UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

POSGRADO EN CIENCIAS E INGENIERIA
DE LA COMPUTACION

"DE LAS HERRAMIENTAS CASE A LAS HERRAMIENTAS DE APOYO A PROCESOS DE DESARROLLO DE SOFTWARE"

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRA EN CIENCIAS

P R E S E N T A:

MA. DE JESUS CASAS VALADEZ

DIRECTORA DE TESIS:
M. EN C. MA. GUADALUPE E. IBARGUENGOITIA G.

CD. DE MEXICO; SEPTIEMBRE DEL 2001

A mi madre ausente:

Con todo cariño y admiración.

A Jaime:

Mi maestro, amigo, socio, cómplice y esposo,
sin cuya ayuda hubiese sido imposible esta empresa.
Gracias por darme tu tiempo,
te adoro.

A Horacio:

Mi patrón, jefe, tiempo e hijo.
Te quiero mi vida linda.

A mi padre y hermanas:

Por ese afán de salir adelante siempre.

A doña Anita y familia:

Por su disposición para ayudarnos en todo momento.

AGRADECIMIENTOS

Todo trabajo individual es el resultado de una serie de apoyos, brindados muchas veces por personas anónimas, a quienes doy las gracias. Ahora, considero necesario mencionar a quienes contribuyeron con su experiencia y sabiduría para el logro de esta investigación.

A mi directora de tesis, la M. en C. Guadalupe Ibargüengoitia, a quien agradezco sus conocimientos, disposición y tiempo para la culminación de la presente, y, sobre todo, por proporcionarme con su trabajo y personalidad, la directriz profesional que no existía en mi carrera. Maestra Lupita, muchísimas gracias.

A la Dra. Hanna Oktaba, mi admiración y respeto. Y toda mi gratitud por el espacio dedicado a este proyecto. Gracias Doctora.

A la Dra. Begoña Albizuri, a quien sin tener el gusto de conocerla, aceptó, sin objeciones, ser parte del grupo de lectores. Recibiendo la grata sorpresa de encontrar a una profesionista admirable, como las anteriormente mencionadas. Gracias Begoña, por tu tiempo y tus observaciones.

Al Dr. Fernando Gamboa, quien revisó en detalle el borrador de esta tesis. Agradezco su tiempo, dedicación y paciencia.

Al Maestro Gustavo Márquez, su disposición y consejos para la realización de la misma.

A mis compañeros de la Maestría, por el empeño mostrado a pesar de las adversidades. Gracias amigos.

Finalmente, expreso mi gratitud a la UNAM, al CONACYT y al personal académico y administrativo del IIMAS, quienes facilitaron la culminación de esta labor.

Ma. de Jesús Casas Valadez

Cd. Universitaria

México, D. F., septiembre del 2001.

ÍNDICE

INTRODUCCIÓN

Antes de que se determinara el tema principal de este trabajo, se tenía en mente obtener una visión global del uso de las herramientas en la Ingeniería de Software. Al iniciar la recopilación se localizó una cantidad considerable de información con respecto a las herramientas CASE (Ingeniería de Software Asistida por Computadora); escasas referencias a las herramientas I-CASE (Integración de Herramientas CASE), y sólo dos o tres documentos enfocados al análisis de los entornos PSEE (Ambientes de Ingeniería de Software Centrados en el Proceso).

Esta inspección bibliográfica hizo pensar en un primer momento que después del I-CASE, el panorama para apoyar a la ingeniería de software con nuevas herramientas, había llegado a su fin. Sin embargo, después de revisar otros documentos, se encontró al área en pleno auge.

Con estas premisas, el primer objetivo de mi investigación es hacer un recorrido teórico por los conceptos ligados al desarrollo de software auxiliado por computadora. Un segundo objetivo consiste en revisar algunas de las características de estos tipos de herramientas, para analizarlos en la práctica. A continuación se presenta un resumen de los apartados que forman este trabajo:

En el primer capítulo titulado "CASE", se mencionan sus componentes y se hace una revisión histórica por su desarrollo.

En el segundo capítulo, correspondiente al I-CASE, se reseña la definición del término, sus componentes y características, enfocándose principalmente en el depósito, elemento que lo caracteriza.

En el tercer capítulo se parte de la definición de los PSEE, donde se incluye, entre otros aspectos, sus características, arquitectura, áreas de aplicación y estado del arte.

Finalmente, en el cuarto capítulo, titulado "Revisión de características teóricas en dos ambientes comerciales", se analizan y comentan algunos de los aspectos mencionados en secciones anteriores, evaluándolos en dos de los productos disponibles en el mercado mexicano: Rational Unified Process (RUP), de la compañía Rational, y Norma Control, de la empresa Jónima.

CAPÍTULO 1

CASE

1. Computer-Aided/Assisted/Automated Software/System Engineering

1.1. Definición

Al jefe del Instituto de Ingeniería de Software de la Universidad Carnegie Mellon, J. Manley, se le atribuye la creación —en 1981— del acrónimo CASE,[1] cuyo significado es "Computer-Aided/Assisted/Automated Software/System Engineering", que en español se traduce como "Ingeniería de Software asistida por computadora".

Datos precisos como los anteriores (los del autor y la fecha), contrastan con la información que se tiene con respecto al momento en el que aparecieron las herramientas CASE. Por ejemplo, en el artículo de Camilo Ocampo (et al),[2] se afirma que el surgimiento de las herramientas CASE ocurrió a principio de los ochenta con el uso de los diagramas de flujo (DFD), los diagramas de entidad-relación (ERD) y las gráficas estructuradas —herramientas gráficas utilizadas en el análisis y el diseño de sistemas. Mientras tanto,

---

[1] Camilo Ocampo, Begoña Albizuri y Pere Botella, "De la Tecnología CASE a la Tecnología de procesos" en Soluciones Avanzadas, Enero 1999, núm. 65, p.52.
[2] Ibid., p. 51.

Fuggeta[3] sostiene que las herramientas CASE existen desde los sesenta, con la creación del ensamblador para brindar apoyo en el desarrollo de programas de software.

Como el CASE fue evolucionando con el tiempo, las definiciones fueron cambiando. Al mismo término CASE se le han agregado los significados de "Aided/Assisted/Automated" (apoyo, asistencia y automatización) ya que distintas definiciones así lo especifican. De acuerdo con varios autores:

a) CASE "abarca una colección de herramientas y métodos automatizados que asisten a la ingeniería de software en las fases del ciclo de vida del desarrollo de software".[4]

b) CASE "comprende las herramientas individuales que ayudan a los desarrolladores de software y/o administradores de proyectos durante una o más fases del desarrollo (o mantenimiento) del software".[5]

c) CASE "es una combinación de herramientas de software y metodologías estructuradas para apoyar el desarrollo de software".[6]

d) CASE "es el término utilizado para designar el apoyo proporcionado por las herramientas de software a los procesos de la ingeniería de software".[7]

e) CASE es "cualquier tecnología que ayuda en el proceso de software".[8]

Pero Fuggeta y Sommerville resumen el concepto como sigue:

> "CASE comprende el apoyo proporcionado por las herramientas de software a los procesos de la ingeniería de software".[9]

---

[3] Alfonso Fuggetta,, "A classification of CASE Technology", en Pankaj K. Garg y Mehdi Jazayeri (eds.), Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Los Alamitos, CA, 1996 [Computer, Diciembre 1993], p. 294.
[4] J. Sodhi, Software Eng.: Methods, Management, and CASE Tools, McGraw-Hill, Blue Ridge Summit, Pa, 1991, tomado de Alfonso Fuggetta, ibid., p. 290.
[5] B. Terry and D. Logee, "Terminology for Software Engineering and Computer-aided Software Engineering" en Software Engineering Notes, abril, 1990.
[6] Carma McClure, "CASE Experience" en BYTE, abril, 89, p.235.
[7] I. Sommerville, Software Eng. Addison-Wesley, Reading, Mass, 1992, tomado de Alfonso Fuggetta, op. cit., p. 290.
[8] Camilo Ocampo, Begoña Albizuri y Pere Botella, op. cit., p.52.
[9] Alfonso Fuggetta, op. cit., p. 290.

A pesar de que no se cuenta con "la definición" del término, los autores investigados coinciden con respecto a la idea básica detrás del CASE: asistir en las tareas de cada una de las fases del proceso de desarrollo de software con herramientas que ...

- brinden apoyo a las primeras fases con productos que generen diagramas y pantallas prototipo;
- se enfoquen en la etapa de la implementación, y en la generación de código automatizado y de casos de prueba;
- sean utilizadas en conjunto con lenguajes de tercera y cuarta generación, producidos a través de programas que controlen las especificaciones del proyecto.

1.2. Panorama Histórico

El uso de herramientas para facilitar las tareas humanas se remonta a épocas lejanas. En nuestros días ocurre lo mismo, con la adquisición de nuevas técnicas se procura hacer el trabajo más sencillo y eficiente. Por ejemplo, a mediados del siglo XX muchas áreas del conocimiento se valían de libros, tablas, bolígrafos, lápices, etc., para efectuar el trabajo. Posteriormente, las fórmulas matemáticas utilizadas en distintas tareas, fueron integrándose en algoritmos para ser resueltos mediante programas de cómputo, originando con esto el proceso de automatización.

Hoy en día, la importancia de la tecnología informática es tal que se le ha llegado a comparar —en cuanto a los beneficios suscitados— con uno de los hechos históricos más importantes del mundo moderno: la Revolución Industrial —ocurrida a finales del siglo XVIII— pues ambos acontecimientos transformaron la economía y la sociedad en su momento.

De acuerdo con John Parkinson,[10] una de las primeras herramientas usadas en el desarrollo de software fueron las tarjetas perforadas que permitieron la creación de programas en línea utilizando el código binario. La productividad mejoró aún más con la llegada del editor, pues resultaba más fácil modificar una línea del programa que perforar una nueva tarjeta. Además, el conservar físicamente un archivo creado con un editor representaba menos trabajo que el mantener un conjunto de tarjetas.

Por los años cincuenta, aparece la programación tipo "Batch" o por lotes utilizando el lenguaje de máquina.

En la década siguiente empieza a codificarse en un lenguaje de programación sencillo, el lenguaje ensamblador y a ocuparse técnicas gráficas conocidas como diagramas de flujo, herramientas enfocadas a revisar y corregir posibles errores de lógica. Con la aparición de los lenguajes de "alto nivel", a mediados de los sesenta, los diagramas presentaron la novedad de ocupar una metodología alejada del famoso "go to", usando técnicas de análisis y diseño estructurado.

A partir de la "Crisis del Software", en 1968, se consideró como la principal causa del problema a la producción del código, por lo que la tecnología informática —llamada desde entonces Ingeniería de Software— utilizó herramientas exclusivamente en las últimas etapas de producción. De aquí el éxito de los diagramas de flujo, pues con ellos se esperaba corregir el problema, pero a pesar de los intentos realizados, el dilema continuó, obligando a un cambio de enfoque: prestar atención a las etapas iniciales del desarrollo de software utilizando nuevas técnicas en la especificación, el análisis y el diseño.

[10] John Parkinson, Making CASE Work, strategies for the successful introduction of CASE technology in comercial information systems development, NCC, London, 1991, p. 15.

Estos incidentes establecieron las dos generaciones del CASE:

a) La primera generación del CASE.

Esta generación, iniciada en los setenta, basaba su interés en la codificación y ejecución del programa. Las herramientas utilizadas para estos procesos fueron los compiladores, los depuradores y los ensambladores que, agrupados, forman parte de las herramientas CASE de "bajo nivel".

Dentro de esta generación se consideran a las herramientas que corren bajo un "mainframe" y están basadas en el texto, como el editor.

b) La segunda generación del CASE.

La introducción de computadoras personales —a inicios de los ochenta— cambió las herramientas basadas en el texto por herramientas gráficas que dieron apoyo al análisis y al diseño de sistemas. Estas herramientas que auxilian a las etapas iniciales del desarrollo de software, son llamadas herramientas CASE de "alto nivel".

A mitad de los setenta, la importancia dada al manejo de cantidades impresionantes de datos originó los Sistemas Manejadores de Bases de Datos (SMBD) y los lenguajes de cuarta generación —empleados para accesar de una manera sencilla los datos contenidos en una base de datos. Eventualmente las herramientas gráficas se integraron con las bases de datos para producir poderosas herramientas de desarrollo.

Por los ochenta, el uso de redes y de sistemas abiertos fueron desde entonces indispensables para los grupos de trabajo. Particularmente con el crecimiento de las empresas y la aparición de nuevos departamentos y con ellos, necesidades específicas.

Parte del panorama histórico del CASE queda plasmado en la Figura No. 1, que se muestra a continuación:

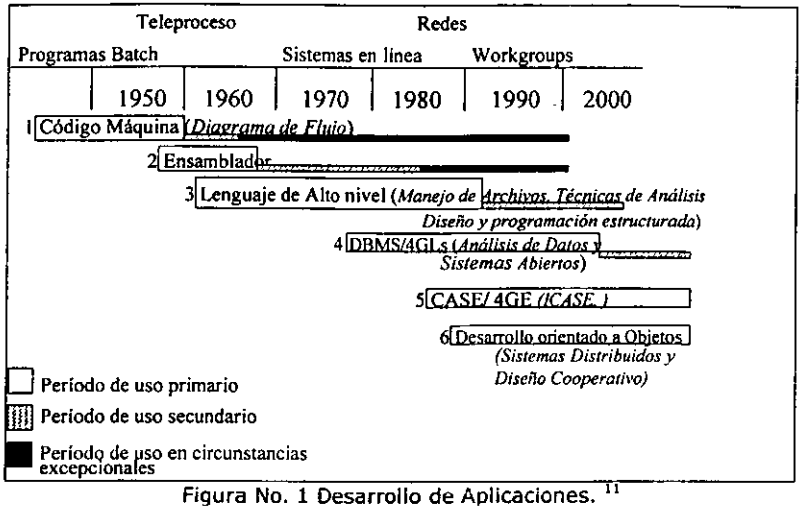


Figura No. 1 Desarrollo de Aplicaciones. [11]

Durante los noventa, el desarrollo y la investigación acerca del CASE fue reduciéndose debido a problemas como su alto costo, la resistencia a utilizarlos y, la ignorancia para echarlos a andar por desconocer su metodología base. Además de los problemas surgidos por el uso de las herramientas CASE:

- El CASE se entendía como una herramienta independiente de los procesos, de los entornos, de la administración y de los usuarios.
- El soporte a multiusuarios era insuficiente.
- La interfaz con el usuario era deficiente.
- La documentación generada era pobre.
- El CASE no generaba prototipos valiosos utilizando lenguajes de cuarta generación.
- No existía integración entre las herramientas de desarrollo.

[11] John Parkinson, op. cit., p. 8.

- No eran visibles la reducción de tiempo y costo en el desarrollo de sistemas.
- La etapa de mantenimiento ocupaba la mayor parte del presupuesto establecido para la realización de un proyecto, pues en ese momento se trataba de corregir errores, principalmente los generados en la fase de especificación de requerimientos, etapa donde se produce el mayor número de fallas (Ver Figura No. 2).

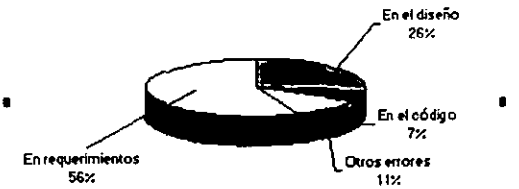


Figura No. 2 Fuentes de error en el desarrollo de sistemas.[12]

- Existía un gran porcentaje de proyectos jamás terminados o utilizados, (Ver Figura No. 3).

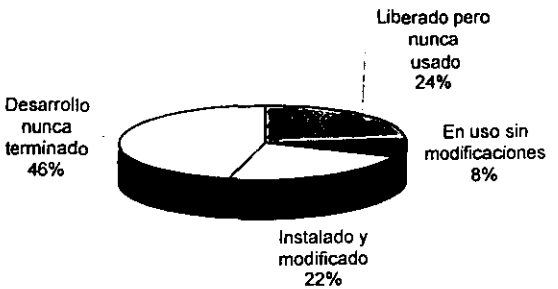


Figura No. 3 Liberación y uso del software.[13]

Por estas y otras deficiencias, la tecnología CASE ha tenido que evolucionar a lo que se conoce hoy como tecnología de procesos. Anteriormente, el interés de estudio del

[12] John Parkinson, op. cit., p. 11.
[13] Ibid., p. 10.

CASE era solamente obtener un buen producto de software, ahora, para mejorar la calidad de los productos de software y conseguir los objetivos trazados a un tiempo y costo previamente establecidos, hay que ocuparse del proceso.

1.3. Componentes del CASE

Un buen entorno de Ingeniería de Software debe construirse sobre una arquitectura que englobe los elementos de hardware y software, además de las metodologías aplicadas en el proceso de desarrollo.

La arquitectura de un entorno CASE está conformada por los bloques siguientes:

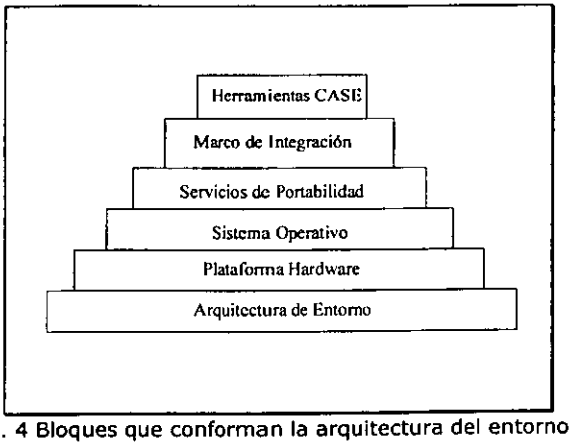


Figura No. 4 Bloques que conforman la arquitectura del entorno CASE.[14]

a) Herramientas CASE.

Pressman[15] afirma que cualquier clasificación con respecto a las herramientas CASE puede crear confusión, ya que es posible colocar una herramienta específica dentro de una categoría cuando se le creía pertenecer a otra. O puede darse el caso de que se

[14] Roger S. Pressman, Ingeniería del Software, un enfoque práctico, 3a. edición, Mc Graw Hill, p. 748.
[15] Roger S. Pressman, op. cit. , p. 750.

omita alguna división, haciendo a un lado al grupo de herramientas que la conforman. A pesar de estos problemas, Pressman mismo considera necesaria la creación de una clasificación de las herramientas CASE, pues asegura que de esta manera puede entenderse fácilmente el alcance y la aplicación de las herramientas CASE dentro del proceso de ingeniería de software. La clasificación de las herramientas CASE que propone Pressman[16], de acuerdo a su funcionalidad, es como sigue:

Función 1) Herramientas de Planificación de Sistemas de Gestión.

Este tipo de herramientas produce un modelo completo de la organización de acuerdo a la forma en que los distintos departamentos manejan la información de la empresa.

Función 2) Herramientas de Gestión de Proyectos.

Herramientas con las cuales el administrador puede estimar el esfuerzo, el costo, la duración del proyecto, el número de elementos necesarios para llevarlo a cabo, planificándolo de una manera realista y dándole seguimiento en sus distintas etapas, por ejemplo: Microsoft Project.

Función 3) Herramientas de Soporte.

Las herramientas de documentación, herramientas para la gestión de redes y el software del sistema, herramientas para la gestión de bases de datos pertenecen a esta clasificación de las herramientas CASE.

Función 4) Herramientas de Análisis y Diseño.

Este tipo de herramientas permite la creación de un modelo del sistema que se va a desarrollar, empezando con los requisitos y terminando con el diseño de su arquitectura, además, evalúa la calidad del modelo y auxilia en la eliminación de los errores —antes de que estos se propaguen a fases posteriores al diseño.

Función 5) Herramientas de Programación.

Dentro de esta clasificación quedan comprendidos los editores, los compiladores, los depuradores, los sistemas de consulta a bases de datos, los generadores de código,

[16] Roger S. Pressman, op. cit., p. 750.

los lenguajes estructurados de cuarta generación y los lenguajes orientados a objetos, por ejemplo: System Architect.

Función 6) Herramientas de Integración y Prueba.

Incluye a las herramientas generadoras de datos de prueba; a las que analizan el código fuente antes y después de la ejecución; también a todas aquellas herramientas que simulan el trabajo del hardware y a las que planifican el desarrollo y control de las pruebas.

Función 7) Herramientas de Creación de Prototipos.

Son las herramientas que crean desde un prototipo en papel, para mostrar al cliente el funcionamiento y comportamiento del sistema, hasta aquellas que diseñan rápidamente pantallas y/o generan código fuente.

Función 8) Herramientas de Mantenimiento.

Comprende todas aquellas herramientas que efectúan la ingeniería inversa, también a las que ayudan en la reestructuración y en el análisis del código.

Función 9) Herramientas de Estructura.

Son herramientas que gestionan las bases de datos, la configuración y sobre todo la integración entre herramientas CASE, forman parte de un entorno I-CASE.

Fuggetta[17], en comparación, propone una clasificación de los sistemas CASE en tres categorías:

Categoría 1) Herramientas.

Componentes de software que apoyan una tarea especifica en el proceso de software, véase tabla No. 1:

Clases de herramientas	Subclases
Edición	Editores de texto y editores gráficos
Programación	Ensambladores, Compiladores, Depuradores, Intérpretes, etc.
Verificación y Validación[18]	Analizadores estáticos y dinámicos.
Administración de la Configuración	Administración de versiones, Identificación de cada elemento de software, control de cambios y la administración de esta información.
Métricas	Herramientas que analizan el código fuente y herramientas que monitorean la ejecución de los programas.
Administración del Proyecto	Productos usados para estimar los costos de la producción de software, herramientas que apoyan la planeación del proyecto y las herramientas que apoyan la comunicación entre los integrantes de un equipo de trabajo.
Herramientas varias	Productos difíciles de clasificar como las hojas de cálculo.

Tabla No. 1 Clases de herramientas CASE.[19]

[17] Alfonso Fuggetta, op. cit., p. 290.
[18] Verificación: comprobación de que el producto de software cumpla con la definición de los requerimientos. Validación: revisión de que el producto haga lo que el cliente desea, ibid., p.294.
[19] Ibidem.

Categoría 2) Workbenches.

Conjunto de herramientas con algún grado de integración que apoyan actividades específicas del proceso de software, véase tabla No. 2:

Clases de Workbenches	
Planeación y modelado de negocios	Construyen modelos de empresas para evaluar los flujos de información y los requerimientos generales, e identifican prioridades en el desarrollo de los sistemas de información.
Análisis y diseño	Productos que ayudan en las actividades del análisis y el diseño con diagramas de flujo, relación/entidad, de estados, etc.
Desarrollo de la interfaz usuario	Cuenta con editores gráficos, simuladores para probar el desarrollo de la interfaz antes de integrarla con las aplicaciones, etc.
Programación	Proporciona las facilidades de edición, compilación y depuración como apoyo a la programación.
Verificación y Validación	Incluye analizadores estáticos y dinámicos, herramientas para producir, almacenar y administrar los datos de prueba, etc.
Ingeniería inversa y mantenimiento	Incluye un reestructurador de código, un diagramador de flujo y, un generador de referencias cruzadas, entre otras herramientas.
Administración de la Configuración	Este tipo de workbenches integra herramientas que soportan el control de versiones y de cambios.
Administración del proyecto	Cuenta con herramientas para el monitoreo de las distintas actividades involucradas en la administración de un proyecto.

Tabla No. 2 Clases de Workbenches.[20]

[20] Alfonso Fuggetta, op. cit., p. 296.

Categoría 3) Entornos.

Colección de herramientas y workbenches que apoyan a todo el proceso de software o a una parte sustancial del mismo, existen cinco clases de entornos según el grado de integración. Como se observa en la tabla siguiente:

Toolkits o juegos de herramientas.	Apoyan diferentes actividades en el proceso de software, p. ej. programación, administración de la configuración y la administración de proyectos.
Entornos centrados en el lenguaje.	Por lo regular estos entornos están escritos en el lenguaje para el cual fue desarrollado, por lo que es común reusar parte del código incluido.
Entornos integrados.	Utilizan mecanismos estándares para integrar las herramientas y las workbenches.
Entornos de cuarta generación.	Subclase de los ambientes integrados, apoya el desarrollo de una clase particular de programas.
Entornos centrados en el proceso.	Véase capítulo 3 de la tesis.

Tabla No. 3 Clases de entornos.[21]

b) Marco de Integración.

Conjunto de programas "especializados" de software que permiten la comunicación de los datos entre las distintas herramientas CASE.

Para añadir nuevas herramientas a cada bloque y hacerlas compatibles entre sí, se crearon estándares. Uno de ellos, el IPSE (*Integrated Project Support Environment*), cuya traducción es Entorno Integrado de Soporte de Proyectos, el cual es simplemente software que corre en red, e integra a un conjunto de herramientas mediante una interfaz usuario, desde donde es posible accesar todas las actividades generadas por las herramientas. IPSE permite compartir información entre las herramientas y el acceso a un depósito común.

[21] Adaptado a partir de Alfonso Fuggetta, op. cit., p. 299.

c) Servicios de portabilidad.

Para un entorno, los servicios de portabilidad hacen posible el uso de las herramientas CASE y el marco de integración en plataformas distintas sin mayores problemas. Este servicio es el paso intermedio entre las herramientas CASE y el marco de integración.

d) Sistema operativo.

Se encarga de la administración de la red y de la gestión a la base de datos.

e) Plataforma Hardware.

Es el equipo utilizado para correr el software —las herramientas CASE, el marco de integración y los servicios de portabilidad— del entorno, con ayuda del sistema operativo.

f) Arquitectura del Entorno.

Constituye la base del CASE, está compuesta por la plataforma hardware y el soporte del sistema operativo, pero debe considerar también el trabajo humano que se aplica durante el proceso de software.

Otra clasificación localizada en la bibliografía, son los tres niveles CASE de acuerdo a la tecnología:

a) Nivel que apoya al proceso de producción.

Incluye herramientas que auxilian a las actividades de especificación, de diseño, de implementación, de prueba, etc. Las herramientas CASE consideradas dentro de este punto han sido las más utilizadas, pues fueron las primeras en ocuparse. Existe un número considerable de herramientas enfocadas en las actividades de diseño e implementación, pero no puede decirse lo mismo con respecto a las que manejan los requerimientos y la etapa de mantenimiento.

b) Nivel que administra el proceso.

Incluye herramientas que apoyan el modelado y la administración del proceso.

La tecnología de procesos, es decir, el modelado, la evaluación, el mantenimiento y la gestión del proceso de software, divide a la ingeniería de software en dos grandes áreas: la ingeniería del producto y la ingeniería del proceso.

c) Meta-CASE.

Son herramientas generadoras de nuevas herramientas para apoyar el proceso de producción y a la administración del proyecto.

1.4. Selección de una herramienta CASE

Seleccionar una herramienta CASE no es una tarea fácil, ya que no existe la mejor herramienta y para elegirla es necesario hacer todo un estudio. No obstante, se considera funcional a todo CASE que cumple las siguientes características, conocidas con la frase de "las tres C":

a) Consistency.- Consistencia o estabilidad.

Las acciones que promete efectuar la herramienta CASE, las debe realizar independientemente de la plataforma utilizada.

b) Completeness.- Integridad.

Proteger a la información, muchas veces compartida entre herramientas, de una pérdida accidental debido a:

- Caídas del sistema durante el procesamiento de las transacciones.
- Anomalías causadas por el acceso concurrente a una BD.
- Anomalías causadas por la distribución de los datos en varias computadoras.
- Errores lógicos que violen la consistencia de la información.

c) Conformance to standards.- Ajuste a los estándares.

Las herramientas CASE incrementan su productividad al basarse en estándares, por ejemplo, existen herramientas CASE que se ajustan a la norma internacional ISO

9075-1987, para estandarizar su lenguaje de consulta y manejo de datos SQL (Structured Query Language).

El proceso de selección y evaluación del CASE debe ser tomado como un proyecto de la mayor importancia. La primera etapa a estudiar es el análisis de las necesidades existentes que tendrán que ser satisfechas a través de la implantación de la herramienta que se va a seleccionar, por lo que deben identificarse:

- Las funciones que debe cumplir la herramienta.
- Las facilidades de uso que debe prestar.
- Las limitaciones y restricciones que se derivan de su operación.

En función de lo anterior, podrá deducirse qué tipo de herramienta podría ser la más adecuada.

Otros factores a tomar en cuenta en la selección de las herramientas son:

- Tipo(s) de plataforma(s) sobre la(s) que deberá funcionar la herramienta.
- Requisitos físicos (espacio en disco, memoria RAM, etc.).
- Necesidad de integración con herramientas existentes.
- Necesidad de acceso simultáneo para diferentes usuarios.
- Necesidad de compartir datos con aplicaciones externas.
- Si en la organización ya existe una metodología, la herramienta deberá soportarla.
- Capacidad de integración a la arquitectura existente.

1.5. Conceptos relacionados

Algunos de los conceptos que se emplean en el mundo de las herramientas CASE son los siguientes:

CASE TECHNOLOGY

Tecnología de software que se basa en una metodología con la cual proporciona una disciplina de ingeniería automatizada para el desarrollo de software y la administración del proyecto.

HERRAMIENTA CASE

Herramienta individual que automatiza una tarea del ciclo de vida del software.

SISTEMA CASE

Conjunto de herramientas CASE integradas que comparten una interfaz de usuario y corren bajo un ambiente compartido común.

CASE TOOL KIT

Conjunto de herramientas CASE integradas que han sido diseñadas para trabajar juntas y de esa manera automatizar completamente el ciclo de vida del software.

METODOLOGÍA CASE COMPANION

Conjunto de herramientas CASE que automatizan, de acuerdo a una metodología CASE en particular, la producción de la documentación y otras tareas.

CASE WORKSTATION

Es una Workstation equipada con herramientas CASE que automatizan varias funciones del ciclo de vida del software.

CASE HARDWARE PLATFORM

Plataforma operativa para las herramientas CASE cuya arquitectura de sistema está compuesta por varios equipos de hardware.

CASE WORKBENCH

Conjunto de herramientas integradas de software que apoyan actividades específicas del proceso de software.

1.6. Beneficios del CASE

El uso correcto de las herramientas CASE dentro de una organización se espera generen los siguientes beneficios en cada una de las fases del proceso de desarrollo de software:

- Fortalecer un ambiente interactivo.
- Reducir costos, no sólo los ocasionados por el mantenimiento.
- Mejorar la calidad del software.
- Reducir el tiempo empleado en el desarrollo de sistemas.
- Incrementar la precisión en lo que se hace y cómo se hace.
- En general, aumentar la productividad.

Además, se espera, con las herramientas CASE generar la documentación del proceso de software, desde la definición de los requerimientos hasta el diseño y la implementación.

1.7. Estado del Arte

La tecnología CASE pensada como "la solución" para muchos de los problemas de la ingeniería de software, es tan solo una de tantas formas de ayudar en el proceso de desarrollo de software. Como se ha venido repitiendo, en la actualidad, esta tecnología apoya en casi todos los procesos administrativos de una organización por pequeña que ésta sea, mas la tecnología no es suficiente.

Anteriormente, los analistas en sistemas enfatizaban su interés en las habilidades técnicas —las bases de datos, la programación, el ambiente técnico, etc.— haciendo a un

lado el conocimiento del negocio y la comunicación. En la actualidad, el CASE ha girado hacia temas como la automatización, la gestión del proceso de software y la reingeniería, pero haciendo hincapié en el conocimiento del negocio, de los recursos, de los productos y del mercado.

A continuación se muestra la evolución del CASE, partiendo del concepto de producto hasta un conocimiento más general que incluye el producto y el soporte del proceso.

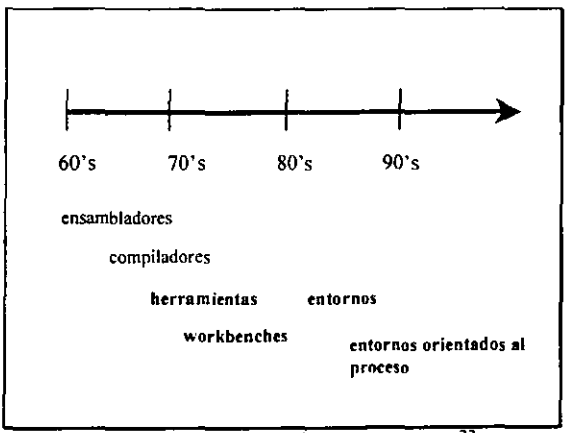


Figura No. 5 Evolución de los CASE.[22]

[22] Camilo Ocampo, Begoña Albizuri y Pere Botella, op.cit., p. 52.

CAPÍTULO 2

I-CASE

2. Integración de Herramientas CASE

2.1. Definición

"La integración exige representaciones consistentes de la información, interfaces estandarizadas entre las herramientas, un mecanismo homogéneo para la comunicación entre el ingeniero de software y cada herramienta y un enfoque efectivo que permita al I-CASE ejecutarse sobre distintas plataformas hardware y sistemas operativos".[23]

---

[23]Roger S. Pressman, op. cit., pp. 771-772.

De acuerdo con Pressman, todo I-CASE presenta las siguientes características:

- Autoriza el acceso directo a cualquier herramienta contenida en el entorno.
- Provee de un mecanismo que permite compartir la información entre estas herramientas.
- Comunica los cambios a los elementos relacionados.
- Mantiene un control global sobre la gestión de la configuración del proyecto, reuniendo información técnica útil.
- Presenta una interfaz hombre-máquina consistente para cada una de las herramientas.
- Presenta la característica de interoperabilidad, es decir, permite el acceso común por parte de las herramientas a los datos compartidos.

También Pressman apunta que un I-CASE puede alcanzar la integración total siempre y cuando presente las siguientes características:

a) Capacidad de control.

   Mantener la integridad del entorno, automatizar procesos y procedimientos estándares. Notificar a cada una de las herramientas relacionadas sucesos importantes y enviar peticiones para efectuar acciones y servicios sobre otras herramientas.

b) Gestión de metadatos.

   A la información técnica generada por las distintas herramientas CASE se le conoce como metadatos e incluye:

   - Definiciones de Objetos —como tipos, atributos, representaciones y relaciones válidas.
   - Relaciones y dependencias entre objetos.
   - Reglas de diseño del software.
   - Procedimientos de las distintas fases e informes del proceso de desarrollo.

2.2. Componentes del I-CASE

El I-CASE engloba los elementos del CASE en cuatro elementos, mostrados en la figura No. 6. En el arreglo de los bloques del I-CASE, no existe alguna diferencia fundamental, simplemente es otra forma de integrar los componentes:

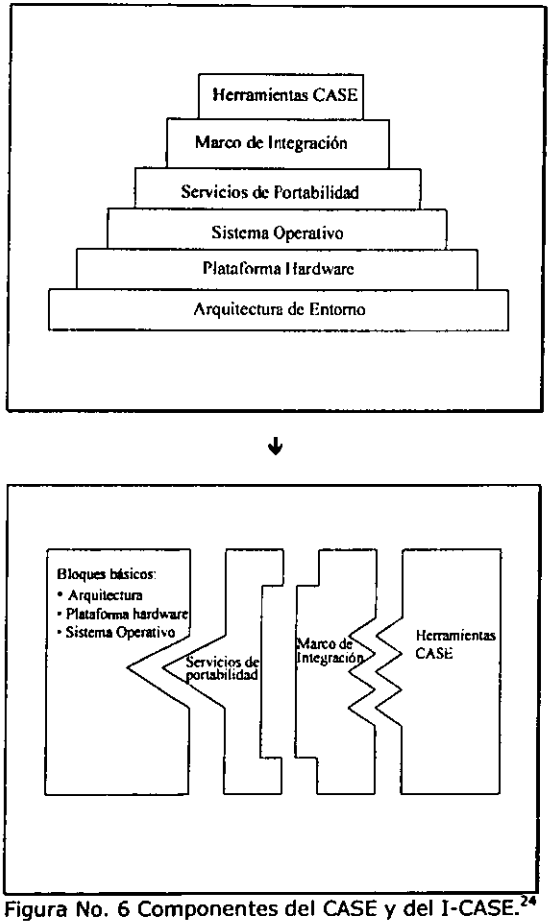


Figura No. 6 Componentes del CASE y del I-CASE.[24]

[24] Roger S. Pressman, op. cit., p. 773.

a) Bloques básicos.

Dentro del primero de los bloques quedan comprendidos la arquitectura del entorno, la plataforma hardware y el sistema operativo, elementos independientes en la arquitectura CASE tradicional.

b) Servicios de portabilidad.

Servicios que permiten utilizar las herramientas integradas por cualquier sistema operativo actuando bajo un tipo de arquitectura indistinta.

c) Marco de Integración.

Elemento que facilita la transferencia de información dentro y fuera del almacén que la guarda.

La principal diferencia que existe entre la arquitectura CASE y la del I-CASE radica precisamente en el Marco de Integración, ya que en el I-CASE su estructura es más compleja, véase la figura No. 7:

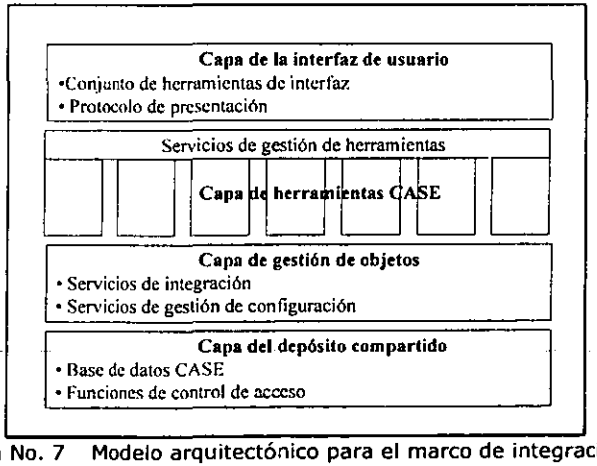


Figura No. 7 Modelo arquitectónico para el marco de integración.[25]

[25] Roger S. Pressman, op. cit., p. 777.

a) Capa de la interfaz de usuarios.

Proporciona un medio de comunicación consistente entre el usuario y las herramientas del entorno, pues aparte de contener software para lograr la comunicación hombre-máquina, cuenta con una biblioteca de objetos para la representación gráfica. Esta capa incorpora un conjunto de herramientas manejadas por una interfaz normalizada mediante un protocolo común de representación, es decir, un conjunto de normas que dan el mismo aspecto a todas las herramientas y que define los mecanismos para su acceso. Especifica la forma de utilizar el teclado, el ratón, los nombres de los objetos, los iconos, la organización y los nombres de menús y los convenios de presentación en pantalla.

b) Capa de herramientas CASE.

Además de herramientas, esta capa contiene un servicio de gestión de herramientas (SGH), mecanismo de control que coordina el acceso a éstas; además de administrar el comportamiento, la sincronización y la intercomunicación del flujo de información entre el "depósito" y las herramientas; por lo que cada herramienta CASE debe tener acceso a la capa de gestión de objetos. La integración de datos se consigue mediante las capas de gestión de objetos y el depósito compartido.

c) Capa de gestión de objetos.

El software localizado en esta capa, conocida como capa de gestión de objetos (CGO), proporciona el mecanismo para integrar las herramientas. A la CGO la componen los servicios de integración, es decir, el conjunto de módulos estándar que acoplan las herramientas al sistema de almacenamiento; proporcionando servicios de configuración de la información, identificando a todos los objetos, controlando versiones y proporcionando soporte para el control de cambios, así como para las auditorías y la contabilidad.

d) Capa del depósito compartido.

Presenta una base de datos que es vista como un depósito común, lugar donde se almacenan la base de datos CASE y las funciones de control de acceso que permiten a la capa de gestión de objetos (CGO) interactuar con la base de datos. Los mecanismos que integran las distintas herramientas en un entorno I-CASE son implementados de acuerdo a la arquitectura, la plataforma y la filosofía del diseñador del entorno. Pero todos sin excepción, implementan mecanismos de ejecución y mecanismos de comunicación utilizando uno de los estándares de los I-CASE, como lo es el PCTE, el Entorno de Herramientas Portables Comunes.

El PCTE establece como mecanismos básicos a los mecanismos de ejecución y de comunicación que manipulan entidades u objetos; otro mecanismo presente en los I-CASE de acuerdo al PCTE es el mecanismo de distribución, enseguida se explican cada uno de ellos:

a) Mecanismos de ejecución.

Son los mecanismos contenidos en el marco de integración que permiten suspender, reactivar y finalizar un proceso o una herramienta.

b) Mecanismos de comunicación.

Son los mecanismos contenidos en el marco de integración que gestionan la comunicación entre distintos procesos, estableciendo colas de mensajes que llevan información entre las herramientas.

c) Mecanismos de distribución.

Administran la red y supervisan las estaciones de trabajo que se le conectan; gestionan cada nodo de la red como un objeto; distribuyen la operación de los mecanismos básicos de una manera "transparente" a través de la red.

2.3. El depósito en un sistema I-CASE

El depósito es un metamodelo que determina, entre otras cosas, la forma de almacenar la información, el uso de los datos por parte de las herramientas, el control de la seguridad e integridad de los datos y el almacenamiento de nuevas necesidades. En un metamodelo de este tipo se encuentra contenida toda la información de la ingeniería de software del entorno.

El depósito es una base de datos que almacena toda la información que utiliza un ingeniero de software, quien accede a la información a través del conjunto de herramientas integradas. La información que se almacena en el depósito es del siguiente tipo:

Información de la empresa	Guiones de prueba de regresión
Estructura organizacional	Resultados de prueba
Análisis del área comercial	Análisis estadísticos
Funciones comerciales	Métricas de calidad del software
Reglas comerciales	Información de gestión de proyectos
Modelos de proceso	Planes del proyecto
Arquitectura de la información	Estructura de partición
Diseño de la aplicación	Estimaciones
Reglas metodológicas	Agendas
Representaciones gráficas	Carga de recursos
Diagramas del Sistema	Informes de problemas
Estándares de denominación	Peticiones de cambio
Reglas de integridad referencial	Informes de estado
Estructuras de datos	Información de auditoría
Definiciones de procesos	Documentación del sistema
Definiciones de clases	Documentos de requisitos
Árboles de menús	Diseños externo/interno
Criterios de rendimiento	Manuales de usuario
Restricciones de tiempo	Construcción
Definiciones de pantallas	Código fuente
Definiciones de informes	Código objeto
Definiciones lógicas	Instrucciones de construcción del sistema
Lógica de comportamiento	Imágenes binarias
Algoritmos	Dependencias de configuración
Reglas de transformación	Información de cambio
Validación y verificación	
Plan de prueba	
Casos de datos prueba	

Tabla No. 4 Contenido del depósito I-CASE.[26]

[26] Roger S. Pressman, op. cit., p. 784.

Resumiendo el contenido de la tabla No. 4, en el depósito I-CASE se almacena información del siguiente tipo:

- El problema a resolver.
- Información acerca del problema.
- La solución del problema de acuerdo a como se vaya resolviendo.
- La metodología a seguir en el proceso de desarrollo de software.
- La planeación del proyecto.
- Información acerca de la organización.

2.3.1. Servicios proporcionados por todo depósito I-CASE

En general, un depósito I-CASE debe proporcionar los siguientes servicios básicos:

a) Los mismos servicios que proporciona un sistema gestor de Bases de datos sofisticado:

- Almacenamiento no redundante.

  El depósito I-CASE almacena toda la información relacionada con el desarrollo de software.

- Acceso de alto nivel.

  Se utiliza un solo mecanismo para que cada herramienta CASE tenga acceso a los datos comunes.

- Independencia de los datos.

  Los cambios en la configuración física de los datos no afectan a las aplicaciones de las herramientas CASE.

- Control de transacciones.

  El depósito se encarga de mantener la integridad de los datos cuando ocurren accesos simultáneos o caídas del sistema.

- Seguridad.

  El depósito controla los accesos y las modificaciones a los datos almacenados.

- Consultas a los datos y generación de informes.

  La consulta debe realizarse utilizando una interfaz apropiada como SQL.

- Apertura.

  El depósito CASE debe facilitar la importación/exportación de los datos.

- Soporte multiusuario.

  Debe permitir el acceso simultáneo a los datos facilitando la interacción al almacenamiento de acuerdo a los protocolos y a las facilidades de la red.

b) Los servicios específicos de todo depósito I-CASE:

- Almacenamiento de estructuras de datos sofisticadas.

  El depósito almacena estructuras de datos complejos, diagramas, documentos, archivos, datos simples, el metamodelo y la descripción de los sistemas en desarrollo.

- Mantenimiento de la integridad.

  El depósito contiene reglas que describen restricciones y requisitos sobre la información contenida por medio de las herramientas CASE.

- Interfaz de gran contenido semántico.

  La semántica contenida en el depósito permite compartir los datos entre las herramientas.

- Gestión de procesos o proyectos.

  El depósito contiene información sobre las características de cada uno de los proyectos en particular y sobre la organización general del desarrollo de software, esto ayuda a coordinar automáticamente el desarrollo técnico con la gestión del proyecto. Se puede actualizar el estado del proyecto como resultado del uso de las herramientas CASE.

c) El depósito I-CASE incorpora, además, las funciones siguientes:

- Integridad de datos.

  Asegura la consistencia de los datos entre objetos relacionados y valida mediante una función las entradas al depósito.

- Información compartida.

  Por medio de un mecanismo se comparte información entre las distintas herramientas; además de que gestiona el acceso multiusuario a los datos usando la integridad.

- Integración datos-herramienta.

  Controla el acceso a los datos estableciendo un modelo al cual todas las herramientas I-CASE acceden.

- Integración datos-datos.

  El sistema de gestión de la base de datos (SGBD) relaciona todos los objetos.

- Establecimiento de la metodología.

  Los datos contenidos en el depósito son definidos de acuerdo a una metodología.

- Estandarización de documentos.

  Define un enfoque estándar para crear documentos ligados al proyecto a desarrollar.

2.3.2. Características del depósito enfocadas a la gestión de la configuración

Otras características del depósito enfocadas a la gestión de la configuración son:

- Gestión de versiones.

  El depósito almacena las distintas versiones de un desarrollo utilizando un algoritmo de compresión que minimiza el espacio de almacenamiento, y las recupera con gran prontitud.

- Seguimiento de dependencias y gestión de cambios.

  El depósito controla y almacena todas las relaciones que existen entre los distintos datos u objetos que almacena.

- Seguimiento de los requisitos.

  Esta función depende de la gestión de enlace. Puede tratarse de un seguimiento hacia delante, resultado de una especificación de requisitos o de un seguimiento hacia atrás, identificando los requisitos generados de un informe específico.

- Gestión de la configuración.

  Llevada a cabo por herramientas ajenas al depósito. Depende de los datos contenidos y de las utilidades de gestión de relaciones para trabajar correctamente. En particular, depende de la gestión de enlace que inicia la regeneración del código destino para reflejar el cambio y montar los módulos modificados.

- Trayectorias de auditoría.

  La gestión de cambios genera información acerca de la razón, del cuándo y por quién se realizaron las modificaciones, un mecanismo de activación del depósito introduce este tipo de información adicional.

2.3.3. Estándares con respecto al depósito I-CASE

Los estándares permiten a las herramientas CASE el intercambio de información contenida en el depósito, algunos ejemplos se mencionan a continuación:

a) Estándar de Diccionario de Recursos de Información (IRDS). Estándar oficialmente aprobado por el ANSI. Crea "puentes" entre las herramientas de análisis y desarrollo y los generadores de código.

b) Estándar de Integración de Herramientas Atherton (ATIS). Se basa en la definición de una arquitectura de almacenamiento, comprende la gestión de configuración, la

integración de herramientas, la seguridad de los datos y la portabilidad entre plataformas.

c) Estándar de Interfaz Común de Ada (CASI). Basado en herramientas que desarrollan software en lenguaje Ada, define la interfaz entre herramientas que conforman un entorno de desarrollo en lenguaje Ada.

d) Entorno de Herramientas Comunes Portables (PCTE). Modelo de arquitectura desarrollado por la comunidad europea para el uso de las herramientas CASE. Se basa en la portabilidad, el control de concurrencia, las redes distribuidas, la arquitectura de los datos y la interfaz de usuario.

e) Generador Industrial de Software y Ayudas al Mantenimiento (SIGMA). Estándar utilizado en países orientales, entre ellos Japón. SIGMA se utiliza en forma similar al PCTE.

f) Formato de Intercambio para el Diseño Electrónico (EDIF). Se basa en el formato de los datos para el intercambio de información entre las distintas herramientas CASE.

2.4. CASE vs I-CASE

I-CASE hace referencia al entorno de soporte que integra a las distintas herramientas utilizadas por la Ingeniería de Software, y CASE es el conjunto de herramientas que automatizan las actividades manuales del ingeniero. Sin embargo, ambos conceptos tienen el mismo objetivo.

Dos características distinguen al I-CASE de un juego de herramientas CASE:

a) Los datos compartidos entre herramientas y localizados en un depósito común.

b) El uso de las herramientas independientes puede seguir una o varias metodologías de desarrollo de software.

2.5. Beneficios del I-CASE

No se puede negar que cada una de las herramientas CASE contribuye enormemente en el trabajo del ingeniero, pero realmente el potencial de éstas se localiza al momento de integrarlas, dando como resultado los siguientes beneficios:

a) Permiten compartir datos mediante una transferencia de información proveniente de modelos, programas y documentos entre herramientas y entre las distintas etapas del proceso de ingeniería de software.
b) Minimizan el esfuerzo y el tiempo requerido para llevar a cabo actividades de soporte, por ejemplo: la gestión de la configuración del software, el control de calidad y la generación de documentos.
c) Logran un mayor control de los proyectos.
d) Mejoran la comunicación entre los analistas.
e) Benefician la coordinación entre roles, sobre todo en grandes proyectos.

2.6. Estado del Arte

Existen artículos donde se indica la muerte del I-CASE. Seguramente esto no sea del todo cierto, pues en la actualidad, los I-CASE ampliaron su horizonte hacia procesos dinámicos, pues asumieron que cualquier soporte de sistemas con un enfoque estático no sería útil por un largo tiempo. El IPSE (Integrated Project Support Environment) extendió las herramientas I-CASE añadiendo servicios para la administración de proyectos, la implementación y el mantenimiento de sistemas. De aquí el florecimiento de los ambientes PSEE.

CAPÍTULO 3

PSEE

3. Ambientes de Ingeniería de Software Centrados en el Proceso (PSEE)[27]

3.1. Definición

Antes de que la tecnología de procesos surgiera, los modelos creados utilizando la tecnología CASE se consideraban representaciones incompletas por lo complicado de su mantenimiento. En cambio, con la llegada de la tecnología de procesos, los CASE mejoraron al considerárseles un apoyo total en el proceso de software. Por su parte, la tecnología I-CASE —ideal para el desarrollo de sistemas estáticos— fue sustituida por ambientes activos llamados comúnmente PSEE, los cuales se han especializado en el manejo de procesos dinámicos de la ingeniería de sistemas, por ejemplo: el trabajo cooperativo, la comunicación interactiva, la calidad total y la administración de la configuración.

---

[27] Para este capítulo me guío principalmente en el libro de Madhavji, N.H., Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Pankaj K. Garg, y Mehdi Jazayeri (eds.), Los Alamitos, CA, 1996.

El término "Ambientes Centrados en el Proceso" —o "Ambientes Orientados al Proceso"— fue inventado en 1990 por Madhavji, Gruhn, Deiters y Schäfer para el proyecto PRISM.[28] Posteriormente se les conoce como Ambientes de Ingeniería de Software Centrados/Orientados en el Proceso, y para referirse a ellos se usan comúnmente las siglas en inglés: PSEE, que provienen de Process-Centered Software Engineering Environment.[29]

Los PSEE son ambientes de ingeniería de software cuyos procesos son definidos y modelados claramente por el usuario con el objetivo de incrementar la productividad y la calidad del software. Sus herramientas fueron desarrolladas para ayudar en actividades individuales del proceso de software proporcionando medios para modelar, analizar, mejorar, medir y, cuando es razonable y conveniente, automatizar las actividades de producción de software.

3.2. Conceptos y Terminología del Modelado de Procesos

Cualquier representación de un proceso es un modelo, veamos las siguientes definiciones:

a) Proceso.

El proceso de software queda definido como ...

> "una red parcialmente ordenada de actividades interactuando, realizadas por agentes humanos y soportada por herramientas, que ayuda a producir un producto de software".[30]

---

[28] Modelo que describe los cambios que opera un ambiente de Ingeniería de Software.
[29] De aquí en adelante estos ambientes serán referidos con las siglas PSEE.
[30] Camilo Ocampo, Begoña Albizuri y Pere Botella, op. cit., p.55.

El proceso de software presenta dos componentes:

- Proceso de Producción.

  Involucra todas las actividades, los métodos, las herramientas y las estructuras ocupadas en el proceso de software.

- El Meta-proceso.

  Incluye toda actividad que debe ser realizada para modelar, analizar y apoyar un proceso.

Pocos ambientes soportan modificaciones dinámicas de la definición de un proceso de software durante su ejecución, los PSEE reconocen esta naturaleza modificable.

Una función importante de los PSEE es precisamente dar soporte a la ingeniería de procesos. Esta disciplina se encarga del estudio de las entidades dinámicas que componen al proceso de software, y se encuentra asociada con las actividades de modelado, definición, análisis y simulación. En la ingeniería de software, el objetivo es construir un producto de software o mejorar uno existente. En tanto, en la ingeniería de procesos la meta es desarrollar o mejorar un modelo de procesos.

El proceso de software se adapta a las necesidades del ambiente en el cual se aplica, es decir, debe basarse en la experiencia del equipo o en la disponibilidad de recursos, además —reitero—permite modificaciones aún y cuando el proceso está siendo ejecutado.

b) Etapa del Proceso.

   Acción atómica de un proceso que no tiene una estructura externa visible, es una abstracción básica del proceso. Ejemplo: Compilación de un módulo.

c) Elemento del proceso.

   Agrupa una o múltiples etapas del proceso, puede representarse como un modelo o plantilla parametrizada.

d) Agente o actor.

Entidad que ejecuta una etapa del proceso, puede ser una computadora o un ser humano.

e) Recurso.

Entidad requerida para ejecutar una etapa del proceso, puede ser cualquier actor.

f) Producto.

Es el resultado de una etapa del proceso.

Usando los términos anteriores es posible definir el concepto de modelo de proceso como...

" un conjunto de actividades o etapas llevadas a cabo por agentes, usando recursos para producir un producto".

Pero en sí, un modelo es una representación del proceso conformado por los siguientes elementos claves:

a) Las actividades que deben ser ejecutadas.

b) Los actores que ejecutan las actividades.

c) Los productos que son producidos.

d) Los recursos que son necesitados por una actividad.

- - - Un modelo puede ayudar en el diseño, análisis, automatización, monitoreo y el compartimiento de procesos de software ya que representa un proceso en diferentes niveles de abstracción, y puede ser usado:

- Para entender y comunicar el proceso.
- Como una base para analizar y mejorar el proceso.
- Para administrar el proceso.

Un modelo de procesos puede ser usado como base para su automatización, pues su representación formal hace posible su estudio. En la práctica, las organizaciones han verificado que definiendo los procesos de esta manera se mejora la efectividad.

Los modelos de procesos de software son interpretados y ejecutados en entornos como los PSEE ya que estos se basan en modelos de procesos predefinidos. Los PSEE son construidos como si fuesen componentes independientes que interactúan entre ellos y comparten información.

3.3. Características del PSEE

La principal características de un PSEE es la de representar explícitamente un proceso de software, hecho por el cual ofrece un buen soporte a las actividades de los procesos.

Existen PSEE disponibles tanto para el área académica como para el área comercial, pero no todos los PSEE poseen las mismas características, y aún cuando las comparten, su acceso puede ser diferente. Enseguida se listan algunas características importantes de los PSEE:

1). Definición del Proceso.

Todo PSEE define un proceso que es utilizado por uno o más proyectos. Actualmente los PSEE ofrecen una gama amplia de propuestas para especificar un proceso. Otros PSEE proporcionan una combinación de propuestas, permitiendo la traducción de uno a otro.

Algunos PSEE:

- Presentan notaciones gráficas como los diagramas de estados, los de Entidad-Relación, las redes de Petri, los diagramas de flujo de datos, etc.
- Permiten especificar el proceso mediante tablas textuales o formas.

- Usan un lenguaje de programación para definir un proceso. Entre los paradigmas de lenguajes utilizados tenemos a los de tipo script, los basados en reglas, los triggers en las bases de datos y los lenguajes de programación imperativa.

2). Análisis del Proceso.

Un PSEE determina actividades redundantes en un modelo, el cual puede ser analizado de acuerdo a la estabilidad, la integridad y la disposición de ajustarse a los estándares. Además, el proceso puede ser revisado de acuerdo a la pauta propuesta por un estándar de calidad que especifique un conjunto de actividades llevadas en un orden en particular.

3). Presentación del Proceso.

El proceso de software describe el flujo de actividades de una manera gráfica —ya sea en pantalla o en papel. Algunos PSEE proporcionan vistas múltiples, por ejemplo: la administrativa y la de ingeniería.

4). Simulación del Proceso.

Algunos de los PSEE soportan la simulación computacional de un proceso, evaluando, de esta manera, su utilidad antes del gasto de recursos.

5). Automatización del Proceso.

Por lo regular un proceso de software incluye actividades automatizadas cuya ejecución no requiere de la participación humana, por ejemplo, la acción de notificar al personal afectado acerca de los cambios elaborados a la información de algún producto. Una vez que el proceso ha sido definido en un PSEE y las actividades identificadas por el ingeniero de procesos, el PSEE las automatiza. El alcance de esta automatización dependerá del proceso, de la información del producto almacenada en el PSEE y del formalismo utilizado para definir el proceso.

6). Monitoreo del Proceso.

Un PSEE monitorea la ejecución de un proceso y registra la historia de las actividades llevadas a cabo. Esta información es utilizada para mejorar el desarrollo de procesos

futuros. Algunos de los PSEE proporcionan un monitoreo en tiempo real con resultados gráficos, otros producen información para ser analizada y visualizada con herramientas especiales. Este tipo de resultados puede ser considerado como una medida útil para mejorar la práctica de la ingeniería de software.

7). Apoya los cambios en los procesos.

Con un PSEE, una organización debe ser capaz de cambiar su definición de procesos sin interrumpir el trabajo en la empresa. Entre las modificaciones que apoya el PSEE se tiene la conversión de programación procedural a programación orientada a objetos, el cambio de terminales basadas en caracteres a programación orientada a ventanas, etc.

8). Guía de procesos.

Un PSEE, por lo regular, guía al ingeniero de software indicándole cuáles son las siguientes etapas a realizar, basándose en el proceso modelado y en su estado actual.

El PSEE presenta al usuario:

- Una lista de tareas de acción.
- Los objetos del producto actual y las operaciones de que dispone.
- Un ambiente de programación tradicional.

9). Interfaz de Usuario.

Un PSEE, basado en el proceso de modelado, puede utilizar la interfaz para reducir la cantidad de información presentada al usuario, mostrando únicamente la información necesaria. [31]

El PSEE pone a disposición del usuario las herramientas necesarias para realizar una etapa, de igual forma limita los productos visibles al usuario.

---

[31] Situación que no presenta Unix.

10). Apertura.

Los PSEE operan en ambientes donde tienen el control parcial o total de las herramientas de software disponibles. Algunos PSEE proporcionan facilidades para intercambiar datos entre las herramientas y el PSEE; otros se integran a las ya existentes.

11). Soporte Multiusuario.

Un PSEE comúnmente da soporte a un equipo de trabajo enfocado a un determinado proyecto. El PSEE asegura la concurrencia al proceso y a los datos del producto.

3.4. Arquitectura

Los PSEE son relativamente nuevos por lo que no tienen una arquitectura estándar aceptada, sin embargo, un elemento clave en la misma es el depósito donde se almacenan todos los productos de trabajo.

Enseguida se muestra la arquitectura de un PSEE donde se relacionan los componentes de un sistema:

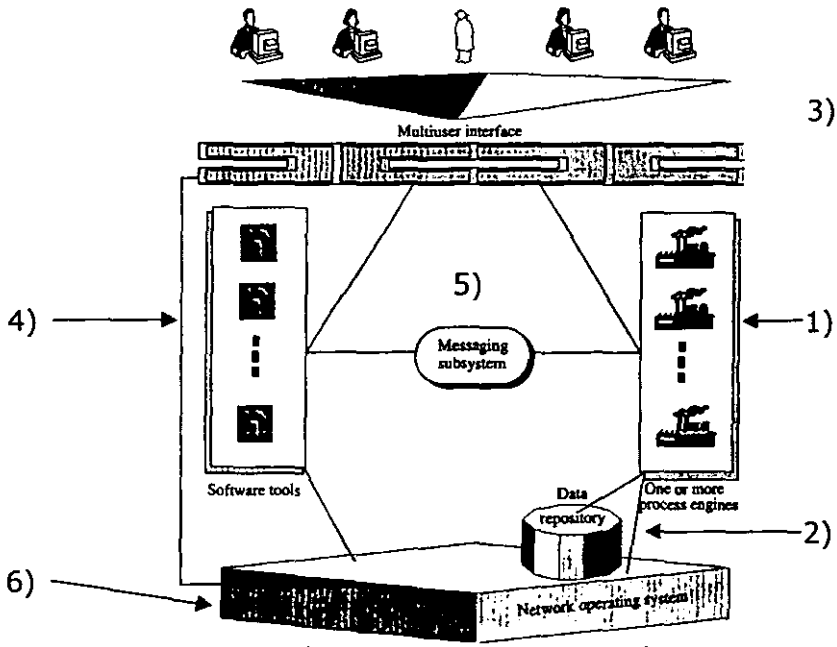


Figura No. 8 Arquitectura genérica de un ambiente de ingeniería de software centrado en el proceso.

Componentes de todo PSEE:

1. Un conjunto de una o más máquinas de procesos, cada una proporcionando sus respectivas normas o estatutos.
2. Un depósito de datos que almacena los procesos y la información de los productos.
3. Una interfaz multiusuario.
4. Un conjunto de herramientas de software que ayuda en las etapas de los procesos.
5. Una infraestructura de comunicación que facilita la comunicación y la coordinación entre las herramientas, las máquinas de procesos y la interfaz de usuario.
6. El sistema operativo de red que proporciona la administración de los recursos de hardware básico mientras varios componentes interfaz usuario facilitan una interfaz multiusuario.

Un factor clave que afecta a muchos de los aspectos de un PSEE, incluyendo su utilidad, es la forma en la que se representa al proceso. Las propuestas pueden dividirse en dos líneas básicas:

- La orientada a la actividad.

  El proceso es definido en cuanto al trabajo que va a ser realizado, como en el modelo de las redes de Petri.

- La orientada al producto.

  El proceso queda definido en función de los productos de trabajo que son producidos por el proceso.

Algunos PSEE combinan las características formales orientadas a la actividad y al producto.

3.5. Problemas fundamentales de diseño

En el diseño de un PSEE es necesario evaluar los siguientes dos puntos que no son específicos del PSEE, pues cualquier ambiente de ingeniería de software debe tomarlos en cuenta:

a) Determinar la existencia de un almacén de datos propio o de un depósito común para cada una de las herramientas.

b) Especificar la creación de un almacén para contener tanto los datos del producto como los datos del proceso, o dos almacenes para mantenerlos separados.

A continuación se mencionan consideraciones específicas de todo PSEE:[32]

c) Para representar un proceso dentro del PSEE, primero es necesario seleccionar el lenguaje, ya sea imperativo, basado en reglas, orientado a objetos, o una combinación de todos ellos.

---

[32] La numeración de los índices es a propósito, ya que listan consideraciones propias de todo PSEE.

d) Seleccionar una buena representación de la máquina de procesos.

e) Identificar las etapas del proceso que son preprogramadas por las herramientas dentro del ambiente y reconocer de esta preprogramación cuál puede representarse explícitamente y cuál puede ser modificada por el usuario.

Otros elementos de diseño son:

a) La integración de los datos.

Las herramientas invocadas durante la ejecución del proceso, se coordinan para procesar la información, por ejemplo, mediante el uso de un editor se modifica uno de los módulos de un programa, posteriormente el compilador da acceso a la información modificada, procesándola.

b) La integración del control.

Cuando múltiples herramientas son empleadas durante la ejecución de un proceso, sus llamadas necesitan ser coordinadas. Por ejemplo, en un ambiente Unix, para recompilar un módulo fuente alterado, un compilador necesita ser invocado después de que la modificación ha sido almacenada en un archivo por el editor. Los PSEE proporcionan un control integrado e inteligente para las herramientas debido al conocimiento que tienen acerca del proceso.

c) Administración de la información del producto.

Algunos problemas crecen cuando se trata de administrar la información del producto dentro de un PSEE pues es necesario determinar:

- El tamaño que ocupará la información del producto.
- La información del producto tendrá el nivel de una función, un módulo o de un sistema.
- La forma en que se deben administrar los diversos niveles de la información.
- El tipo de organización en que se debe presentar la información del producto, ejemplo: lineal o jerarquizada, entre otros.

- La forma en que se almacenará la información del producto dentro del PSEE, directamente o por referencia.
- El sistema operativo de red y la base de datos a usar, pues ambas tecnologías tienen un impacto significativo en el oficio de administrar la información del producto dentro del PSEE.

d) La configuración de la base de datos.

El PSEE usa características propias de los sistemas de administración de bases de datos, por lo que casi siempre parece un DBMS tradicional.

Algunos de los problemas relacionados a la configuración de los sistemas que administran los datos del PSEE son:

- Seleccionar entre una base de datos modificable o una base de datos permanente.
- La base de datos debe estar centralizada o distribuida.
- Determinar el tipo de consulta que debe proporcionar la base de datos.
- Especificar el control de concurrencia que proporcionará la base datos.
- La base de datos a utilizar es de uso general o individual.

e) Personalizar el proceso.

Determinar la información que puede ser codificada en el ambiente utilizando sus propias herramientas, además de especificar la información que será visible y modificable por el usuario.

f) Soporte multiusuario.

Los ambientes PSEE apoyan el trabajo cooperativo de los procesos de software. Los mecanismos que el ambiente proporciona para el soporte multiusuario varían significativamente, van desde el apoyo activo a todo un proyecto utilizando el sistema CSCW hasta el soporte a actividades de usuarios individuales. Como se sabe, los problemas crecen cuando se trata de soporte multiusuario, pues hay que considerar cómo y cuándo notificar a un usuario de los cambios en la información.

g) Configurar la máquina de normas o de estatutos.

Los ambientes PSEE apoyan las normas del proceso facilitando la secuencia, la aplicación, y la automatización de las actividades del proceso, pero un problema propio de diseño es configurar la máquina de normas.

h) La interfaz usuario.

La interfaz usuario está preparada para diferentes tipos de usuario de los ambientes PSEE. La interfaz presenta:

- Una vista del proceso desde el cual los usuarios pueden seleccionar sus tareas a ejecutar.
- Los objetos del producto siendo manipulados por el proceso.
- El alcance de las actividades realizadas en aquellos objetos.

3.6. Áreas que apoyan los ambientes PSEE

El concepto de proceso permite a los ambientes PSEE apoyar y relacionar las siguientes tres funciones básicas de toda organización de desarrollo de software:

a) Ingeniería de proceso.

La meta de la ingeniería de procesos es definir, evaluar y mantener los modelos de procesos. El ingeniero de procesos, es quien define el modelo de procesos en un lenguaje útil para la máquina de procesos. Mientras tanto, los ambientes PSEE proporcionan la definición y el análisis de procesos.

La ingeniería de procesos involucra un conjunto de actividades que se ocupan del proceso durante todo el ciclo de vida. Entre las actividades que son consideradas primarias se cuentan las siguientes:

- La representación del modelo de procesos.

  Se debe tener una representación del modelo ya sea formal o informal, ya que el análisis y el soporte automático son posibles únicamente para la representación formal.

- El análisis del proceso.

  Una vez que el modelo ha sido descrito, puede ser analizado para detectar si existe redundancia en las etapas del proceso.

- La instancia del proceso.

  Un proceso es "instanciado" cuando se le especifican variables propias del modelo de procesos, las variables pueden ser los productos, los recursos y los agentes.

- Señalamiento de las normas del proceso.

  Se refiere a la ejecución de un proceso llevado a cabo por un ambiente computacional o por un ser humano, donde la ejecución del proceso es monitoreada.

b) Ingeniería de software.

Para las funciones de la ingeniería de software, los ambientes PSEE dan soporte antes, durante y después de la ejecución de un proceso. Proporcionan, además, información del estado actual de los procesos, produciendo y accesando la información localizada en el depósito de datos.

Los ambientes PSEE consideran a la ingeniería de software como la ejecución de un proceso definido claramente.

c) Administración de proyectos.

El objetivo de la administración de proyectos es asegurar el seguimiento de un proceso en particular, ya que entre sus funciones se cuentan la coordinación y el

monitoreo de las actividades de la ingeniería de software.

Los ambientes PSEE apoyan la administración del proyecto proporcionando estados y funciones de monitoreo, presentan, además, un número determinado de vistas del proceso y de los datos del producto para ayudar al entendimiento del proyecto.

En los ambientes PSEE se selecciona un modelo de procesos, generado por la ingeniería de procesos, utilizado para monitorear la ejecución del proyecto.

Los ambientes PSEE usan al proceso como el tema central de todas las actividades en una organización de desarrollo de software, la figura siguiente divide estas actividades de acuerdo con la función que apoya cada una. Las tres funciones mostradas en la Figura No. 9 son comúnmente soportadas por herramientas independientes, por ejemplo, el software de calendarización lo proporciona la administración de proyectos, el control de calidad lo soporta la ingeniería de procesos, y la administración de configuración es realizada por la ingeniería de software.

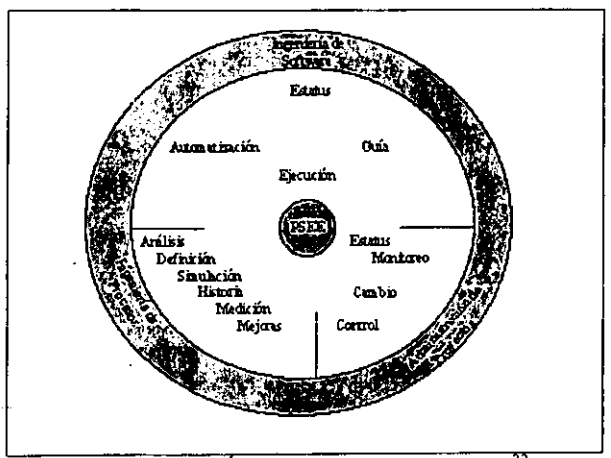


Figura. No. 9 Áreas que apoyan los PSEE.[33]

[33] Pankaj K. Garg y Mehdi Jazayeri (eds.), Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Los Alamitos, CA, 1996, p.8

La Figura No. 10, muestra a la ingeniería de procesos como la tecnología que desarrolla los modelos de procesos y que son usados por la administración de proyectos para crear un proceso en particular dentro de un proyecto determinado. Posteriormente señala a la ingeniería de software siguiendo las especificaciones proporcionadas por el proceso para desarrollar un producto.

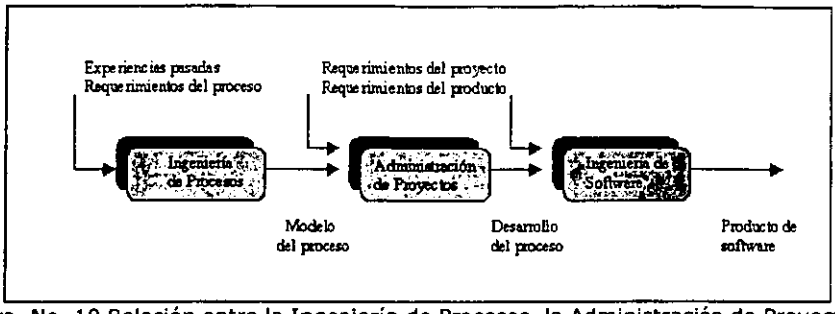


Figura. No. 10 Relación entre la Ingeniería de Procesos, la Administración de Proyectos y la Ingeniería de Software.[34]

Los ambientes centrados en el proceso integran:

a) Los requerimientos del producto, punto de interés para la ingeniería de software.
b) Los requerimientos del proceso, parte central de la administración de proyectos y de la ingeniería de procesos.

El proceso al que se hace mención es el de un proceso interactivo, útil para modificar el modelo existente, donde todo cambio es registrado por los ambientes PSEE. Además de esta función, los PSEE tienen la capacidad de almacenar, analizar, simular y modificar modelos de procesos de acuerdo a instancias apropiadas especificadas por distintos proyectos.

[34] Pankaj K. Garg y Mehdi Jazayeri, op. cit., p.9.

3.7. Aplicaciones

Los PSEE pueden encontrar aplicaciones en algunas de estas áreas:

a) Trabajo cooperativo apoyado por la computadora (CSCW).

Los productos Groupware son usados en ambientes de trabajo para apoyar la coordinación y la comunicación de los grupos de trabajo distribuidos, por ejemplo: el correo electrónico y los boletines —precursores de tales sistemas— son regulados por ambientes PSEE.

b) Tecnología de base de datos.

Los ambientes de ingeniería de software almacenan y administran el acceso de objetos de diferentes tipos, entre ellos: la información del propietario de los documentos, el código fuente, el código objeto, etc. Los ambientes PSEE coordinan estos procesos y las relaciones entre ellos. Las bases de datos tradicionales no se adaptan a las necesidades de los ambientes de software que requieren objetos con diferentes patrones de acceso. En la mayoría de los casos, los PSEE construyen sistemas de bases de datos especializados.

c) Automatización en las organizaciones.

La automatización de oficinas y la organización computacional combina facetas manejadas por los ambientes PSEE. La automatización se inicia con las rutinarias tareas administrativas que pueden ser entendidas e integradas. El alcance de los PSEE crece, ya que pasa del soporte de desarrollo de software al de procesos de negocios.

d) Los sistemas administrativos de seguimiento del trabajo (Workflow).

Ofrecen soporte especialmente a los procesos organizacionales. La unidad básica de cualquier empresa son los procesos de trabajo, los cuales surgen a partir de la comunicación entre un cliente y un agente. Un modelo de procesos de trabajo se representa mediante una red, conectando usuarios y agentes como nodos, mientras que el flujo de trabajo se simboliza con arcos en la red. El sistema de administración de flujo de trabajo soporta la construcción, el análisis, la automatización de los

procesos de flujo de trabajo. Muchas de estas medidas son parecidas a aquellas que llevan a cabo los PSEE.

3.8. Ventajas y Desventajas

Los ambientes PSEE brindan apoyo al entorno de software como sigue:

- Apoyan la definición y el uso de un proceso en el desarrollo de software. En un soporte mínimo el ambiente apoya la definición de un proceso y su monitoreo para detectar desviaciones del proceso.
- Consideran al proceso de software como una entidad dinámica con su propio ciclo de vida.

Los PSEE reconocen las necesidades de:

- La ingeniería de procesos para definir y evaluar los modelos de procesos.
- La ingeniería de software para seguir un proceso prescrito.
- La administración de proyectos para monitorear, modificar posiblemente, la ejecución del proceso.

La diferencia entre un ambiente centrado en el proceso y los ambientes más convencionales estriba en la cantidad de apoyo que ofrecen a las distintas fases del ciclo de vida del proceso de software.

Los ambientes PSEE proporcionan el siguiente soporte dentro de una organización (Ventajas):

a) Regulan los requerimientos.

Los PSEE producen fácilmente la documentación del modelo de procesos utilizando estándares de calidad, por ejemplo, ISO 9000.

b) Coordinan la efectividad del equipo.

Los PSEE dan a conocer el estado actual de cada una de las actividades que componen el proyecto.

c) Disciplina para mejorar la productividad y la creatividad.

Los PSEE informan sobre la siguiente actividad por hacer. Además, proporcionan soporte no solamente a las funciones de desarrollo de software sino también a las asociadas con la administración y el control de calidad. Los PSEE se basan en una definición explícita del proceso de desarrollo de software que cruza los límites funcionales de las siguientes áreas:

- De control de calidad al definir y evaluar un proceso.
- Administrativas al proporcionar información acerca del estado del proyecto.
- De desarrollo al automatizar la ejecución del proceso.

Desventajas.

El uso de los ambientes PSEE implica un cambio en la mentalidad de la gente. Cambio que trae consigo cierta resistencia, sobretodo al monitoreo automático que se hace sobre el usuario del ambiente, quien piensa que los resultados pueden ser usados en su contra. Sin embargo, existe un acuerdo general que limita el uso de los datos del sistema sólo para propósitos estadísticos y no para evaluaciones individuales.

Los datos son útiles para medir la productividad del equipo; pero pueden también ser usados para capturar el número exacto de compilaciones que hace un ingeniero o la cantidad exacta de tiempo que le toma a un ingeniero producir un módulo en particular.

3.9. Estado del Arte

Los PSEE son sistemas que introducen la automatización en actividades humanas que no son rutinarias, capturando procesos esenciales. Los PSEE apoyan la definición, adquisición y creación de procesos para ambientes particulares; también monitorean la corrida de los procesos conservando su comportamiento para utilizarlo posteriormente en la construcción de modelos semejantes de procesos.

Las tareas que son particularmente útiles en la dirección de todo proyecto y que son llevadas a cabo por los ambientes de ingeniería de software centrados en el proceso (PSEE) son:

- La supervisión del desarrollo, verificando que la gente involucrada esté siguiendo un proceso en particular.
- La planeación de las actividades de desarrollo futuras.
- La automatización de las partes del proceso.

Esta tecnología ayuda a soportar el proceso de producción de software facilitando los medios para planear, analizar, mejorar, medir, y siempre que sea razonable y conveniente, automatizar las actividades de producción de software.

3.10. Futuro del PSEE

a) En la práctica.

Existen pocas dudas de que los PSEE llegarán a formar parte de la fábrica de ingeniería de software en los entornos de trabajo. Algunos de los problemas que determinarán el éxito o el fracaso de estos ambientes son los siguientes:

- Estandarización.

Los estándares deben aplicarse tanto a la descripción de procesos como a la operación de los PSEE, esto se debe a la necesidad de intercambiar los modelos de procesos.

- Integración.

El éxito de los PSEE llegará cuando puedan integrar completamente a toda la organización, utilizando procesos y herramientas.

- Sistemas abiertos.

Un requerimiento necesario para los PSEE es abrirse a nuevas herramientas para incrementar su funcionalidad. Esto ocurre cuando el usuario utiliza un PSEE en particular al cual añade su propio método de análisis, monitoreo y herramientas.

- Interoperabilidad.

Una organización en particular puede seleccionar diferentes PSEE para usar en algunos departamentos o proyectos, por lo que se necesitará tener varios PSEE funcionando.

- Ambiente de ventana y PSEE.

Los PSEE tendrán que aparecer como una evolución de los ambientes existentes de los usuarios, porque los entornos actuales son lentos para soportar procesos, y todavía no se encuentran centrados en el proceso.[35]

b) En la investigación.

La generación actual de los PSEE no utiliza completamente su potencial, por ejemplo, el modificar un modelo de procesos mientras el proceso se está ejecutando no es llevado a cabo correctamente por algunos de los PSEE. De igual forma, el hecho de recolectar la información del proceso no ha tenido éxito aún. La investigación actual está enfocada a tales problemas.

Uno de los desafíos en la investigación es el desarrollo inicial de los procesos, ya que puede tratarse de un proceso seleccionado desde una biblioteca, o basado en historias de procesos ejecutados.

La investigación del PSEE actualmente está dirigida a resolver las dificultades del modelado inicial del proceso y de su mantenimiento, cuando se alcance esta meta, es previsible la expansión de su uso.

[35] Microsoft OLE y DDE son herramientas individuales utilizadas para compartir datos a través de Windows, de tal forma que habilitan herramientas para interactuar con otras.

Otro problema desafiante para el PSEE, con respecto a la investigación, es el diseño apropiado de las interfaces multiusuarios, para resolverlos la investigación se está enfocando a las siguientes dos áreas:

- Orientar el modelo de procesos hacia la meta y a la interfaz usuario.
- Ver la interfaz de usuarios desde el modelo de procesos.

Se espera que la influencia del PSEE llegue a las organizaciones comunes cuyo objetivo es el mejoramiento del proceso continuo. Será importante entonces que el PSEE soporte el ciclo de vida completo de la ingeniería de proceso del software, incluyendo la adquisición del proceso, el modelado, el análisis estático y dinámico, la ejecución, el monitoreo, la retroalimentación, y la evaluación.

CAPÍTULO 4

REVISIÓN DE CARACTERÍSTICAS TEÓRICAS EN DOS AMBIENTES COMERCIALES

4. Revisión de características teóricas en dos ambientes comerciales

Como se mencionó en la introducción de este trabajo, del marco teórico del CASE, I-CASE y PSEE, se seleccionaron cuatro de las características básicas para ser revisadas en dos de los productos que actualmente compiten en el mercado, uno de ellos es el RUP (Rational Unified Process) de la compañía estadunidense RATIONAL y el otro, Norma Control, herramienta netamente mexicana desarrollada por la empresa Jónima. Los elementos evaluados en los dos ambientes de desarrollo forman parte de la arquitectura de todo PSEE, véase página 42:

a) La representación de los procesos.

b) El depósito de datos.

c) Interfaz usuario.

d) Conjunto de herramientas.

Posteriormente, al terminar cada apartado se hace un resumen comparativo de los puntos revisados en ambos productos. Para terminar, se agrega una tabla que incluye,

de acuerdo a las especificaciones del punto 3.3, las características a considerar en todo ambiente PSEE evaluándolas en los productos ya mencionados. Tabla que puede servir para revisiones posteriores.

Antes de esta evaluación se da una breve explicación de las empresas RATIONAL y JÓNIMA.

4.1. Empresas

A) Rational

Rational es una compañía de desarrollo de software orientada al apoyo de las organizaciones en la creación de software propio con calidad y rapidez,[36] combinando herramientas y servicios con la práctica de la ingeniería de software.

Rational se identifica por ser la compañía que establece un estándar para unificar los lenguajes de modelado, creando el lenguaje de modelo unificado (UML). Otra característica sobresaliente de Rational, es conocida con el nombre de "el desarrollo electrónico", que consiste en desarrollar software de alta calidad para Internet a la velocidad del comercio electrónico. Rational se vale de sus herramientas, ya que con ellas es posible construir la propia red de software, administrar el proyecto, probarlo, y estar seguro de que trabajará correctamente en línea. Uno de los productos Rational, el Rational Unified Process (RUP) es utilizado para revisar los elementos ya mencionados. Enseguida se incluye una breve explicación del producto.

[36] De acuerdo a la propaganda misma de Rational, el uso de esta tecnología permite hacer el trabajo 50% más rápido de como se haría normalmente.

RUP

Es un proceso de ingeniería de software que corre en la Web y cuyo objetivo es mejorar la productividad del equipo de trabajo, ocupando prácticas como lo son el modelado de lenguaje unificado (UML); y guías sencillas a base de plantillas en áreas del modelado de negocios y la arquitectura de los procesos en el Web.

B) Jónima

Compañía mexicana dedicada a dar soporte de todo tipo a la plataforma conocida como Lotus Notes.

Jónima ofrece resolver problemas e innovar en los procesos de negocios ofreciendo los servicios siguientes:

- Desarrollo de sistemas.

  Lleva a cabo el análisis, diseño y programación necesarios para la entrega de soluciones.

- Comunicación interactiva.

  Se especializa en el desarrollo de estrategias de comunicación en medios electrónicos, combinando un alto diseño creativo y tecnologías de punta.

- Consultoría tecnológica.

  Jónima brinda asesoría en cuanto a la adquisición de Hardware y Software.

- Investigación, desarrollo y calidad.

  Jónima crea productos y servicios que integra a la solución de sus clientes, asegurando la calidad en todo trabajo prestado. Uno de sus productos, Norma Control ISO2000, es el material que se utiliza para revisar los elementos ya mencionados. Enseguida se incluye una breve explicación del producto.

Norma Control ISO2000

Norma Control lleva un registro eficiente de toda la documentación de una empresa, incluyendo la generada por el departamento de Ingeniería de Sistemas. Norma Control no genera software, pero si fortalece las mismas áreas que un PSEE: el trabajo cooperativo, la comunicación interactiva y la calidad total.

Norma Control aprovecha las facilidades del correo electrónico para comunicar: notificaciones, flujos de trabajo, documentos, etc., para que todas y cada una de las operaciones llevadas a cabo en la empresa se encuentren documentadas.

4.2. La representación de los procesos

A) RUP

Rational utiliza una instancia del proceso unificado, el Rational Unified Process, identificado con las siglas RUP, y basado en las siguientes prácticas:

1a) Un desarrollo de software iterativo.

2a) Administración de los requerimientos.

3a) El uso de una arquitectura basada en componentes.

4a) El modelado visual del software.

5a) La verificación constante de la calidad del software.

6a) El control de cambios en el software.

Estas seis prácticas aplicadas por RUP, son parte de una cultura de desarrollo de software enfocadas a guiar a cada integrante de un equipo de trabajo.

A continuación se describen, a grandes rasgos, cada una de ellas:

1a. práctica) Desarrollo de software iterativo.

RUP utiliza el desarrollo iterativo e incremental sustituyendo al método clásico de cascada:

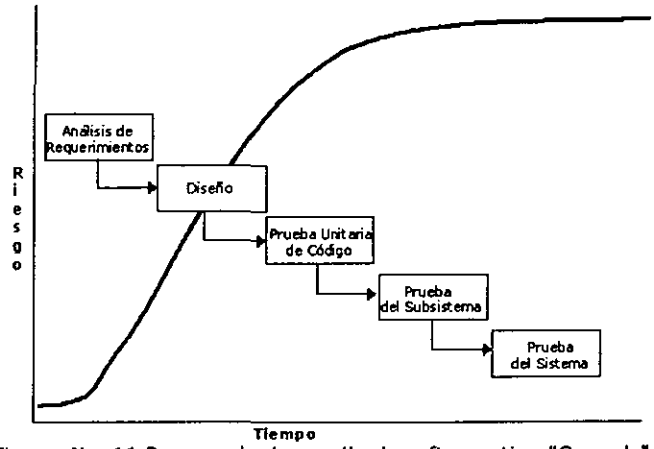


Figura. No. 11 Proceso de desarrollo de software tipo "Cascada".[37]

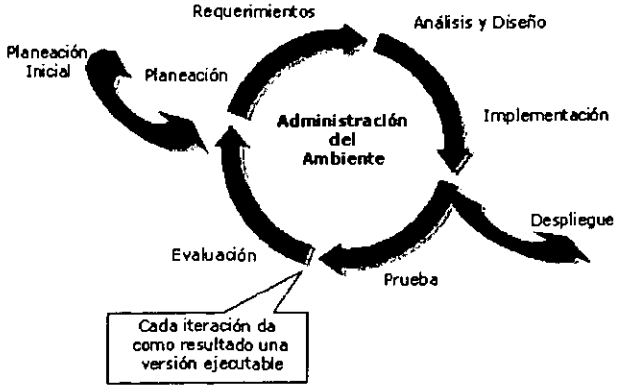


Figura. No. 12 Proceso incremental e iterativo.[38]

[37] Philippe Kruchten, The Rational Unified Process an Introduccion, segunda edición, Addison Wesley, 2000, p.6.
[38] Ibid., p.7

El proceso incremental e iterativo permite la división de un proyecto en miniproyectos. Para llevar a cabo un miniproyecto se necesita de una iteración, llegándose a tener tantas iteraciones como miniproyectos existan. Ahora bien, cada iteración terminada representa un grado de avance con respecto al proyecto total y un incremento en la construcción del producto a elaborar, de esto se deriva el concepto incremental. Con cada iteración se obtiene una versión del producto, que incluye código ejecutable y la documentación del proceso.

Todo miniproyecto tiene planeación propia, por lo que cada iteración cubre las etapas conocidas de: Requerimientos, Análisis, Diseño, Implementación y Prueba. Posteriormente estos miniproyectos o iteraciones se escogen y reparten —de acuerdo a la meta trazada— en las cuatro fases que forman "el ciclo de vida de todo sistema",[39] ver Fig. No. 13:

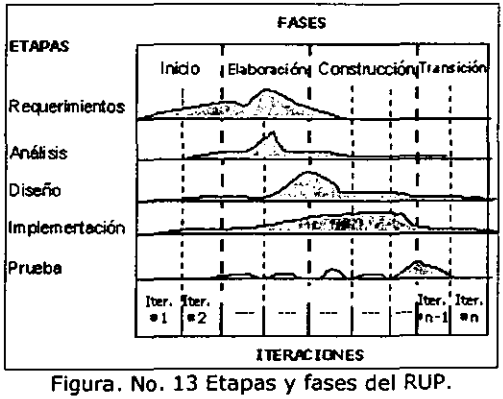


Figura. No. 13 Etapas y fases del RUP.

1a. fase) Inicio.

En esta fase, además de identificarse los riesgos más importantes, se detalla la fase de elaboración, haciendo un estimado del proyecto en tiempo y costo.

[39] Ivar Jacobson, Grady Booch y James Rumbaugh, The Unified Software Development Process, Addison-Wesley, Massachusetts, 1999, p. 8.

2a. fase) Elaboración.

Durante esta fase se diseña la arquitectura del sistema. Al mismo tiempo, el Administrador del proyecto especifica las actividades y los recursos necesarios para llevar a buen término el proyecto.

3a. fase) Construcción.

Esta fase corresponde al desarrollo de software, etapa que utiliza la mayor parte de los recursos solicitados.

4a. fase) Transición.

En esta fase se obtiene la versión beta del producto, la cual es probada y corregida antes de su entrega.

La fase de Transición incluye actividades posteriores a la liberación del sistema, por ejemplo: la capacitación al cliente, la asistencia en línea y el mantenimiento al sistema.

2a. práctica) Administración de los requerimientos.

RUP es un proceso dinámico ya que acepta y maneja cambios constantes en los requerimientos durante el ciclo de vida de un proyecto de software. Como estos cambios son considerados en las iteraciones, el plan inicial no se afecta.

La administración de los requerimientos comprende las siguientes tres actividades:

- Arranque (captura de los requerimientos).
- Organización de los mismos.
- Su documentación.

3a. práctica) Uso de la arquitectura basada en componentes.

El desarrollo de la arquitectura del RUP se basa en el uso de cuatro componentes (CBD=Component Based Development):

- Modelo de componentes objetos de Microsoft (COM).
- El grupo administrador de Objetos (OMG).
- CORBA (Common Object Request Broker Architecture).
- JavaBeans (EJB) de Sun Microsystems.

Lo cual presenta la ventaja de la reutilización.

Un hecho que hay que destacar, es que con cada iteración se produce una arquitectura ejecutable, la cual es medida, probada y evaluada de acuerdo a los requerimientos del sistema.

4a. práctica) Modelado visual del software.

Un modelo es la representación simplificada de la realidad que describe completamente un sistema. Para modelar, RUP utiliza el lenguaje unificado (UML), herramienta gráfica que ayuda al equipo de desarrollo a visualizar, especificar, construir y documentar adecuadamente la estructura y el comportamiento de la arquitectura del sistema.

5a. práctica) Verificación constante de la calidad del software.

RUP propone la evaluación constante —por medio de las iteraciones establecidas— de los elementos que conforman la calidad de un sistema:

- El funcionamiento.

  Implica la revisión y prueba de cada uno de los escenarios y de las actividades involucradas en la construcción del sistema.

- Fiabilidad.

  RUP evalúa hechos concretos a través de escenarios creados, no se enfoca a documentos impresos.

- Desempeño de la aplicación.

  Al revisar objetivos, RUP localiza inconsistencias dentro de los requerimientos en el diseño y la implementación.

- Desempeño del sistema.

  RUP evalúa en cada iteración la ejecución del sistema, identificando errores para su pronta corrección.

6a. práctica) Control de cambios en el software.

Rational maneja los cambios de cada una de las iteraciones y las versiones que de ellas resulten.

Una de las características que identifican a los PSEE, presente en el RUP, es el manejo de procesos dinámicos. Este tipo de procesos permite cambios programados en las iteraciones. Por otra parte, la orientación de los procesos, que conforman el RUP, están orientados hacia las actividades y el producto; puesto que cada iteración implica una serie de actividades a realizar para incrementalmente construir la arquitectura del producto. Además, los procesos se definen de acuerdo a las actividades del rol, por lo que en RUP existen patrones establecidos según el rol y la actividad a efectuar, véase páginas 84 y 85.

B) Norma Control

Norma Control ISO9000 —de aquí en adelante tan solo Norma Control—, obtuvo el primer lugar en 1997 por ser el mejor desarrollo bajo la plataforma Lotus Notes.

Norma Control tiene como objetivo primordial manejar de una manera sencilla los documentos normativos de una empresa, documentos que en un momento dado determinarán la calidad de los procesos, apoyando con esto la certificación del negocio.

Norma Control facilita el trabajo a los administradores ya que permite la planeación, el monitoreo y el control automatizado del proceso de generación, autorización, publicación, distribución y consulta de los documentos corporativos.

Los proceso que maneja Norma Control son de dos tipos:

1o.) Lineales (L)

Procesos que deben realizarse uno después de otro antes de iniciar la operación de Norma Control.

2o.) Paralelos (P)

Procesos que pueden efectuarse a la vez con otros y en cualquier momento.

Los procesos que componen a Norma Control son los siguientes:

a) Configuración de Norma Control (L).

Proceso donde el administrador establece:

- Las normas de calidad.
- El flujo de trabajo, el seguimiento para que un documento en particular sea autorizado.
- Los tipos de documentos, entre ellos los manuales, procedimientos, catálogos de calidad, etc.
- Los distintos departamentos involucrados, por ejemplo calidad, sistemas, etc.
- Las localidades, ubicación de las sucursales participantes, por ejemplo: D.F., Nuevo León, etc.

b) Ciclo de gestión del documento (P).

Ciclo que comprende los siguientes subprocesos:

b1. Creación y modificación del documento.

b2. Revisión y aprobación del documento para su rechazo/autorización.

b3. Publicación del documento autorizado.

b4. Consulta y revisión del documento.

b5. Control de Impresiones.

Los procesos y subprocesos son realizados por uno de los siguientes roles:

Administrador
Autor del documento
Autorizador del documento
Publicador
Lector del documento
Impresor (quien autoriza la impresión del documento)

Enseguida se muestra el ciclo de gestión de Norma Control:

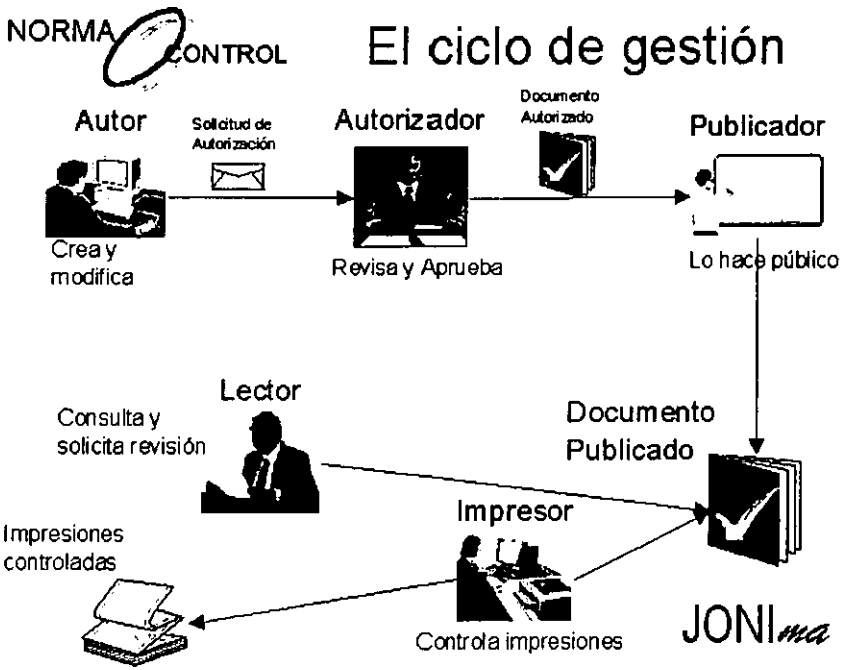


Figura. No. 14 Ciclo de gestión de Norma Control.

Con Norma Control, los administradores documentan todas las actividades correspondientes a su cargo, de esta manera monitorean y revisan el estado actual de cada una. Y como toda actividad se basa en una normatividad, de acuerdo a los estándares de calidad establecidos por la empresa, la calidad se encuentra regulada.

El proceso que genera Norma Control incluye de una manera sencilla la función de la administración del proyecto, tarea que en RUP se encuentra totalmente detallada, por lo que el uso de RUP requiere de gran disciplina.

COMPARACIÓN ENTRE PRODUCTOS:

No.	Características evaluadas	RUP	NC	Observaciones
1	Utiliza una notación o representación formal para definir los procesos	√	x	RUP utiliza UML
2	Los procesos se presentan visualmente	√	x	NC no maneja parte gráfica
3	Se verifica la calidad de los procesos	√	√	Ambos productos utilizan estándares de calidad
4	Utiliza prácticas que guían al desarrollador a realizar su trabajo	√	√	Las actividades a llevar a cabo se establecen en la definición de los procesos
5	Maneja procesos dinámicos	√	x	Una vez que en NC se aprueban los documentos, es difícil su modificación
6	Se da el reuso de los procesos	√	√	
7	Localiza inconsistencia dentro del procesos	√	√	La constante revisión de la calidad, las evita
8	Tiene control sobre los cambios y sus versiones	√	√	
9	Permite el monitoreo de los procesos	√	√	

Notación:

NC = Norma Control

√ = Cumple con la característica

x = No cumple con la característica

4.3. Base de Datos

A) RUP

La herramienta de Rational utilizada para almacenar los proyectos en una base de datos, es el Rational RequisitePro, véase Figura No. 15:

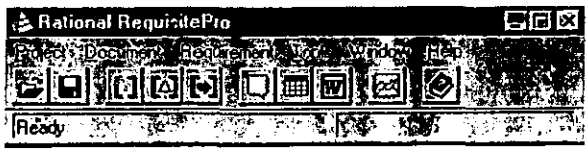


Figura No. 15 Ventana inicial del Rational RequisitePro.

RequisitePro, permite crear un nuevo proyecto o modificar uno existente. Para su creación, se muestra la ventana localizada en la Figura No. 16:

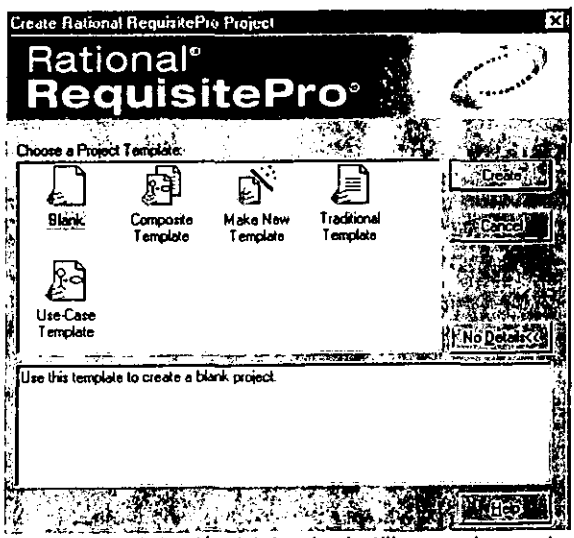
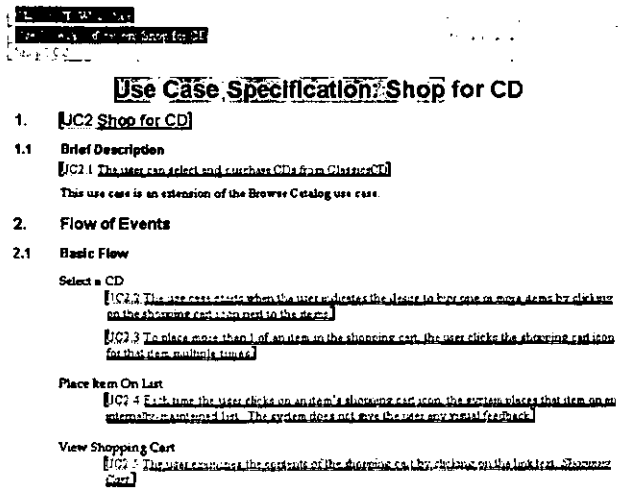


Figura No. 16 Selección del tipo de plantilla para el proyecto.

Como cada proyecto tiene asociado un tipo de documento, es necesario escoger la plantilla (template) que lo contenga:

Plantillas	Uso
Blank	Con esta plantilla se crea un proyecto en blanco.
Composite Template	Utiliza la especificación tradicional para documentar los requerimientos del modelado de los casos de uso incluidos en el proyecto.
Make New Template	Crea una plantilla nueva para un proyecto RequisitePro.
Traditional Template	Plantilla donde se especifican los requerimientos de una manera tradicional.
Use-Case Template	Plantilla ideal para usuarios que integran los casos de uso del RequisitePro con Rational Rose y ClearQuest.

Enseguida se muestra, como ejemplo, el documento base creado a partir de la plantilla de casos de uso, actualizado con información:



Use Case Specification: Shop for CD

1. UC2 Shop for CD

1.1 Brief Description

UC2.1 The user can select and purchase CDs from ClassicsCD.

This use case is an extension of the Browse Catalog use case.

2. Flow of Events

2.1 Basic Flow

Select a CD

UC2.2 The use case starts when the user indicates the desire to buy one or more items by clicking on the shopping cart icon next to the item.

UC2.3 To place more than 1 of an item in the shopping cart, the user clicks the shopping cart icon for that item multiple times.

Place Item On List

UC2.4 Each time the user clicks on an item's shopping cart icon, the system places that item on an internally maintained list. The system does not give the user any visual feedback.

View Shopping Cart

UC2.5 The user examines the contents of the shopping cart by clicking on the link text, Shopping Cart.

Figura No. 17 Ejemplo de documento creado utilizando la plantilla de casos de uso.

Cuando se crea un proyecto basado en alguna de las plantillas los tipos de documentos, los requerimientos, los atributos, la información de seguridad, y las vistas se coplan al nuevo proyecto. Las plantillas son creadas y accesadas con Microsoft Word.

Los tipos de documentos ofrecen un estilo propio para el proyecto, ya que incluyen:

- Distintos formatos para encabezados y párrafos.
- Temas diferentes.
- Estilos para encabezado y párrafos.
- Diferentes tipo de requerimientos.
- Convenciones particulares para estructurar el documento.

Para almacenar los requerimientos de un proyecto, hay que decidir el tipo de base de datos a utilizar. RequisitePro soporta actualmente:

Microsoft Access

Oracle y

Microsoft SQL Server

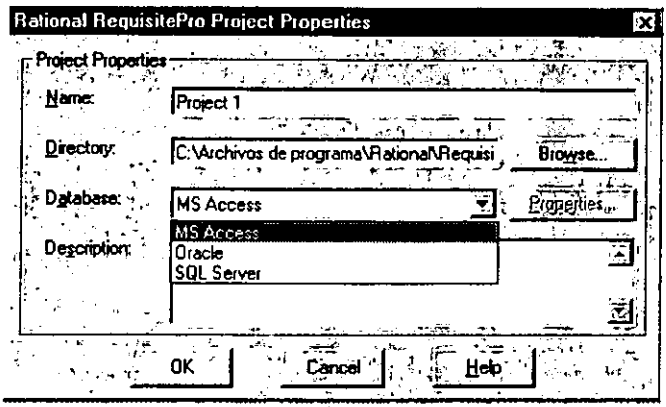


Figura No. 18 Selección del tipo de base de datos.

Se recomienda Microsoft Access para grupos pequeños con un máximo de diez usuarios. Ahora si se va a manejar un gran número de requerimientos, es recomendable utilizar SQL Server u Oracle.

A continuación, debe establecerse el tipo de requerimiento de la especificación seleccionada:

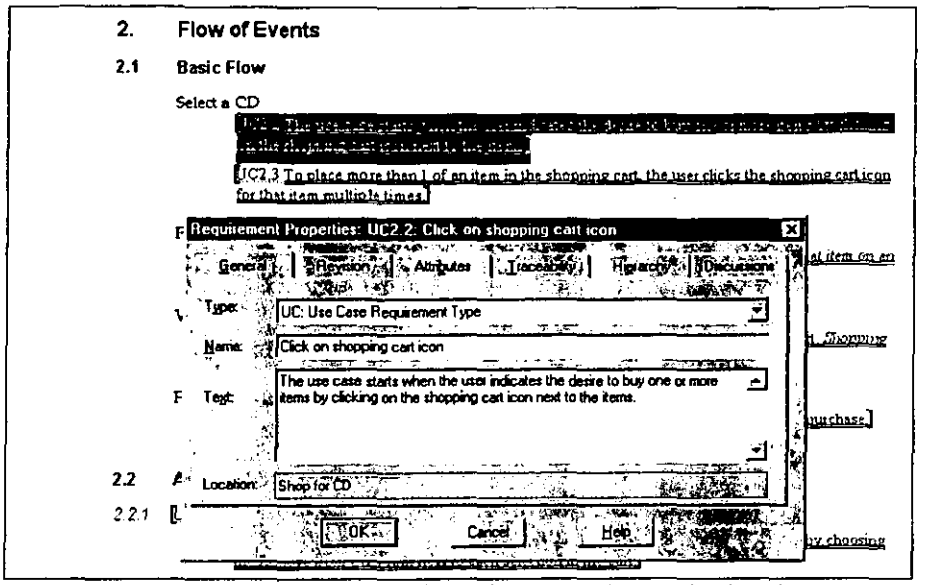


Figura No. 19 Estableciendo el tipo de requerimiento seleccionado.

RequisitePro sugiere mantener entre cinco y ocho tipos de requerimientos para un manejo adecuado. Precisamente el índice de la base de datos del proyecto se conforma por:

el prefijo del tipo, en el ejemplo, "UC" (casos de uso)

+

un número generado automáticamente de acuerdo al tipo de requerimiento

Como puede verse en la Figura No. 19, los requerimientos tienen asociado:

a) Un tipo de requerimiento determinado por su prefijo.

b) Un nombre.

c) Un conjunto de propiedades que los identifican, como son:

c1. La historia de las revisiones del requerimiento (Revisión):

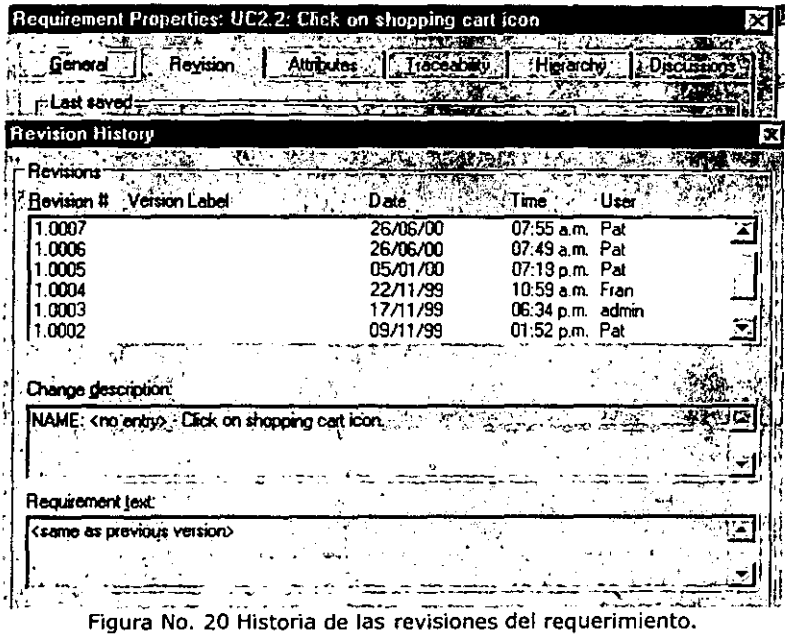


Figura No. 20 Historia de las revisiones del requerimiento.

c2. Atributos (Attributes):

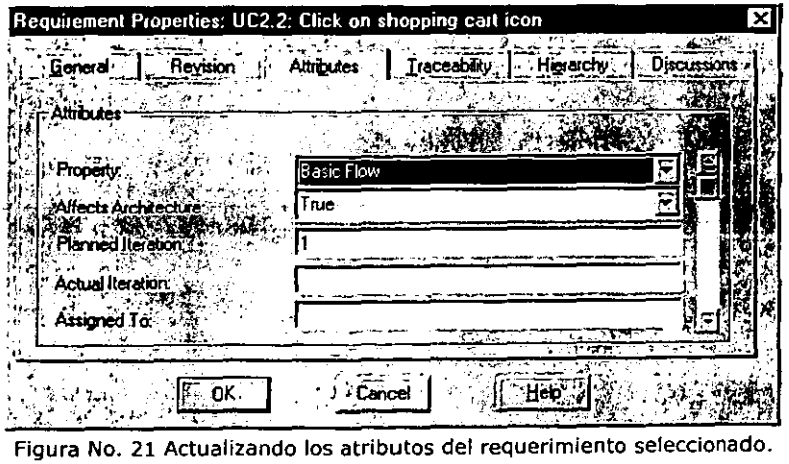


Figura No. 21 Actualizando los atributos del requerimiento seleccionado.

c3. Procedencia (Traceability).

En este punto se crea la relación entre requerimientos del mismo u otro proyecto.

c4. Jerarquía (Hierarchy).

Establece el orden y la prioridad entre requerimientos relacionados.

c5. Y las discusiones (Discussions).

Esta propiedad permite crear, leer, responder, imprimir y borrar alguna discusión con respecto a la especificación de un requerimiento.

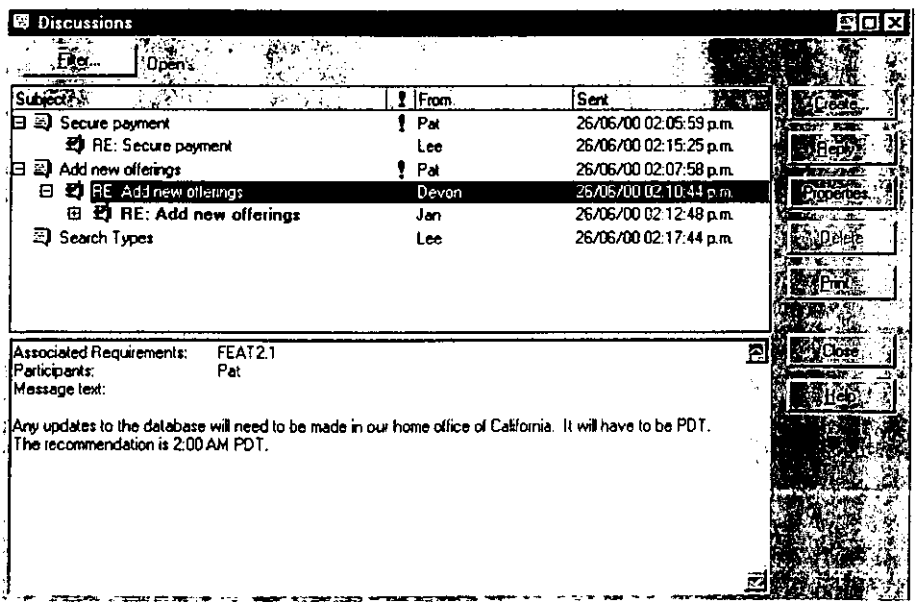
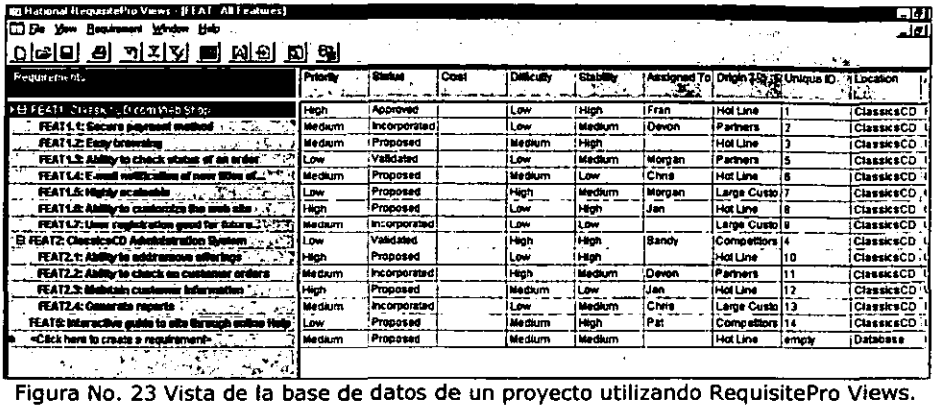


Figura No. 22 Ejemplo de discusiones acerca de un determinado requerimiento.

Para ver el contenido de la base de datos relacionada con el proyecto se llama al RequisitePro Views, desde RequisitePro, el cual ocupa vistas existentes o propias del usuario:



Requirements	Priority	Status	Cost	Difficulty	Stability	Assigned To	Origin	Unique ID	Location
FEAT1: Classics CD Commerce Site	High	Approved		Low	High	Fran	Hot Line	1	ClassicsCD
FEAT1.1: Secure payment method	Medium	Incorporated		Low	Medium	Devon	Partners	2	ClassicsCD
FEAT1.2: Easy browsing	Medium	Proposed		Medium	High		Hot Line	3	ClassicsCD
FEAT1.3: Ability to check status of an order	Low	Validated		Low	Medium	Morgan	Partners	5	ClassicsCD
FEAT1.4: E-mail notification of new titles of...	Medium	Proposed		Medium	Low	Chris	Hot Line	6	ClassicsCD
FEAT1.5: Highly scalable	Low	Proposed		High	Medium	Morgan	Large Custo	7	ClassicsCD
FEAT1.6: Ability to customize the web site	High	Proposed		Low	High	Jan	Hot Line	8	ClassicsCD
FEAT1.7: User registration good for future...	Medium	Incorporated		Low	Low		Large Custo	9	ClassicsCD
FEAT2: ClassicsCD Administration System	Low	Validated		High	High	Sandy	Competitors	4	ClassicsCD
FEAT2.1: Ability to add/remove offerings	High	Proposed		Low	High		Hot Line	10	ClassicsCD
FEAT2.2: Ability to check on customer orders	Medium	Incorporated		High	Medium	Devon	Partners	11	ClassicsCD
FEAT2.3: Maintain customer information	High	Proposed		Medium	Low	Jan	Hot Line	12	ClassicsCD
FEAT2.4: Generate reports	Medium	Incorporated		Low	Medium	Chris	Large Custo	13	ClassicsCD
FEAT3: Interactive guide to site through online help	Low	Proposed		Medium	High	Pat	Competitors	14	ClassicsCD
<Click here to create a requirement>	Medium	Proposed		Medium	Medium		Hot Line	empty	Database

Figura No. 23 Vista de la base de datos de un proyecto utilizando RequisitePro Views.

B) Norma Control

Para controlar la gestión de documentos normativos de la empresa, Norma Control utiliza seis bases de datos desde donde integra toda la aplicación:

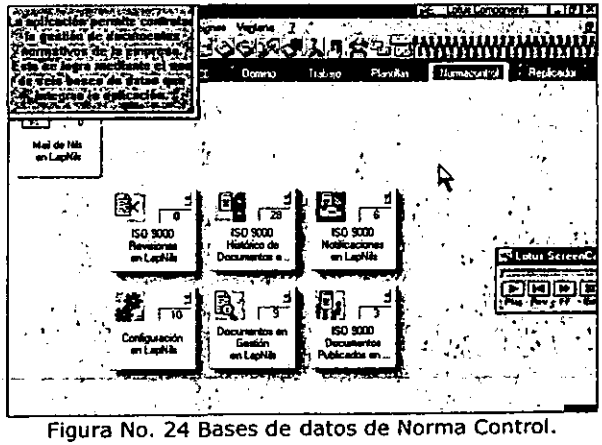


Figura No. 24 Bases de datos de Norma Control.

Abajo se describe el contenido de cada una:

1. Base de datos de configuración.

   Almacena registros con información acerca de:

   - Los tipos de documentos que maneja la empresa: instrucciones de trabajo, manuales departamentales, procedimientos, etc.
   - Localidades y departamentos.
   - Flujos de trabajo, el seguimiento para que un documento en particular sea autorizado.

2. Base de datos de documentos en gestión.

   Almacena los documentos en gestión antes de su autorización.

3. Base de datos de documentos publicados.

   En esta base de datos se publican todos los documentos autorizados listos para ser consultados.

4. Histórico.

   Base de Datos que guarda las versiones anteriores de los documentos publicados.

5. Base de datos de notificaciones.

   Conserva las notificaciones de cambio en los documentos publicados.

6. Base de datos de revisiones.

   Permite al administrador dar seguimiento a todo documento anónimo.

Las bases de datos que maneja Norma Control están especialmente diseñadas para guardar grandes volúmenes de documentos y facilitar su consulta, a este tipo de bases de datos se les conoce con el nombre de bases de datos documentales (BDD).

Las características de este tipo de almacenamiento son las siguientes:

C1. Las BDD cuentan con un sistema de indexación rápido y eficiente, donde el índice puede llegar a ser el mismo texto que conforma el documento.

C2. El lenguaje utilizado por la interfaz usuario de las BDD es semejante al lenguaje natural, buscando con esto obtener consultas apropiadas sin necesidad de programar, ya que la BDD guarda tanto preguntas como respuestas, véase figura No. 25:

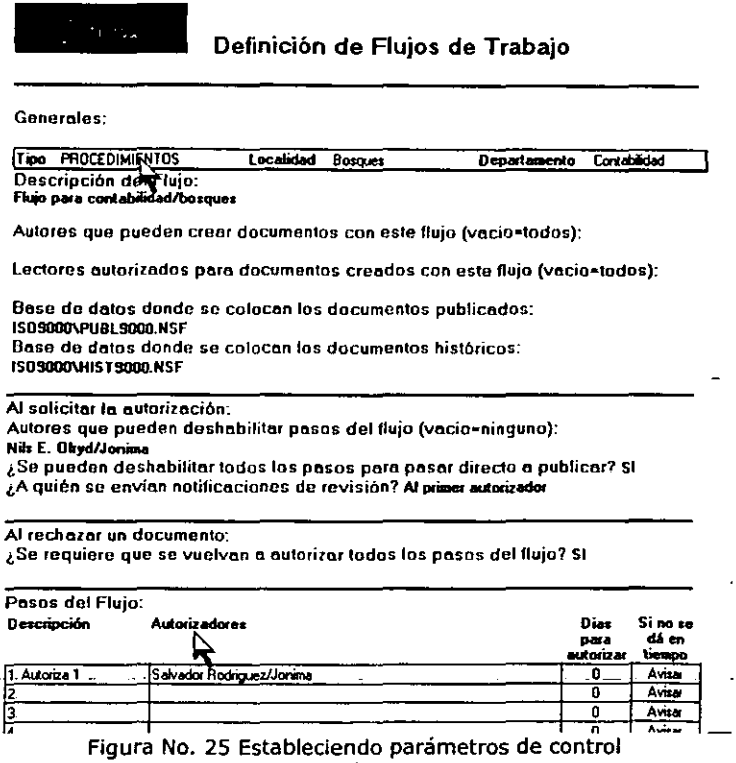


Figura No. 25 Estableciendo parámetros de control para la autorización de documentos.

C3. Como resultado de las características anteriores, se espera obtener con las BDD un sistema valioso de recuperación de información, que pueda minimizar la información no relevante.

Las bases de datos que utiliza Norma Control, utilizan la concatenación de los siguientes conceptos como índice:

- Tipo de documento.
- Localidad.
- Departamento.

El índice, conocido como descriptor en las BDD, presenta en Norma Control cualquiera de los formatos que se listan enseguida:

Tipo de documento + Localidad + Departamento
o
Tipo de documento + Departamento + Localidad
o
algún Formato Especial

Por lo regular, los usuarios de Norma Control utilizan el primero o el segundo, dejando la última opción disponible para crear un nuevo descriptor concatenando conceptos distintos contenidos en la base de datos de configuración.

El formato del descriptor es seleccionado durante el proceso de configuración como sigue:

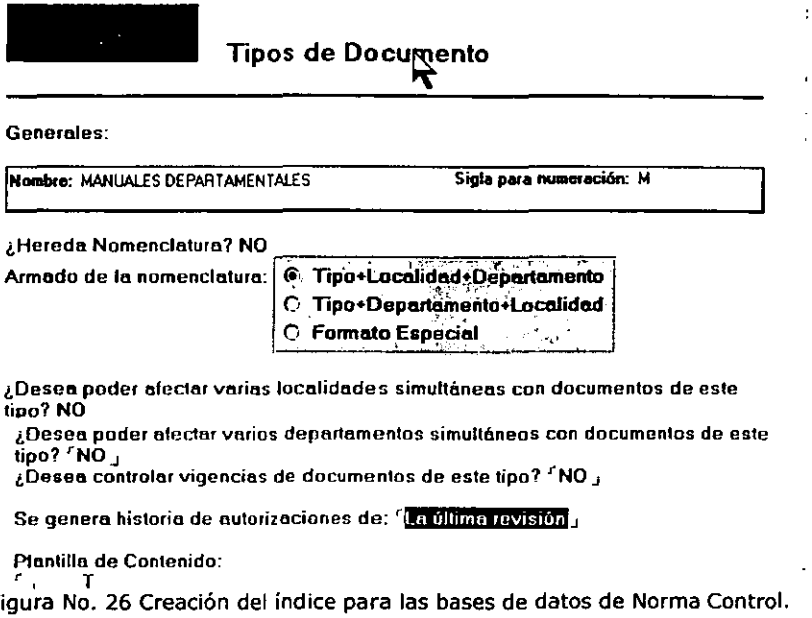


Figura No. 26 Creación del índice para las bases de datos de Norma Control.

Posteriormente, en el momento en el que se crea un documento, se le concatena un número consecutivo al descriptor con el cual se identificará el documento:

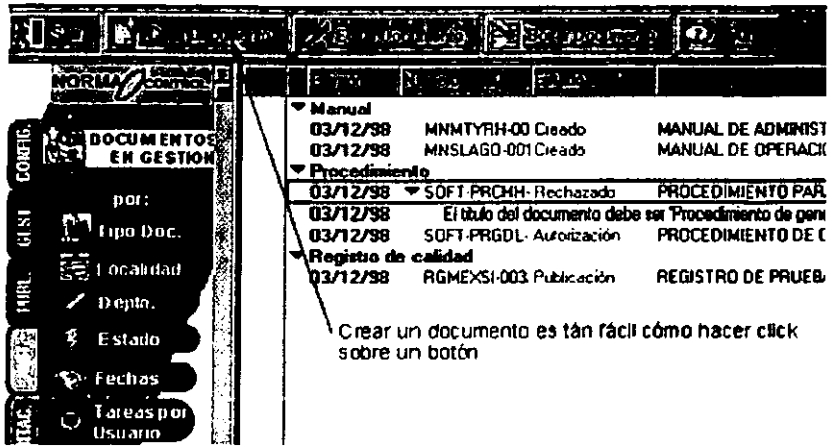


Figura No. 27 Descriptor localizado en la columna de Número.

Otra característica más de las BDD, es la de mantener al descriptor o índice como parte del texto del documento.

Con respecto a las bases de datos, Lotus Notes se encarga entre otros aspectos de:

a) La réplica de datos.

   Cada cliente maneja una copia de la información contenida en la base de datos contenida en diversos servidores y el software de Notes la sincroniza continuamente.

b) La seguridad.

   Notes ocupa la codificación y la firma usando la tecnología de clave pública RSA, que permite identificar cuando un documento no ha sido modificado durante su transmisión.

c) Maneja Listas de Control de Acceso (ACLs) que determinan quien puede tener acceso a cada base de datos y en qué medida.

d) Tiene la habilidad para administrar réplicas remotas de bases de datos desde un lugar central, si el encargado de bases de datos así lo desea.

Norma Control presenta la desventaja de permitir sólo consultas a la información almacenada en las bases de datos desde cualquier navegador de Internet, pues es necesario contar con la plataforma de Lotus Notes para poder tener acceso a las demás funciones del producto Norma Control. Lo cual no ocurre con RUP, que está diseñado precisamente para correr bajo Web.

COMPARACIÓN ENTRE PRODUCTOS:

No.	Características evaluadas	RUP	NC	Observaciones
1	Bases de Datos a utilizar			
	Microsoft Access	√	x	Para RUP se recomienda Access siempre que el número de usuarios sea como máximo de diez
	Oracle	√	x	Para RUP, cuando se tiene una gran cantidad de información, o un número de usuarios superior a diez, se recomienda el uso Oracle o SQL Server
	Microsoft SQL Server	√	x	
	Bases de datos Documentales (BDD)	x	√	NC sólo utiliza BDD
2	Permite relacionar información de otros proyectos	√	x	RUP permite compartir información de requerimientos entre varios proyectos
3	Conservar versiones de documentos anteriores	√	√	Ambos productos guardan versiones anteriores de los documentos.

Notación:

NC = Norma Control

√ = Cumple con la característica

x = No cumple con la característica

4.4. Interfaz usuario

A) RUP

La información del RUP se muestra a través de páginas Web, con características interactivas y dinámicas propias del hipertexto,[40] donde distintos puntos de la imagen llevan al usuario a revisar con mayor profundidad algunos de los conceptos de interés, o a navegar a través del árbol jerárquico desplegado en el marco de la izquierda.

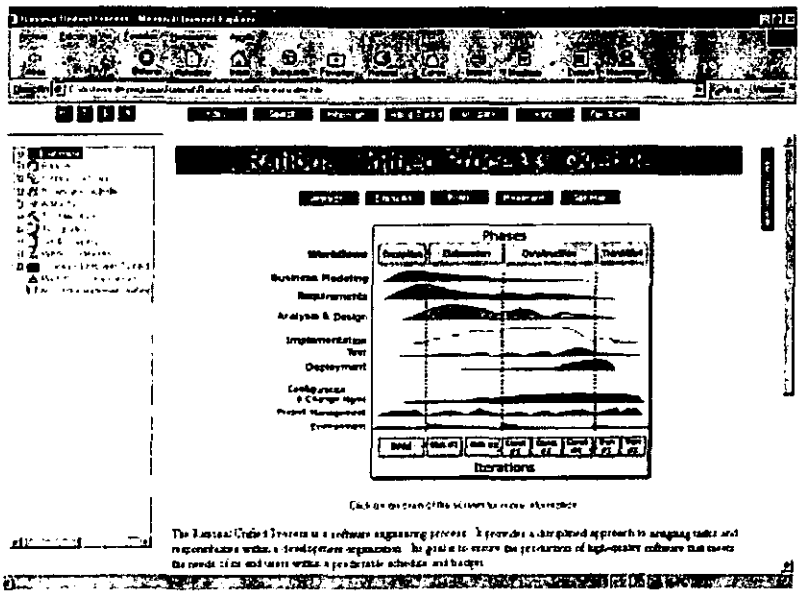


Figura. No. 28 Página principal del RUP.

[40] La idea en la que se basa el hipertexto es el ir de un punto a otro de una manera sencilla, hasta llegar a la información deseada; regresa al primer punto, navega hacia otros temas y se desplaza por el texto como uno lo desee, sin necesidad de seguir una estructura lineal, tipo libro.

Al revisar la interfaz usuario del RUP, sobresalen los siguientes tres elementos: Worker, las actividades y los artefactos[41], ver figura no. 29:

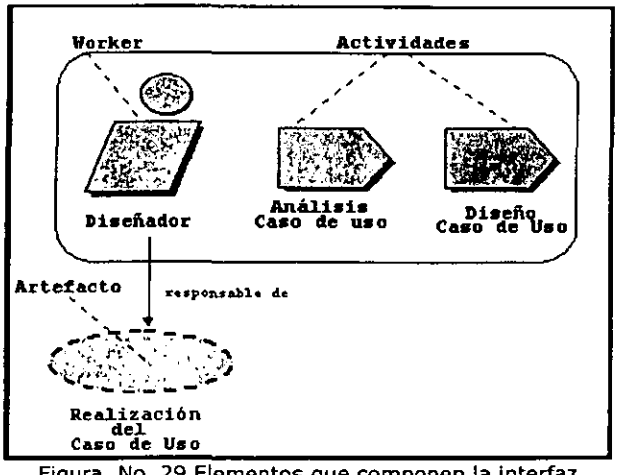


Figura. No. 29 Elementos que componen la interfaz usuario del RUP.[42]

1o.) Worker

El concepto de worker define el comportamiento y las responsabilidades de un miembro o de un grupo de trabajo en una organización de ingeniería de software. También, los workers conforman a los distintos roles que trabajan sobre un proyecto.

[41] Los artefactos son conocidos en otros ambientes con el nombre de productos de trabajo, o unidad de trabajo.
[42] Philippe Kruchten, op.cit., p.36.

Bajo Rational, los roles se encuentran bien definidos y da la apariencia de que la interfaz usuario se encuentra centrada en ellos:

Analistas
- Analista de Procesos de Negocios
- Diseñador de Negocios
- Revisor del modelo de negocios
- Revisor de los requerimientos
- Analista de Sistemas
- Analista (Specifier) de requerimientos
- Diseñador de la Interfaz usuario

Desarrolladores
- Arquitecto de Software
- Revisor de la Arquitectura
- Diseñador de la Cápsula
- Revisor del Código
- Diseñador de la Base de Datos
- Revisor del Diseño
- Diseñador
- Implementador
- Integrador

Probadores
- Diseñador de Prueba
- Probador
- Probadores

Administradores
- Administrador control de cambios
- Administrador de la configuración
- Administrador del despliegue (Deployment)
- Ingeniero de Procesos
- Administrador del Proyecto
- Revisor del Proyecto

Otros roles
- Cualquier Rol
- Desarrollador del Curso

Artista Gráfico
- Stakeholder
- Administrador del sistema

Escritor técnico
- Especialista en herramientas

En los diagramas de acción se indican las actividades que para cada rol maneja RUP, dependerá de la dirección del proyecto el seleccionar los roles que necesite.

2o.) Actividades

Se refieren al trabajo efectuado por un rol determinado con el objetivo de producir un resultado útil. Una actividad puede ser un modelo, una clase o un plan.

3o.) Los artefactos

Representan el producto tangible del proyecto, son piezas de información producidas, modificadas o utilizadas por un proceso, también definen el área de responsabilidad y son sujetos al control de versiones y a la administración de la configuración. Las actividades tienen artefactos de entrada y salida.[43]

Un artefacto puede ser:

- Un modelo: el modelo de un caso de uso o el modelo de diseño.
- Un plan de proyecto almacenado en Microsoft Project.
- Un elemento del modelo, una clase, un caso de uso, o un subsistema.
- Un documento, como el documento de la arquitectura de software.
- El código fuente.
- Los ejecutables.

Existen, además, otros elementos que se añaden a las actividades o a los artefactos para hacer el proceso más claro, estos elementos son:

4o.) Guías

Son documentos que se encuentran ligados a las actividades, etapas o artefactos para describirlos en detalle. Por ejemplo:

- Guías de trabajo, proporcionan consejos prácticos de cómo entender una actividad.

[43] En términos de OO, las actividades son las operaciones en un objeto activo (worker), los artefactos son los parámetros de las actividades.

- Guías de artefactos, describen cómo desarrollar, evaluar y usar el artefacto al cual se les liga.
- Guías de interfaz-usuario, describen el estilo a utilizar para crear las ventanas propias del proyecto, incluye colores, galería de iconos, etc.
- Guías de programación, describen los pasos para crear programas bien formados.

5o.) Templates

Son modelos o plantillas de artefactos, un artefacto puede utilizar uno o más templates.

6o.) Guía de herramientas

Muestra al usuario cómo ejecutar etapa por etapa, indicándole qué herramienta de software utilizar.

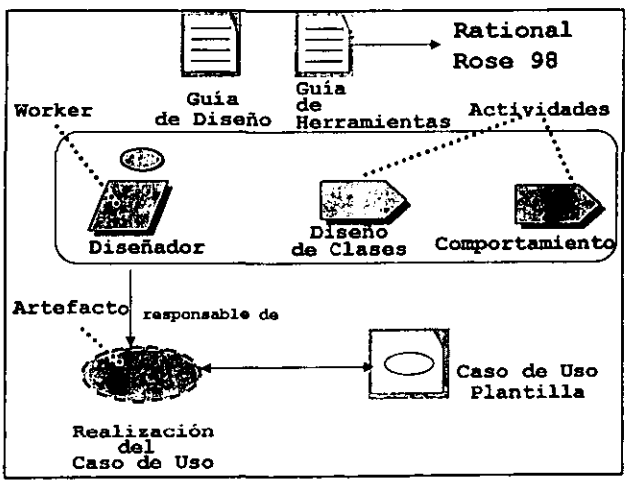


Figura. No. 30 Plantillas, guías de diseño y guías de herramientas.[44]

[44] Philippe Kruchten, op.cit., p.48.

Cada rol tiene acceso a información particular, por ejemplo: el Analista de Procesos de Negocios, quien define el comportamiento (las actividades que realiza una persona) y las responsabilidades (de control, creación y modificación de componentes) de un miembro, o de un equipo de trabajo de ingeniería de software, cuenta con el siguiente diagrama de acción:

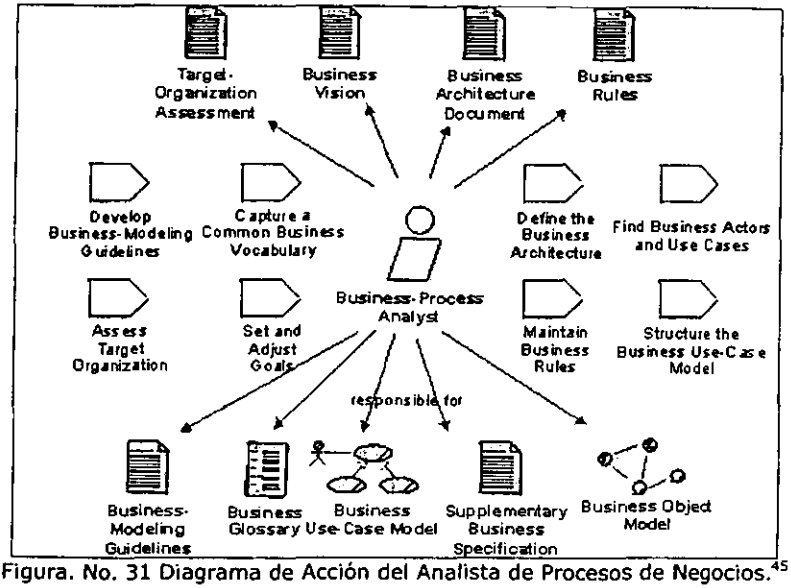


Figura. No. 31 Diagrama de Acción del Analista de Procesos de Negocios.[45]

Diagrama totalmente diferente con respecto al diagrama propiedad del Diseñador de la Interfaz Usuario:

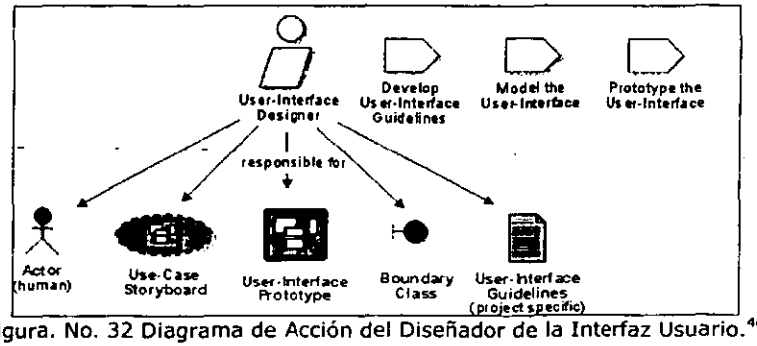


Figura. No. 32 Diagrama de Acción del Diseñador de la Interfaz Usuario.[46]

[45] Diagrama proveniente de la Ayuda del RUP.
[46] Ibidem.

Como se muestra en los diagramas anteriores, RUP asigna actividades y responsabilidades distintas y precisas para cada uno de los roles establecidos, dejando sin acceso a quien no cuente con la debida autorización. Este control facilita una comunicación precisa y sin ambigüedad entre los miembros de un equipo de trabajo, pero al mismo tiempo implica contar con gente capacitada en el ambiente Rational, significando con ello tiempo y dinero.

La interfaz de los productos Rational se encuentra en inglés y no existe la opción de modificar el idioma. Pero presenta la ventaja de navegar con facilidad por la herramienta gracias a que se basa en diagramas visuales, adicionalmente, la ayuda que presenta es muy completa, también es posible solicitar ayuda en línea siempre y cuando se cuente con la licencia.

B) Norma Control

Norma Control también presenta formato Web, disponible para una Intranet que cuente con Lotus Notes Versión 4.5 en adelante y el permiso de acceso, pero fuera de esta plataforma, sólo es posible consultar los documentos publicados con autorización, ya que por seguridad los documentos creados llegan únicamente a las personas indicadas, en el momento deseado.

Pese a que Norma Control no presenta facilidades visuales, maneja gráficos valiéndose de herramientas externas —productos de Microsoft, del mismo Lotus, etc.— con los cuales genera archivos que vincula a documentos de Norma Control:

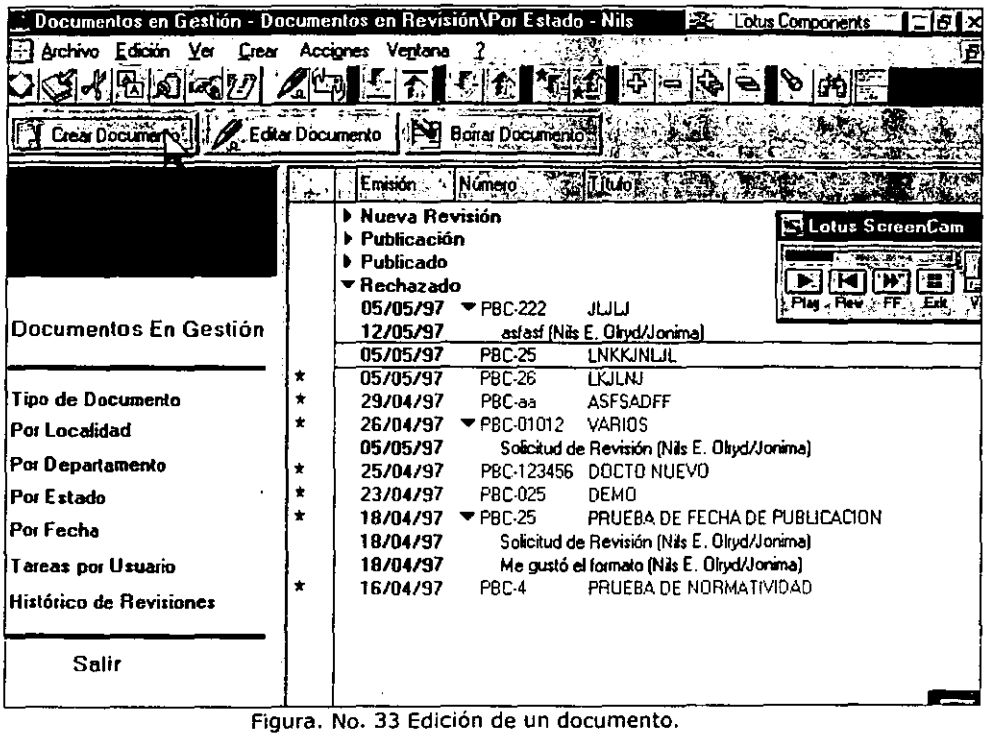


Figura. No. 33 Edición de un documento.

El manejo de Norma Control es muy sencillo, y lo es aún más debido a que la interfaz se encuentra escrita en español, por lo que no es necesario tener una capacitación prolongada y costosa.

COMPARACIÓN ENTRE PRODUCTOS:

No.	Características evaluadas	RUP	NC	Observaciones
1	La información se muestra a través de páginas Web	√	x	NC sólo muestra las consultas a través de páginas Web
2	Presenta facilidades visuales	√	x	NC utiliza herramientas gráficas externas
3	Utiliza plantillas para guiar al desarrollador sobre los pasos a seguir para crear un proceso	√	x	Las plantillas que utiliza RUP son creadas con Microsoft Word
4	Cada rol tiene acceso a información particular	√	√	Los roles y sus actividades están bien definidos
5	Interfaz se encuentra escrita en:			
	Inglés	√	x	RUP no permite modificar el idioma
	Español	x	√	Jónima trabaja para tener la versión en inglés de NC
6	Plataforma en que corren los productos			
	Unix	√	x	RUP corre en cualquiera de
	Linux	√	x	estas plataformas
	Windows NT	√	x	
	Lotus Notes	x	√	NC sólo corre en Lotus Notes

Notación:

NC = Norma Control
√ = Cumple con la característica
x = No cumple con la característica

4.5. Conjunto de Herramientas

A) RUP

La característica primordial de las herramientas propiedad de Rational, es la combinación de herramientas con procesos de ingeniería de software necesarios para dominar la complejidad del desarrollo de software.

Rational apoya con sus herramientas a cada una de las siguientes fases:

1a). Administración del proyecto.

Esta función comprende las tareas de calendarización, rastreo y medición de los procesos. Una mejor definición la ubica como el arte de liberar software de alta calidad en el tiempo establecido y con el presupuesto acordado. Incluye:

- La asignación y calendarización del proyecto.
- La inspección del proyecto.
- La guía del proceso.
- La identificación y administración de los riesgos del proyecto.
- El monitoreo del progreso del equipo y la planeación de cada iteración.

Rational permite, a quien ejerce el rol de administrador, evaluar el estado real del proyecto e implementar un proceso personalizado de desarrollo.

Enseguida se mencionan algunos de los productos del Rational recomendables para ayudar en la administración del proyecto:

- Rational Unified Process (RUP)

Es un proceso de ingeniería de software que mejora la productividad del equipo de trabajo con guías, formatos, y ejemplos para cada una de las actividades involucradas en el desarrollo.

- Rational ClearQuest

Herramienta enfocada a detectar detalles realizados por los administradores del proyecto y que se muestran a través de diagramas e informes. Además, recopila, administra y asigna las solicitudes de cambio en el software. También proporciona una interfaz común para cada una de las personas involucradas en el proyecto, ya sean analistas, desarrolladores, los de prueba, los usuarios.

- Rational RequisitePro

Administra el área común donde se almacenan los requerimientos siempre disponibles para cada uno de los miembros del equipo de trabajo.

- Rational TestManager

Es la pieza fundamental de las herramientas de prueba Rational, ya que controla y administra a todas las actividades prueba. Con esta herramienta se espera tener siempre una visión clara del alcance de los requerimientos, contando en todo momento con la información necesaria para tomar decisiones para administrar el proyecto de una manera eficiente.

2a). Definición del sistema.

Se refiere al proceso de obtener un entendimiento claro de:

- Los problemas reales del negocio.
- Las necesidades de los usuarios.
- La definición de una aplicación para corregir el conflicto a la primera.

Para llevar a cabo este proceso, Rational ocupa el modelado visual y los casos de uso, elementos que permiten al equipo de desarrollo adentrarse gráficamente al conocimiento del negocio, elaborando con ello una solución ajustada a la realidad.

Las herramienta Rational utilizadas para definir un sistema cumplen con las siguientes actividades:

- Resuelven las necesidades cambiantes del usuario.
- Notifican de la liberación tardía de algunas versiones.
- Capacitan al grupo de trabajo para entender mejor los problemas, identificando con mayor eficiencia las necesidades del usuario, y sobre todo comunicando las soluciones propuestas de una manera más clara.

Para definir el sistema, Rational recomienda los siguientes productos:

- Rational Suite AnalystStudio

Conecta los negocios, las aplicaciones y los modeladores de datos a través de un lenguaje de modelado común. Es, además, una herramienta que proporciona un amplio apoyo para los casos de uso asegurando con ello que la solución será diseñada tomando en cuenta al usuario.

- Rational RequisitePro

Integra la administración de los requerimientos dentro del ciclo de vida del desarrollo de software, asegurando con esto una visión común por parte de los miembros del equipo. No obstante la integración de las herramientas, los requerimientos se mantienen disponibles para continuar con el seguimiento, la definición de la arquitectura del sistema, las actividades prueba y la documentación.

- Rational Rose

Es la herramienta que ofrece la capacidad de modelar y visualizar los procesos de negocios, convirtiendo los requerimientos de alto nivel en una arquitectura elástica basada en componentes. Rational encabeza el desarrollo en lenguaje unificado (UML), estándar usado por Rational Rose para especificar, visualizar y construir modelos de software y sistemas.

El Rational Rose integra tres tipos de analistas: los del negocio, los de sistemas y los de datos, habilitándolos para crear y administrar modelos.

- Rational ClearQuest

3a). Desarrollo de software.

Comprende el modelado de:

- Los casos de uso y de los datos.
- La arquitectura.
- Los componentes.
- La construcción del código.
- La prueba unitaria.

En el desarrollo de software, Rational utiliza un conjunto de herramientas que manejan el análisis orientado a objetos, el modelado, el diseño y su construcción.

Las herramientas Rational ayudan a crear software flexible al cambio, administrando las tecnologías involucradas y los requerimientos de diseño, controlando un conjunto inmenso de estructuras, versiones y defectos. Con la unidad prueba es posible encontrar problemas a tiempo para arreglarlos enseguida.

Las herramientas utilizadas en el desarrollo de software se mencionan enseguida:

- Rational Suite DevelopmentStudio

Es una solución integrada, conviene a los arquitectos del software, diseñadores y analistas para construir rápidamente un mejor software.

Crea estructuras flexibles, ya que automáticamente genera pruebas basadas en modelos y escenarios, utilizando herramientas de diagnóstico que localizan fácilmente defectos y errores de ejecución, precisa los cuellos de botella, y verifica el alcance del código.

- Rational Rose
- Rational ClearCase

Herramienta que administra completamente la configuración del software (SCM), por lo que es fundamental para la productividad y la calidad del equipo.

Algunas de las funciones del Rational ClearCase son:

a) Tener control sobre las versiones.

b) Administrar el espacio de trabajo.

c) Configurar el proceso.

d) Establecer la administración del mismo.

- Rational Quality Architect

Extensión poderosa del Rational Rose, ya que genera automáticamente manejadores de prueba de los modelos UML para apoyar funcionalmente la ejecución de algunas

pruebas de sus componentes y subsistemas. Valida la calidad del software de una manera temprana.

🛠 Rational Purify, Rational Quantify, Rational PureCoverage

Estas herramientas localizan automáticamente errores de ejecución y problemas con respecto a la administración de memoria, destacando los cuellos de botella e identificando los códigos que no han sido probados dentro de la aplicación.

🛠 Rational ClearQuest

Con el Rational ClearQuest y con el Rational ClearCase, el equipo puede usar el administrador de cambios unificados (UCM) para manejar los requerimientos modificados, los modelos de diseño, la documentación, los componentes, los casos de prueba y el código fuente.

4a). Administración del contenido.

Herramientas Rational que ayudan a construir aplicaciones para el comercio electrónico con altos estándares de calidad.

Características básicas del administrador del contenido:

- Integra el sistema de información del negocio con el software de comercio electrónico.
- Utiliza un proceso unificado para administrar un buen desarrollo del Web y del código.
- Se enfoca en la plantilla, la aprobación, el despliegue y la integridad del sitio.

Las herramientas correspondientes a la administración del contenido son:

🛠 Rational Suite ContentStudio

Proporciona acceso común a herramientas y procesos. Une las actividades de todos los que contribuyen al sitio Web —incluyendo administradores del proyecto, analistas, desarrolladores de software, administradores del contenido, diseñadores del Web, los de prueba, y los administradores del área de comercio electrónico.

- Rational SiteLoad

Herramienta, basada en el Web, que proporciona simultáneamente el tráfico en Internet y la información precisa en tiempo real de la ejecución del sitio.

5a). Sistema de pruebas.

Se encarga de probar, de una manera temprana y con frecuencia, el diseño y el modelado inicial, notificando acerca de la funcionalidad, fiabilidad, y ejecución del proceso de desarrollo.

Las herramientas recomendables para el sistema de prueba son las siguientes:

- Rational Suite TestStudio

Solución completa de prueba que revisa la funcionalidad y la ejecución del proceso.

- Rational TestManager

Sistema que une a todas las herramientas prueba, los mecanismos, y los datos para ayudar al equipo a definir y corregir sus metas con respecto a la calidad. Contiene el plan prueba que guía al equipo hacia el logro de las metas, ya que guarda las claves necesarias para fijar el estado del sistema en un punto dado durante el ciclo de vida del proyecto.

- Rational TeamTest

Prueba de una manera funcional y durante la ejecución, la búsqueda de defectos.

- Rational Robot

Herramienta de prueba para aplicaciones de comercio electrónico, ya que permite crear, modificar y correr automáticamente pruebas funcionales en el Web y en aplicaciones cliente/servidor.

- Rational VisualTest

Con esta herramienta los miembros de un equipo de trabajo pueden desarrollar pruebas reusables.

- Rational SiteLoad

Herramienta de prueba basada en el navegador de aplicaciones del Web, esencialmente simula el tráfico en Internet y provee, a los analistas de prueba, la información precisa en tiempo real sobre la ejecución del sitio.

- Rational PreVue

Estos productos son soluciones de prueba para Windows X y para aplicaciones basadas en terminales. Esta herramienta automatiza la prueba de ejecución para emular las actividades tanto de los usuarios como de los dispositivos físicos para reportar una representación realista del trabajo de la aplicación.

- Rational TestFoundation for Windows 2000

Es una herramienta "free" para validar las aplicaciones del software comparándolas con las del Windows 2000 Application Specification.

- Rational Purify, Rational Quantify, Rational PureCoverage

Estas herramientas despliegan un amplio rango de información con el diagnóstico del comportamiento de la aplicación evaluada. Rational Purify encuentra localidades de memoria dañadas y errores de ejecución que pueden causar problemas graves; Rational Quantify localiza cuellos de botella en la aplicación y Rational PureCoverage identifica código que no ha sido probado, proporcionando un análisis del mismo.

- Rational QualityAchitect
- Rational ClearQuest

Con esta herramienta se notifica automáticamente a los desarrolladores de los defectos, incluyendo una evaluación y una descripción de las etapas que produjeron los errores. De esta forma, los desarrolladores pueden fácilmente corregir los problemas. Además, Rational ClearQuest registra todos los cambio para cualquier tipo de proyecto en cualquier plataforma.

A continuación se hace un resumen de las características que distinguen a cada una de las fases ya mencionadas:

No.	Fase	Característica
1	Administración del proyecto	➢ Asigna y calendariza el proyecto. ➢ Identifica y administra los riesgos del mismo. ➢ Monitorea el progreso del equipo. ➢ Evalúa el estado real del proyecto, notificando un presupuesto sobregirado o un proyecto atrasado de acuerdo a la planeación. ➢ Implementa un proceso personalizado de desarrollo correspondiente al equipo. ➢ Informa de los cambios efectuados y registrados dentro del sistema. ➢ Incluye procesos de ingeniería de software que auxilian a las organizaciones en el desarrollo de un mejor software en corto tiempo.
2	Definición del sistema	➢ Rational administra los requerimientos, el modelado visual y los casos de uso. ➢ Resuelve las necesidades cambiantes del usuario. ➢ Notifica de la liberación tardía de versiones. ➢ Con esta fase, al equipo de trabajo se le capacita para: ▪ entender mejor los problemas, ▪ identificar las necesidades del usuario y a ▪ comunicar claramente las soluciones propuestas.
3	Desarrollo de software	➢ Crea software que permite cambios. ➢ Permite que la unidad de prueba encuentre problemas tempranamente para un rápido arreglo.
4	Administración del contenido	➢ Integra el sistema de información del negocio con el software de comercio electrónico. ➢ Utiliza el proceso unificado para administrar un buen desarrollo en el Web.
5	Sistema de pruebas	➢ Revisa la funcionalidad, la fiabilidad, y la ejecución de la prueba ➢ Prueba la calidad del software y su desempeño en corto tiempo

Rational cuenta, además, con una plataforma que cubre cada una de las fases anteriores: Rational Suite Team Unifying Plataform.

Algunas características presentes en esta herramienta son:

- Comunica al equipo de trabajo de los cambios efectuados en los procesos.
- Crea procesos de desarrollo confiables, predecibles y soportados por sus propias herramientas.

Esta herramienta apoya la creación del Rational Suite Studio que incluye:

- Rational Unified Process

Asegura que todos los miembros de un equipo de trabajo entiendan y puedan seguir fácilmente un proceso práctico cuando desarrollen el software.

- Rational ClearCaseLT

Proporciona la infraestructura correspondiente para administrar los cambios.

- Rational TestManager

Administra y controla todas las actividades prueba: planeación, administración y ejecución de pruebas a través del proyecto.

- Rational RequisitePro
- Rational ClearQuest

Registra los cambios y los defectos personalizados que dan a todos un entendimiento claro del estado actual de la calidad del software.

- Rational SoDA

Automatiza la generación y el mantenimiento de la documentación, facilitando su distribución.

B) Norma Control

Norma Control permite vínculos con herramientas externas, por ejemplo, de la paquetería de Microsoft accesa comúnmente archivos de Word, Project y Excel.

Con otras de las herramientas de Jónima, Audit Control e ISO Control, comparte la base de datos de configuración, área donde se establecen las reglas para la auditoría de calidad.

Audit Control es una aplicación que se encarga de todos los procesos y requerimientos de una auditoría interna, e ISO Control, facilita la labor administrativa de control dando un seguimiento a las auditorías tipo ISO.

El objetivo principal de la herramienta Norma Control, es el de llevar un control eficiente de toda la documentación de la empresa, alcanzando con ello otras metas:

- Un ahorro en el uso de recursos como tiempo, papel y comunicación.
- Mejorar la productividad y la calidad de la empresa documentando todo proceso.

La herramienta Norma Control, presenta algunas de las características que identifican a los ambientes PSEE:

- Genera plantillas de documentos para un uso posterior.
- Lleva el control de los cambios, conservando distintas versiones de los documentos en la base de datos conocida como Históricos.
- Utiliza un depósito común con acceso a otras herramientas.
- Maneja el concepto de seguridad absoluta en la información y en los procesos, donde sólo a usuarios autorizados se les permite el acceso.
- Se envía a usuarios definidos la notificación de algún cambio en los documentos.
- Ocupa cualesquier tipo de ISO (Control de Auditoría Interna de Calidad), la empresa establece la norma de calidad que necesita obtener.
- Al rechazarse un documento, Norma Control puede requerir la revisión de todo el proceso de autorización.
- Si bien, Norma Control no genera software, si es una herramienta que algunas empresas ocupan para documentar el proceso de desarrollo de software.

COMPARACIÓN ENTRE PRODUCTOS:

No.	Características evaluadas	RUP	NC	Observaciones
1	Áreas que cubre un ambiente PSEE			
	Administración del proyecto	√	√	En sí, es lo fuerte de NC
	Ingeniería de Procesos	√	x	NC permite solamente definir y almacenar la documentación de los procesos y sus versiones
	Ingeniería de software	√	x	
2	Existen vínculos con otras herramientas	√	√	Por ejemplo, para revisar alguna plantilla de RUP, es necesario abrir Word para poder accesarla. NC genera ligas con la paquetería de Microsoft ya que no cuenta con facilidades gráficas
3	Las herramientas tienen acceso a un depósito común	√	√	
4	Existe notificación de cambio entre herramientas	√	√	
5	Evalúa el avance del proceso	√	√	

Notación:

NC = Norma Control
√ = Cumple con la característica
x = No cumple con la característica

4.6. Ventajas y desventajas de estos productos

A) RUP

RUP y las herramientas Rational presentan todas las características que requiere un ambiente PSEE. Abarcando, sin duda, cada una de las fases del proceso de desarrollo de software.

RUP deja en libertad al desarrollador para crear sus propios modelos de procesos, los cuales debe seguir y modificar con ayuda de las iteraciones que maneja.

RUP y la tecnología Rational presentan una mercadotecnia que promete con seguridad el alcance de objetivos inmediatos en el desarrollo de software. Pero les falta enfatizar que para conseguir resultados positivos es necesario, entre otras cosas, capacitar al personal involucrado en orientación a objetos, en UML y en la tecnología Rational. Sólo de esta manera es posible alcanzar los objetivos esperados.

Algunas de las ventajas que presenta RUP, se listan a continuación:

- Maneja procesos dinámicos, procesos que pueden ser corregidos al estarse ejecutando.
- El desarrollador puede crear sus propios procesos o valerse de las plantillas donde se encuentran definidos.
- RUP maneja ayuda en línea para sus clientes, al menos en Estados Unidos.
- Se cuenta con varios libros en el mercado que tratan acerca del RUP y UML.
- La ayuda del RUP es extensa.

Desventajas que presenta el uso de RUP:

- Los usuarios pueden requerir de capacitación para manejar tanta plantilla y no perderse entre ellas.
- Es necesaria cierta disciplina para seguir los pasos indicados por RUP al momento de precisar los detalles que conforman un proyecto.

- Seguramente será necesario preparar al personal en alguna de las áreas siguientes: la orientación a objetos, el lenguaje de modelo unificado o la tecnología Rational.

B) Norma Control

Familiarizarse con Norma Control es realmente sencillo, por ejemplo, si hay alguna duda sobre los pasos necesarios para autorizar algún documento, basta con revisar el flujo de trabajo; ahora bien si se desea conocer en qué punto de la autorización se encuentra el documento es suficiente consultar el concepto de estado. Conocer Norma Control es muy fácil, tanto así que la capacitación enfocada a los usuarios finales: autor, autorizador y publicador, dura únicamente cuatro horas, y para líderes de proyecto del área de calidad y administradores tarda 8 horas.

Algunas ventajas que presenta Norma Control son:

- La interfaz que maneja es sencilla.
- Su precio es de $13 000.
- El tiempo que se ocupa para capacitar a los usuarios es corto, puede tratarse de cuatro u ocho horas, si se trata de administradores.
- El uso de Norma Control podría ser adecuado para llevar el control de la documentación de proyectos pequeños de desarrollo de software.
- Podría ser una herramienta adecuada de comunicación entre usuarios de áreas ajenas a la Ingeniería de Software.
- Algunas desventajas que presenta Norma Control son listadas enseguida:
- Los documentos publicados son los únicos que pueden verse a través del Web, todo lo demás es posible accesar siempre que se cuente con Lotus Notes y se tenga permiso de acceso.
- Como se utiliza para controlar cualquier proceso de la empresa registrando todo documento, no cuenta con alguna metodología formal como lo hace Rational.

4.7. Cuadro comparativo entre productos

En este apartado se hace un resumen, en forma de tabla, de las once características que debe contener todo PSEE de acuerdo con el punto 3.3. Se espera que esta tabla sirva para futuras revisiones.

Determina procesos redundantes en el modelo	✓	✓	Ambos productos utilizan estándares de calidad
Incluye vistas formales o informales mostrando las actividades del proceso	✓	✓	Cada rol cuenta con una vista del proceso
Evaluación del proceso antes del gasto de recursos	x	x	Puede valerse de herramientas como Rational SiteLoad, que simula la ejecución de los procesos en Internet, o Rational PreVue, que emula las actividades de los usuarios.
Automatización de actividades para su notificación	✓	✓	
Monitoreo de la ejecución del proceso y su registro	✓	✓	
Cambios en procesos en plena ejecución	✓	x	RUP maneja procesos dinámicos
Muestra la siguiente etapa a realizar	✓	✓	
Presenta la información o herramientas necesarias de acuerdo al usuario y a la etapa a efectuar.	✓	✓	
Intercambio de datos	✓	x	Norma Control no maneja datos
Apoya a un equipo de trabajo	✓	✓	

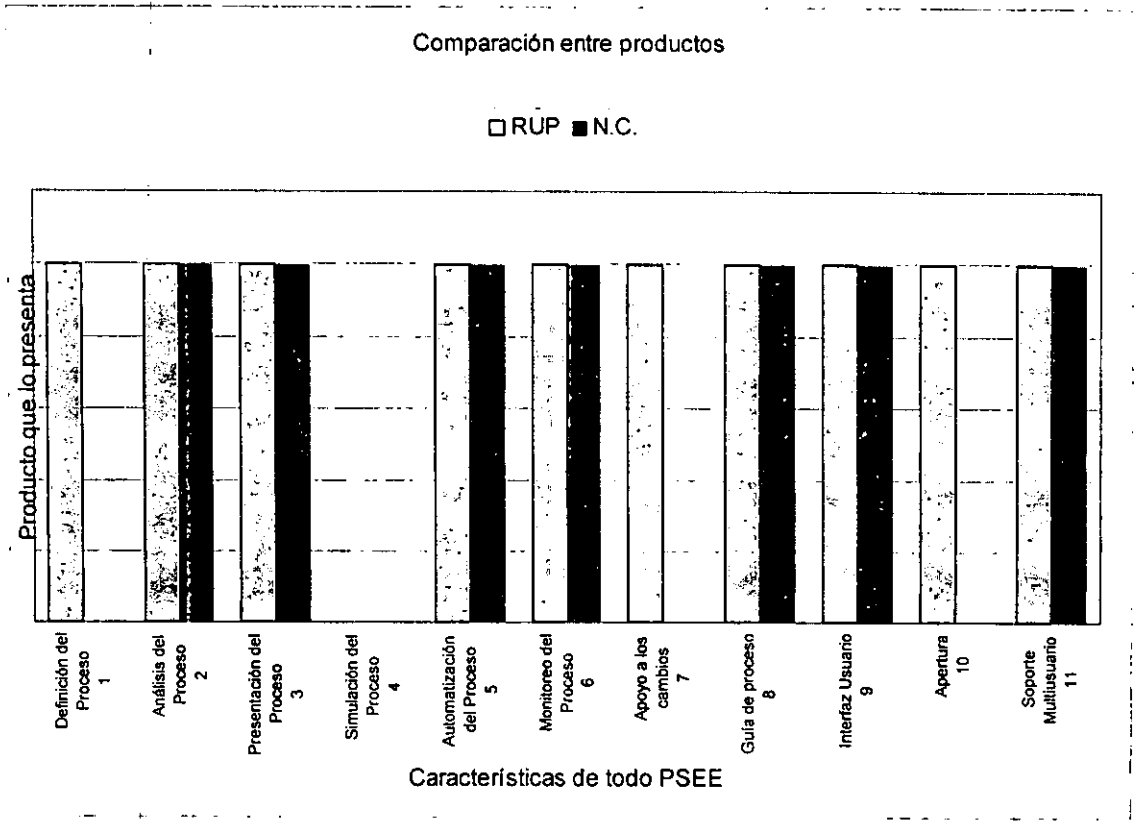
Notación:

NC = Norma Control
√ = Cumple con la característica
x = No cumple con la característica

Como puede verse en la tabla, RUP cumple con casi todas las características que conforman un ambiente PSEE, no así Norma Control, que apenas puede considerársele como un ambiente incompleto pero eficiente, de acuerdo a las tareas que promete y realiza.

GRÁFICA:

Enseguida se muestra la tabla anterior en forma gráfica:



CONCLUSIONES

Para auxiliar en el desarrollo de sistemas computacionales se cuenta, hoy en día, con herramientas de software que conjugan un cúmulo de experiencias resultado de varios años de investigación por parte de estudiosos de todo el mundo. Las herramientas CASE, I-CASE y los ambientes PSEE, surgen para brindar tal apoyo.

De acuerdo con lo mostrado en esta investigación, el término PSEE absorbe por completo los términos CASE e I-CASE, pues mientras el primero está dirigido a un área específica, el I-CASE busca integrar a las herramientas tratando de compartir un depósito común. El PSEE, en cambio, tiene entre sus objetivos las tareas anteriores, pero con un enfoque global y dinámico de los procesos de la empresa.

El empleo dinámico de los procesos, es decir, la posibilidad de corregirlos al estar siendo ejecutados, es una de las características primordiales que distinguen a los PSEE. En el RUP, uno de los productos analizados en este trabajo, cumple con tal cualidad, mientras Norma Control, el otro producto, no lo hace, a pesar de cubrir muchas de las características de este tipo de ambientes. Quizá se deba a que el enfoque de Norma Control fue simplemente dar un seguimiento a la generación de los distintos documentos organizacionales y controlar su impresión.

La compañía Rational con sus productos, entre ellos RUP, está procurando crear estándares útiles para apoyar el trabajo del ingeniero de software, sin embargo, debido a sus precios y costos de capacitación, estos productos son difícilmente adquiridos por pequeñas y medianas compañías; las cuales, muchas veces, apenas adquieren las herramientas indispensables para trabajar.

Con respecto a las grandes empresas, a pesar de contar con todo tipo de herramientas para facilitar el desarrollo de sistemas computacionales, muchas de estas compañías, utilizan y conocen un reducido porcentaje de los recursos propios de software. Por lo anterior, se va haciendo indispensable introducir en los planes de estudio universitarios, su análisis y aplicación, para, posteriormente, promover su aplicación en distintos ámbitos.

Se espera, con este trabajo, haber dado un panorama general de las herramientas de software hasta la época actual. Dejando —para futuras evaluaciones de productos— un resumen en forma de tabla, de las cualidades que de acuerdo a los textos consultados, debe contener todo sistema para ser considerado un ambiente PSEE. Ambientes cuyo objetivo primordial es mejorar la calidad de los productos de software y conseguir los objetivos trazados por el personal de sistemas, a un tiempo y costo previamente establecidos.

BIBLIOGRAFÍA

Feiler, P.H. y W.S. Humphrey, "Software Process Development and Enactment: Concepts and Definitions" en Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Pankaj K. Garg, y Mehdi Jazayeri (eds.), Los Alamitos, CA, 1996 [Proceedings of the 2h International Conference on Software Process, 1993], pp. 37-49.

Fuggetta, Alfonso, "A classification of CASE Technology", en Pankaj K. Garg, y Mehdi Jazayeri (eds.), Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Los Alamitos, CA, 1996 [Computer, diciembre 1993], pp. 289-302.

Garg, Pankaj K., y Mehdi Jazayeri (eds.), Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Los Alamitos, CA, 1996.

Hsieh, David, "'David Hsieh, of LBMS, 'Integrated CASE is dead'" [entrevista de VARbusiness a David Hsieh] en Manhasset, Nov. 1, 1995, p. 136.

Jacobson, Ivar, Grady Booch y James Rumbaugh, The Unified Software Development Process, Addison-Wesley, Massachusetts, 1999.

Kruchten, Philippe, The Rational Unified Process an Introduccion, Addison Wesley, Boston, segunda edición, 2000.

Lehman, M M., "Process Models, Process Programs, Programming Support, en Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Pankaj K. Garg, y Mehdi Jazayeri (eds.), Los Alamitos, CA, 1996 [Proceedings of the 9th International Conference on Software Engineering, 1987], pp. 34-36.

Madhavji, N.H., "The Process Cycle", en Process-Centered Software Engineering Environments, IEEE-Computer Society Press, Pankaj K. Garg, y Mehdi Jazayeri (eds.), Los Alamitos, CA, 1996 [Software Engineering Journal, septiembre, 1991], pp. 50-58.

McClure, Carma, "CASE Experience" en BYTE, Abril, 1989.

Microsoft Corp., "La Revolución Industrial", Enciclopedia Microsoft Encarta 99, 1990-1995.

Ocampo, Camilo, Begoña Albizuri y Pere Botella, "De la Tecnología CASE a la Tecnología de procesos" en Soluciones Avanzada, Enero 1999, pp.50-57.

Oktaba, Hanna, y Guadalupe Ibargüengoitia González, "Software process modeled with objects: static view", en Computación y sistemas, Revista iberoamericana de computación, vol. 1, no. 4, abril-junio, 1998, pp. 228-238 [Instituto Politécnico Nacional, Centro de Investigación en Computación].

Parkinson, John, Making CASE Work, strategies for the successful introduction of CASE technology in comercial information systems development, NCC,

London, 1991.

Pressman, Roger S., Ingeniería del Software, un enfoque práctico, 3a. edición, Mc Graw Hill.

Silberschatz, Abraham, Henry F. Korth, S. Sudarshan, Fundamentos de Bases de Datos, trad. Fernando Sáenz Pérez, Mc Graw Hill, Madrid, 3a edición, 1999.

Terry, B. and D.Logee, "Terminology for Software Engineering and Computer-aided Software Engineering" en Software Engineering Notes, Abril, 1990.

Referencias electrónicas:

Stobart, Simon, http://osiris.sunderland.ac.uk/sst/case2/welcome.html

http://www.aimware.com

http://www.fisicc-ufm.edu/~progra6/documental

http://www.jonima.com.mx

http://www.lotus-notes-software.com

http://www.Rational.com