobre la organización al igual que nos hace reflexionar en qué somos fuertes y en qué no, para enfretar la competencia y especialmente para la supervivencia de la organización a largo plazo. Modernamente el PESI se apoya en escenarios a partir de la visión en el horizonte de la planeación para inferir de la ingeniería de requerimientos y de las habilidades humanas y empresariales que debe desarrollar la organización para alcanzar los objetivos previamente cuantificados. La moderna PESI puede hacer uso de varias decenas de diferentes clases de técnicas y herramientas, muchas de ellas desconocidas en el país.

Cuando se pasa al ANÁLISIS o mejor la ingeniería de requerimientos, se está considerando la fase de ingeniería de las necesidades donde comienza a hacerse el estudio multidisciplinar de los requerimientos generales y específicos tales como solución de problemas, decisiones, información,

GRÁFICA No.2
Ciclo de la Ingeniería de Software según J. Martín



conocimientos, interfaces hombre-máquina-sistemas, comunicaciones, operaciones, seguridad, protección, privacidad, entre muchos otros. Así mismo la ingeniería de requerimientos cuenta con formalismos matemáticos para documentar estos requerimientos y las especificaciones de diseño. Esta matemática posibilita a los ingenieros de software identificar ex ante su construcción, inconsistencias en las especificaciones y demostrar que el software alcanza los objetivos propuestos entre otros.

La fase de DISEÑO debe fincarse en la creatividad y la innovación como uno de los requerimientos necesarios para generar un espacio de soluciones concretadas en especificaciones estructurales, funcionales, interfaciales, cognoscitivas, sistemas y relacionadas. Las especificaciones de diseño debidamente formalizadas posibilitan la generación de código mediante instrumentos tales como UN-PROLOG.

La fase de CONSTRUCCIÓN es un proceso tecnológico altamente automatizado para la generación de los programas y su correspondiente implementación, lo cual se logra en un 90% del total de software requerido quedando un remanente para afinamiento de detalles. La construcción apoyada por las herramientas tecnológicas modernas ya mencionadas de IV y V generación exige un trabajo disciplinado aunque menos complicado, en que la parte de programación está altamente automatizada, adelgazando el tamaño del proyecto de software pero enriqueciéndolo con tecnología: CASE, herramientas de productividad y de gestión de conocimientos. Por imperativos tecnológicos y de las mismas necesidades empresariales fue necesario introducir técnicas ingenieriles para evolucionar e inteligenciar el software sin cambiar las especificaciones funcionales (reingeniería y además recuperar documentación, especificaciones y modelos de datos, físicos, etc.), a partir del código (retroingeniería). Como se podrá apreciar, el ciclo de vida del software evolucionó hacia lo multidireccional en contraposición a los otros métodos tradicionales de las ciencias de la computación. Más concretamente se puede emplear la ingeniería en forma directa, en dirección opuesta (retroingeniería, i.e. patir del código para recuperar las especificaciones de diseño o los requerimientos) o trabajar sobre un espacio del ciclo con fines de optimización (reingeniería de software, e.g. redefinir estructuras de información o archivos sin alterar la funcionalidad del sistema o modificar el modelo de datos).

INGENIERÍA Y REINGENIERÍA DE PROCESOS EMPRESARIALES

En esta parte se contempla el desarrollo de conceptos generales, enfoques, motivos que impulsan a las empresas al proceso de ingeniería y reingeniería, requisitos para enprender proyectos de ingeniería de información y de procesos y en general todos los elementos que se deben tener en cuenta en este proceso de cambio con el fin de dar respuesta a las presiones del mercado actual enmarcado en un proceso de globalización e innovación de productos y servicios para satis-

facer necesidades de los clientes quienes conforman segmentos cada vez más específicos.

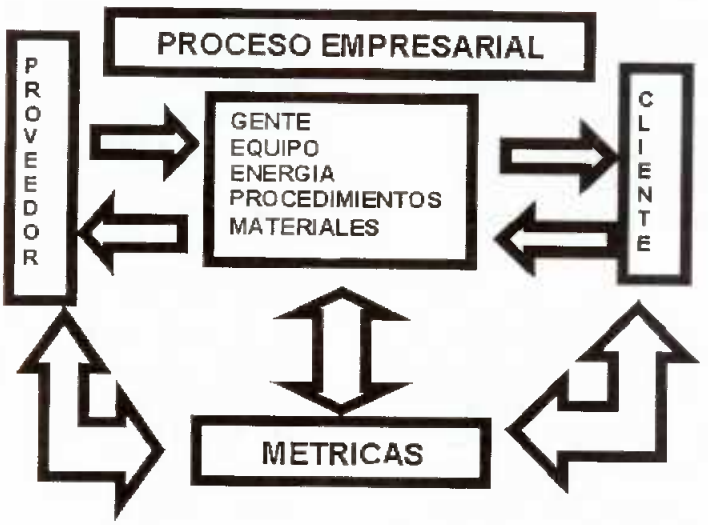
La ingeniería de procesos dispone hoy en día de herramientas matemáticas (modelo interindustrial de Leontief, Álgebras de Procesos) e instrumentos gráficos (e.g. BAM, Ishikagua). Esta ingeniería de procesos como es característico de toda la ingeniería de software es altamente intensiva en tecnología necesaria y suficiente para que la empresa sobreviva manteniendo sus ventajas competitivas. El proceso puede verse como la esencia del negocio. No sólo la mayor parte del trabajo se hace a través de procesos, sino que gran parte de los aspectos que en realidad diferencian a las compañías entre si, es inherente a su proceso particular de trabajo. Un proceso se define como una actividad que se lleva a cabo en una serie de etapas para producir un resultado específico o un grupo coherente de resultados específicos. Es un conjunto de actividades relacionadas que transforman unas entradas en productos o servicios con valor para el cliente. Un proceso empresarial es una secuencia de actividades repetitivas caracterizadas por tener entradas y salidas con sus metas y actividades con valor agregado; comprende organización, personal, equipo, procedimientos y material de trabajo requerido para producir un resultado final específico (producto). Los procesos, y no las organizaciones en sí, ni tampoco las funciones, son el objeto de la reingeniería.

REINGENIERÍA DE SOFTWARE

Evidentemente se trata de utilizar las nuevas tecnologías de conocimientos y de la inteligencia que posibilitan la migración de un sistema convencional de información a un SIIG (Sistema Inteligente de Información Gerencial) lo que requiere de la aplicación de ingeniería de sistemas de información en dos frentes: la reingeniería de software y la ingeniería de conocimientos. La aplicación de reingeniería se puede visualizar de la siguiente manera:

*Sea Rj una representación del SIIG; la reingeniería constituye la re-creación de una parte de sistema en una nueva forma y su implementación R*j.*

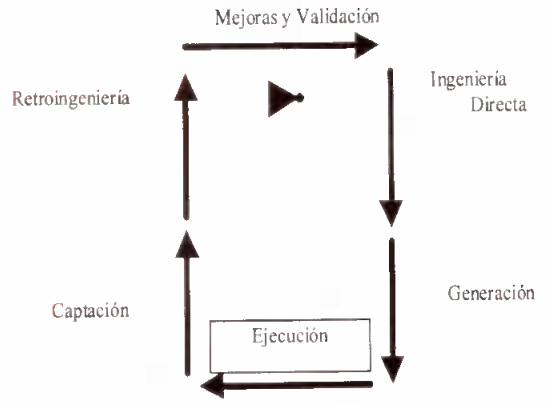
GRÁFICA No. 3
EL PROCESO



Es decir la renovación de los mecanismos internos, la modernización de las estructuras de datos y su arquitectura , el redesarrollo del SI para mejorar su desempeño, la renovación y la recontextualización de su estructura y evidentemente su implementación para mejorar su desempeño conservando sus características funcionales básicas. Es decir la moderna ingeniería de software es multidireccional (claramente distingue la ingeniería directa y la retroingeniería) cuya verticalidad es fruto de intensa investigación y la transversabilidad que posibilita la optimización por capas o niveles de la empresa. Andrews define esta reingeniería como una tecnología de valor agregado. Charles W. Bachman (89) definió una

metodología la cual ha enriquecido indudablemente el quehacer ingenieril cuyo ciclo completo de reingeniería y retroingeniería se muestra en la Gráfica No. 4.

GRÁFICA No. 4
CICLO DINÁMICO DE BACHMAN



INGENIERÍA DE CONOCIMIENTOS

La aplicación de la ingeniería de conocimientos se puede entender como el proceso de inteligenciación de un sistema o la reingenierización para darle inteligencia al negocio representando los conocimientos para razonar. Es como la integración sistémica de capas o metasistemas de conocimeintos a la arquitectura del sistema para lograr un desempeño más acorde con las exigencias actuales, que supere las limitaciones analíticas, cuantitativas y cualitativas de los datos y se ofrezcan nuevos espacios de pensamiento y solución oportuna de problemas. En estas capas se persigue la representación de problemas del sistema, la representación e información sobre el usuario y su estilo cognitivo, el sistema de razonamiento decisional, la inteligencia en la base de datos, entre otros.

En este cometido se integran varias capas de conocimientos donde conjugan la representación de conocimientos de los diferentes agentes que actúan en el escenario de un sistema de información y los modos de razonamiento, tales como el conocimiento experto en temas gerenciales, conocimiento del usuario o ejecutivo del SI y su representación computacional, las herramientas para resolver problemas, el conocimiento y razonamiento sobre los datos y la tecnología inteligente para apoyar la gestión y la planeación. Del grado de aplicación de dichas tecnologías dependerá el grado de conocimiento experto que procese el sistema y por ende se logrará un espacio para la solución continua de problemas de diferente índole.

El equipo integrado por investigadores del proyecto EIDOS hizo las pruebas de campo de la metodología para la incorporación de capas de conocimientos a un sistema, que consiste fundamentalmente en dos eventos donde se hace clara distinción del deber-ser del sistema actual y la realidad del mismo:

a. La recuperación del sistema actual incluyendo la captación de la documentación y,

b. El mismo proceso de reingeniería.

El primer evento se concentra desde el comienzo y se entiende que la captación de dicho conocimiento es marginalmente decreciente a lo largo de toda la metodología. El segundo evento integrado por actividades tales como la recomposición, la reconstrucción, la misma conversión acompañados de un sistema de monitoreo y seguimiento, incluye una actividad intensa de ingeniería de conocimientos pari passu la recuperación del sistema y la propia reingeniería. Un sistema de ingeniería de costos constituye una variable de decisión importante. Esta metodología incluye una discusión sobre el perfil del recurso humano que debe integrar un equipo de reingeniería.

RETROINGENIERÍA DE SOFTWARE

La retroingeniería es el proceso de analizar un sistema, para identificar componentes del sistema y sus interrelaciones y crear representaciones del sistema en otra forma o en un nivel de mayor abstracción. La retroingeniería dentro y fuera de si misma no implica cambiar el sistema o crear un sistema nuevo, solo es un proceso de inteligencia. Su objetivo principal es incrementar el aprendizaje y la comprensión total del sistema para mantenimiento y un nuevo desarrollo. La retroingeniería tiene otros objetivos importantes como son:

✍ Disminuir la complejidad. Es la reducción del volumen, la simplificación del número de variables y restricciones y la complejidad de los sistemas por medio de ciertos métodos, como el soporte automatizado. Los métodos y herramientas de la retroingeniería combinados con ambientes CASE, proveen un camino para extraer información importante, para que los fabricantes puedan controlar el proceso y el producto en la evolución del sistema.

✍ Producir vistas alternativas. Es la generación o regeneración de representaciones gráficas como ayudas de comprensión. Sin embargo la creación y el mantenimiento puede llevar a un embotellamiento.

✍ Recuperar información perdida. Es la recuperación de cualquier sistema existente, a partir de la recuperación del diseño, además deja obtener un manual de los sistemas para entender lo que se ha hecho o cómo sus programas individuales interactúan como un sistema.

✍ Detectar efectos secundarios. Es encontrar ramificaciones no deseadas o efectos secundarios, que aparecen en el diseño y las modificaciones sucesivas, que impiden una adecuada ejecución del sistema.

Cabe observar que el uso de las herramientas CASE cuyo empleo es prácticamente indispensable en el proceso de reingeniería y retroingeniería. Dentro del proyecto EIDOS se contó con IDMS-ARCHITECT para el análisis y diseño de sistemas de información y el ADW/PWS de knowledge Ware (hoy Sternling) para la planeación estratégica de un SIG. Entre los lenguajes de V generación contamos con UNPROLOG con todas sus extensiones y en especial su característica de transportabilidad y poder expresivo. Hay varios Shells para trabajar las capas de conocimientos. Uno de ellos es NEXPERT OBJECT, herramienta de ingeniería de conocimientos adquirida para el proyecto EIDOS con el cual se ha trabajado.

INGENIERÍA DE PROTOTIPOS

El ciclo de vida de desarrollo de software presenta diferentes modelos para desarrollo de software entre los que se tiene: modelo lineal secuencial, modelo de construcción de prototipos y modelo RAD. Adicionalmente se encuentran los modelos de procesos evolutivos de software como: modelo incremental, modelo en espiral, modelo de ensamblaje de componentes y modelo de desarrollo concurrente. El prototipo es un proceso que permite al desarrollador crear un modelo de software que tiene que ser construido rápidamente. Es una aplicación que funciona y facilita la comunicación con el usuario final. Se presenta como una alternativa del ciclo de vida del desarrollo de sistemas. El prototipo puede contribuir a estabilizar los requerimientos del nuevo sistema y las mejoras al sistema actual, ya que un prototipo debe contener las características esenciales de un sistema sea uno nuevo o uno actual.

A manera de síntesis y para emplear un lenguaje universal, siguiendo a Kruchten [1984] y de acuerdo con Lewis et. al. [1989], un ***prototipo P*** es un conjunto de objetos del interfaz del usuario U, un conjunto de acciones A y una función de transformación F, la cual se puede expresar como:

$$P = \{U,A,F\}$$

F: Especifica una clase de computaciones, con datos de entrada, actuando como parámetros en el cual el diseño asegura que los aspectos principales y/o críticos del sistema, actúan de acuerdo con las especificaciones.

A: El conjunto de acciones se ilustra mejor, dentro del modelo de ciclo evolutivo e incrementalista del software.

U: Representa las necesidades y herramientas tales como editores gráficos, depuradores, (para mejorar y corregir pantallas, menús, diálogos), exploradores del sistema que permitan interactuar e.g. software de la base de datos (para examinar y recuperar posibles componentes reutilizables de software), grabar las pruebas con casos exitosos, para que al final se pueda escoger mejor el escenario para corridas de las demostraciones del prototipo. (Luqi, 89), (Luqi y Ketabchi, 88).

El desarrollo evolutivo y progresivo de prototipos de aplicaciones es una actividad que consiste en crear un modelo que trabaje para un sistema de información. Un prototipo contiene todas las características esenciales en el sistema. Por lo tanto, en esta parte de la investigación se consideró la fase de Prototipos como un modelo para seguir en el desarrollo de un software dentro del ciclo de vida del software para un proyecto específico y que vaya de la mano con los objetivos y metas de una organización. Boehm (88). Entre las consideraciones están:

- Generar una base de conocimientos sobre los prototipos, desarrollando temas como las clases de prototipos, los prototipos como una alternativa del ciclo de vida del software, métodos para el desarrollo de prototipos, herramientas y una metodología para el desarrollo rápido de aplicaciones.

- Explicar a los usuarios finales por qué se pueden beneficiar con el uso de un prototipo y cuál es su papel en su evolución.

- Decidir si debe emplearse un método de desarrollo de prototipos para una situación particular.

- Seleccionar las herramientas más adecuadas para preparar un prototipo específico.

- Planificar la secuencia de actividades para construir un prototipo de aplicación.

- Seleccionar una técnica creación de prototipos que sea la más costeable para lograr el objetivo de un producto eficiente.

- Para decidir si se debe realizar un prototipo se tiene en cuenta dentro de una las fases del ciclo de vida del desarrollo de sistemas preferiblemente en la fase de ingeniería de los requerimientos.

- Explicar los diferentes tipos de prototipos para la construcción de un modelo.

- Explicar las diferentes herramientas con las que se pueden desarrollar los prototipos y el porqué de su utilización.

- Mostrar el desarrollo rápido de aplicaciones como una metodología para el desarrollo de software lineal y secuencial que se apoya en un ciclo de desarrollo extremadamente corto pero evolucionando.

- Explicar a los usuarios finales las ventajas y desventajas de la utilización de prototipos dentro del desarrollo del ciclo de vida del software.

TUTOR VIRTUAL EN EL ESPACIO DE LA EDUMÁTICA INTELIGENTE

Este componente estructura la parte de gestión inteligente para guiar el aprendizaje autónomo del usuario. El sistema debe entre otros acceder a la base, problemas y pruebas, gestionar la evaluación y deducir su progreso. Debe dialogar con el usuario sobre las respuestas buenas, malas o no contestadas. En caso que el estudiante lo requiera, se debe desplegar la explicación registrada por el administrador.

Además debe hacer los cálculos estadísticos y demás comprobaciones. Por su parte el dominio representado en un mapa conceptual, le permite al sistema identificar los conocimientos en forma de una red semántica.

Complementariamente cuenta con un evaluador cognoscitivo del usuario para verificación de conceptos, controlado por el mismo tutor.

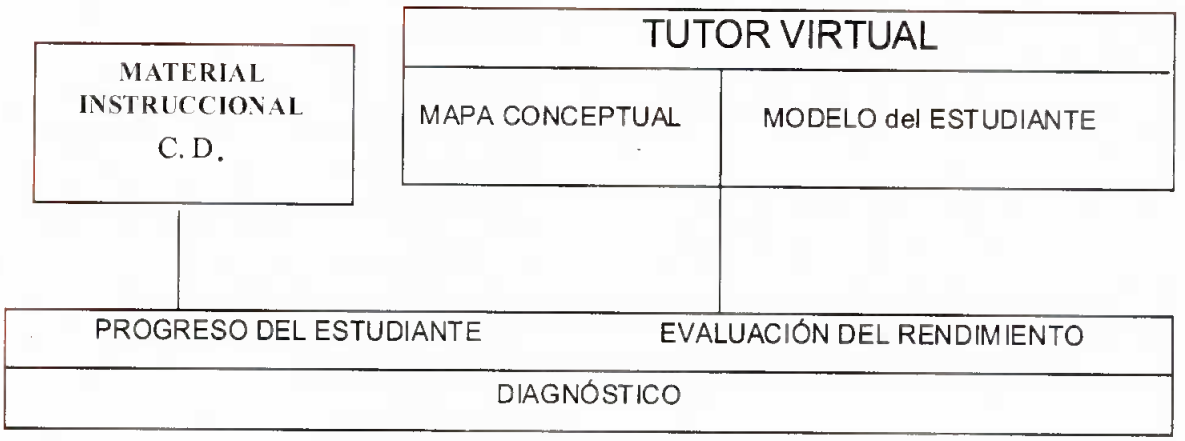
El modelo del estudiante es un constructo informático que representa el conocimiento actual del usuario. El planificador posibilita la generación de un plan para remediación y progreso del estudiante. El sistema se concibió para trabajar sobre internet. El esquema operacional se ilustra en la Gráfica No.6.

CONCLUSIONES

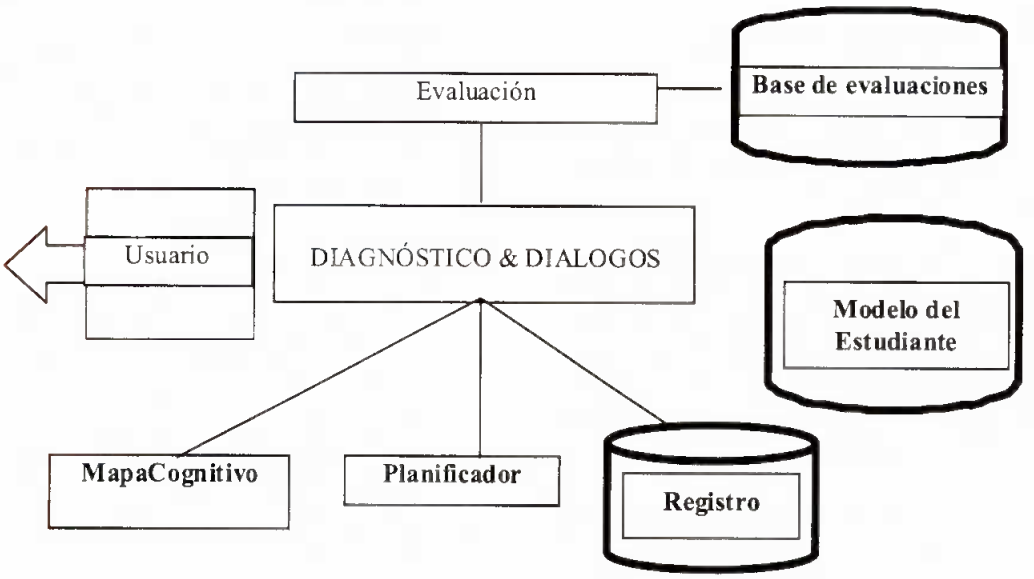
El Asistente Inteligente presentado para el aprendizaje de estas metodologías en el contexto de la moderna ingeniería de software es una herramienta apoyada por un modelo pedagógico computacional discutido en Pérez Gama (96 y 99) cuyo propósito final es el desarrollo de habilidades profesionales para la construcción de software de buena calidad según las exigencias de la sociedad del conocimiento.

Valga anotar que en Colombia el mercado de tecnologías de conocimientos y de la inteligenca es

GRÁFICA No. 5
ESQUEMA



GRÁFICA No.6
COMPONENTES DEL TUTOR VIRTUAL



incipiente y muchos distribuidores de software consideran que dicho mercado es costoso, con poca demanda y por consiguiente de alto riesgo, dándose a translucir que no existe un interés mínimo a este respecto. Los nuevos paradigmas de la reingeniería y retroingeniería de software con el apoyo de herramientas CASE, no han sido bien entendidas en Colombia como posibilidades para innovar los sistemas de información.

Nuestro planteamiento, centrado en la INNOVACIÓN y RE-CREACIÓN mediatizada por la reflexión intensa tanto a nivel teórico como conceptual y sancionada por la experimentación construyendo gran variedad de prototipos nos ha posibilitado el conocimiento que se ha dado en el espacio de nuestra investigación dentro de una perspectiva de doble uso: en el terreno de la ingeniería de software profesional y para su empleo en educación haciendo utilización intensa de herramientas CASE y de instrumentos de ingeniería de conocimientos potenciada por el quehacer diario desde nuestra cátedra universitaria.

Finalmente hemos enfatizado sobre la aplicación de la inteligencia artificial y de la ingeniería de software en investigación sobre nuevas arquitecturas. Seguimos investigando sobre la aplicación de técnicas de Aprendizaje Maquinal y descubrimiento de conocimiento a partir de bases de datos, para recontextualizar el problema del aprendizaje humano en ambientes electrónicos.

BIBLIOGRAFÍA

BACHMAN CHARLES (89): «A personal chronicle: Creating Better Information Systems, with Some Guiding Principles» IEEE Transactions on Knowledge and Data Engineering. Vol.1, #1 de Marzo de 1989.

BLANNING (93) R. & D. King: «Current Research in Decision Support Technology» IEEE Computer Society 1993.

BOHEM Barry (88) "A Spiral Model of Software Development and Enhancement" IEEE Computer May 1988.

EDWARDS (2000) John, "Redefining IT Career Paths for the new Millennium", IEEE Computer January 2000.

JONES(99) Casper, " The Euro, Y2K, and the US Software Labor Shortage", IEEE Software May/ June 1999.

KRUCHTE N Phillipe, Ed. Schomberg, Jacob Schwartz (1984): "Software Prototyping Using SETL Programming Language" IEEE Software Vol 1 #4.

LEWIS (89) T.G., Fred Handloser III, S Bose, S. Yang: "Prototypes from Standard User Interfaces Management Systems", IEEE Computer Vol 22, No. 5.

LUQI & Mohammed Ketabchi (1988):"A Computer Aided Prototyping Systems", IEEE Software Vol 5 # 2.

LUQI (1989): "Software Evolution through Rapid Prototyping" IEEE Computer Vol 22, #5.

PÉREZ GAMA Alfonso (98) "La investigacion en informática avanzada: de la utopía a la realidad" Revista Sistemas de ACIS Enero - Marzo de 1998 Bogotá.

PEREZ GAMA Alfonso (96) -"De la reelaboracion del conocimiento a la generacion de conocimiento nuevo mediante edumatica e inteligencia artificial" XV Reunión Nacional de ACOFI -CARTAGENA 1996.

PÉREZ GAMA Alfonso (94) HADA JESSICA PÉREZ. JAVIER GARCÍA Y RAÚL ALEXANDER ALONSO. "Reingeniería de Software con el Apoyo de Conocimientos: UN-METOD-REING" I Congreso Nacional de Reingeniería - Santa Fe de Bogotá Dic 1994 - ReEngineering Corporation.

PÉREZ GAMA Alfonso (99) "La Universidad Virtual en la Sociedad del Conocimiento" Revista INGENIERIA, No. 2 1999 Universidad Distrital Francisco José de Caldas.

PEREZ GAMA Alfonso (90), VICTOR HUGO MEDINA GARCIA "Desarrollo de sistemas y prototipos con apoyo de computador CASE –Universidad Nacional 1990.

PEREZ GAMA Alfonso (99) Informe de Investigación, sobre METODOLOGÍAS EXPERTAS PARA DESARROLLO DE SOFTWARE; Centro de Investigaciones EAN.