Uso de herramientas CASE para la gestión de proyectos de software

Juan Ignacio Cerca Vázquez
Instituto Tecnológico de Celaya
nacho@itcelaya.edu.mx

Luis Alberto López González
Instituto Tecnológico de Celaya
luis.lopez@itcelaya.edu.mx

José Jesús Sánchez Farías
Instituto Tecnológico de Celaya
jesus.sanchez@itcelaya.edu.mx

Oscar Grimaldo Aguayo
Instituto Tecnológico de Celaya
oscar.grimaldo@itcelaya.edu.mx

Ramón Eduardo Mendoza Méndez
Instituto Tecnológico de Celaya
10030925@itcelaya.edu.mx

Resumen

En consideración de la constante actualización de las tecnologías emergentes que la práctica cotidiana requiere dentro de este campo, el presente trabajo presenta un compendio que resume una muestra de aplicaciones de apoyo al desarrollo e ingeniería de programas de cómputo, denominadas herramientas CASE, por sus siglas en inglés, tomando en consideración también un enfoque didáctico y la contribución que estas brindan al gestor o líder de proyecto de desarrollo de sistemas, se han agrupado por tipo de tecnología y en categorías de acuerdo a las distintas etapas que posee el ciclo de vida de un software, así como su colaboración en la metodología a implementar.

Palabra(s) Clave(s): Desarrollo de software, metodología, ciclo de vida, CASE

Introducción

CASE es el acrónimo de Computer Aided Software Engineering, ó ingeniería de software asistida por computadora, y se refiere al uso de programas de cómputo para organizar y administrar el desarrollo de software, sobre todo en aquellos proyectos donde se involucre una cantidad considerable de recursos y personal, tarea que para el gestor de proyectos representa una inversión considerable de tiempo, por tal motivo estas herramientas tienen como propósito el fungir como apoyo durante el ciclo de vida de todo el proyecto incluyendo todos sus componentes. El uso de estas herramientas también debe ser categorizado dentro del contexto del tipo de tecnología en la que se encuentre disponible la herramienta, así como por el propósito de su uso, para este trabajo considera la clasificación en función de la fase del ciclo de vida: Upper (administradores de proyectos), middle (herramientas de diseño) y Lower (generadores de código, repositorios, reporteadores.

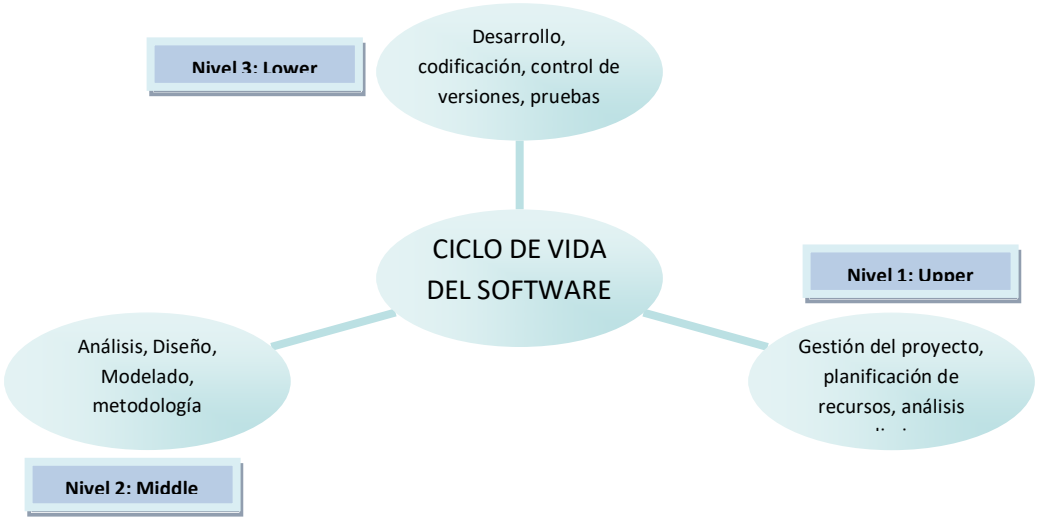

Fig. 1. Las herramientas case por su función en el ciclo de vida son agrupadas como Middle, Upper y Lower

Es importante señalar que los niveles del ciclo de vida en este caso se refieren a las etapas en las que el software se encuentra en desarrollo y que comprenden hasta las pruebas e implementación.

Aunque las herramientas se agrupan por la etapa del ciclo de vida en la que son utilizadas, es conveniente también, categorizarlas en base a otros aspectos que permitirán identificar al gestor de proyectos de software las herramientas en base a su tecnología y tipo de licencia ya que este es un factor fundamental para seleccionar entre un conjunto de herramientas que son de niveles iguales, definiendo una descripción para cada categoría:

Por su tecnología: Las tecnologías de información emergentes se encuentran en constante evolución al paso del tiempo brindando cada vez servicios de mayor movilidad y portabilidad, en la actualidad se encuentran herramientas locales u offline que son las que se instalan en el equipo cliente y que son ejecutadas en el mismo lugar donde son utilizadas, generalmente aplicaciones de funcionalidad compleja y alto consumo de recursos. Las herramientas cliente-servidor que son las que instalan una pequeña instancia del software en el equipo del cliente, pero que requieren comunicación constante con un equipo remoto servidor, estas generalmente tienen funcionalidad asistida y es donde se ubican las aplicaciones móviles, y todas aquellas que necesiten tener una interacción constante con sus servicios, Las herramientas en línea o en internet que no requieren una instalación del cliente que son ejecutadas en el equipo cliente pero físicamente ubicadas en un servicio en la nube (Cloud Computing) [1], generalmente con funciones y características básicas y limitadas.

Por su Licencia: Antes de comenzar a operar una herramienta es importante documentarse acerca del tipo de licencia (CLUF) del software que se va a utilizar, ya que el software es una propiedad intelectual que está sujeta a un contrato de arrendamiento que se realiza antes de instalarse o utilizarse, este convenio puede o no requerir la cobertura de una cantidad económica por el pago de los derechos de uso, instalación, edición del producto en cuestión [2]. así mismo esta variará en cuanto a si el código fuente está abierto(GNU)[3] para su edición, evaluación y mejora por parte del

mismo usuario, o si está cerrado, es decir, si el código fuente del programa esta restringido para uso privado y solo se puede conceder el uso del producto terminado, Existen diversos acuerdos de licencia, esta será de acuerdo a la herramienta que se esté considerando utilizar, para el caso de las que aquí se mencionan se encuentran: La licencia o contrato para software comercial, que es el convenio para usar el producto software a código cerrado, con los fines que el contrato establezcan, y por los que hay que pagar una cantidad económica para poder tener acceso a las características de las herramientas como uso, instalación, compartición, uso con fines de lucro. La licencia o contrato para uso de software libre o de código abierto que faculta al usuario a tener una copia del software y de su código fuente para poder evaluarlo, usarlo y en algún momento de acuerdo al contrato poder modificarlo y adaptarlo a las nuevas necesidades de una comunidad o grupo de usuarios, por tal motivo este software no puede ser comercializado, ya que el propósito de esta licencia es el de ir agregando nuevas funcionalidades, valiéndose de la experiencia de los mismos usuarios, en el caso de las plataformas móviles o en línea el contrato de licencia permite el uso y acceso a una versión restringida y limitada del software poniendo las características completas a disposición del cliente para su descarga e instalación o para su uso online en la nube mediante el pago de un derecho o la suscripción a una membresía proporcionada por el fabricante, y que permitirá el uso de la plataforma de manera ilimitada.

Para el desarrollo del presente trabajo se considerarán algunas herramientas de cualquiera de los tipos antes mencionados haciendo referencia en el impacto que poseen en el proceso de gestión de proyectos de software.

Desarrollo

Como su nombre lo sugiere una herramienta debe de funcionar para potenciar la práctica de una actividad, se comienza con el inicio del ciclo de vida que es la gestión del proyecto y la planificación de recursos, para esta primera etapa que es el upper o nivel uno las características deseables de una herramienta pudieran ser las siguientes:

Gestión de múltiples proyectos: gestionar múltiples proyectos desde una sola interfaz. Cada proyecto puede tener una configuración totalmente diferente y el usuario tener un rol distinto en cada uno. Dentro de cada proyecto pueden definirse varios subproyectos.

Personalización de proyectos: El proyecto debe ser totalmente personalizable, pudiendo encontrar proyectos muy distintos entre sí según sus objetivos. Lo más importante son los módulos que se pueden desactivar o activar para cada proyecto: peticiones, control del tiempo, documentos, archivos.

Sistema flexible de seguimiento de tareas: Una de las mecánicas más útiles para el desarrollo de un proyecto son las peticiones y su visualización. Estas peticiones se pueden ser errores, tareas o soporte y deben poder asignarse a un miembro del proyecto. Indicar una fecha de inicio y fin para cada petición, e incluso llevar un control del tiempo y porcentaje realizado. También se le puede asignar una prioridad, antecesores y sucesores.

Uso de calendario y diagrama de Gantt: Debe incluir un calendario para visualizar todas las peticiones a lo largo de un periodo elegido, marcando claramente el día de inicio y de fin de cada petición. Igualmente ocurre con la vista en diagrama de Gantt, que va marcando el porcentaje completado conforme avanzan los días. Las peticiones que se visualizan en ambos casos deben estar sujetas a la personalización que el usuario decida utilizar

Notificaciones: Configurando previamente un servidor de correo electrónico automatizado debe permite enviar notificaciones por correo electrónico en todos los proyectos, definiendo antes los eventos que activan estos avisos.

Exportación a distintos formatos: Los informes de peticiones que pueden generarse añadiendo filtros, y que permiten visualizar las diferentes tareas de un proyecto, pueden exportarse en PDF o formato CSV, pudiendo así imprimirlos posteriormente en un formato organizado.

Las herramientas que cumplen con estas características deseables pueden ser:

Redmine: que es una aplicación cliente servidor web, que se instala en un servidor de proyectos, maneja la licencia GNU/GPL2.0 [3] y permite que el administrador de proyectos asigne una cuenta de usuario a cada integrante del equipo, de esta manera se puede proporcionar una solución integral de apoyo para las actividades del nivel 1, es portable para sistemas operativos Windows y Linux, e incluso un autoejecutable que configura automáticamente el servidor con un instalador offline

Project: Es una herramienta de licencia comercial, funciona en un entorno de servidor de proyectos en ambientes Windows, permite la integración con otras tecnologías del mismo fabricante del software así como con la Microsoft Office, su instalación es automática con un servidor offline.

Open Project: La versión GNU/GPL2, de código abierto de project siendo mas bien una aplicación de escritorio offline local que permite el manejo de las actividades del nivel, esta plataforma está disponible para Linux y Windows

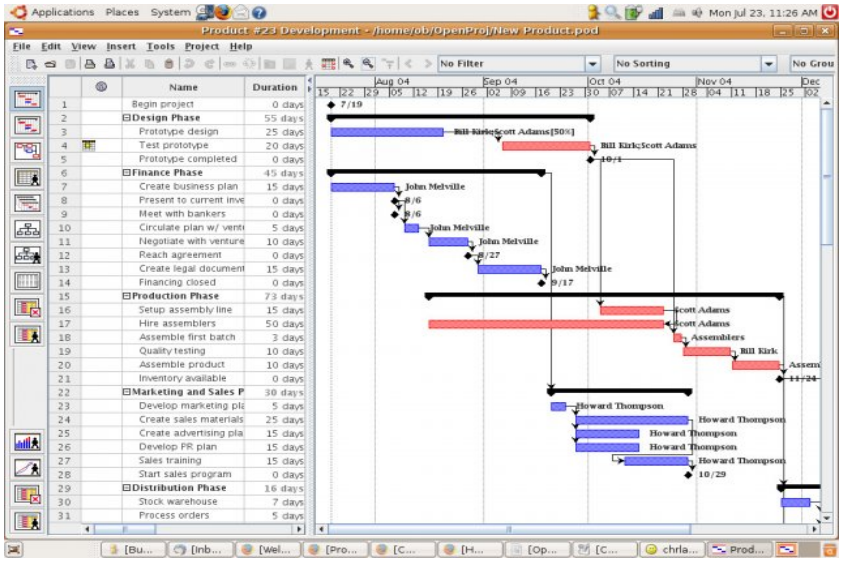

Fig. 2. Diagrama De Gantt Realizado con Open Project en el sistema operativo Linux Ubuntu

En el nivel dos o middle se ubican las actividades más importantes de todo desarrollo de software y que se refiere a la etapa de diseño y modelado, dentro de este contexto

se ubican una gran cantidad de herramientas, incluso existen algunas que permiten atravesar al siguiente nivel generando código a partir de los diseños, por tal motivo son herramientas muy atractivas, dentro de las características deseables en este nivel se encuentran.

Diseño Orientado a Objetos UML UML (Unified Modeling Language) Consiste en una metodología basada en el lenguaje de modelado que permite, construir y documentar el conjunto de elementos que forman un desarrollo de software orientado a objetos. Fue desarrollado en 1995 por: Grady Booch, Ivar Jacobson y Jim Rumbaugh. Estos autores fueron contratados por la empresa Rational Software Co. para crear una notación unificada en la que basar la construcción de sus herramientas de de CASE. Esta notación ha sido ampliamente aceptada debido al prestigio de sus creadores y debido a que incorpora las principales ventajas de cada uno de los métodos particulares en los que se basa Con UML se fusiona la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros software que trabajan en el desarrollo orientado a objetos. Un modelo representa a un sistema software desde una perspectiva específica. Al igual que la vista de planta y capas de una figura en dibujo técnico nos muestran la misma figura enfocada desde distintos ángulos, cada modelo entrega un panorama con un aspecto distinto del sistema.

Adaptar el proceso: Cada proceso deberá adaptarse a las necesidades de cliente considerando como lo más importante la interacción con éste, con enfoque hacia las características propias del proyecto u organización, el tamaño del mismo, así como su tipo o las regulaciones que lo condicionan e influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto en un área subformal para hacer un proceso de satisfacción del software.

Elevar el nivel de abstracción: Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, modelo vista controlador, lenguajes visuales, o marcos de referencia (frameworks) por nombrar algunos. Esto evita que los ingenieros de software vayan directamente de los requisitos a la codificación de software a la medida del cliente

Diseño de la base de datos: Para el desarrollo de un software es importante que se pueda emplear una herramienta que permita generar las bases de datos que se utilizaran dentro del proceso, pero también es importante que este programa le permita manipularla, consultarla y crear diagramas que puedan ser consultados y modificados de acuerdo a como se avanza en la etapa de desarrollo.

El software que se utilizan en la práctica docente y que cumplen alguna de estas características deseables son:

Gliffy.com Es un sitio web que permite el uso de una herramienta en la nube para el modelado y desarrollo de diagramas UML, entidad relación, DFD, e incluso diagramas de otros proyectos como de redes de computadoras y de procesos industriales, el software tiene membresía comercial es decir es de uso limitado hasta que se adquieren los derechos de uso extendido y que facultan al desarrollador a tener acceso a todas las características del software, los diseños son descargados de la plataforma en cualquier formato como PDF,PNG,JPG etc..

Power Designer: Es una herramienta de modelado colaborativo producida por la empresa Sybase, es de licencia comercial por tal motivo su uso es mediante pago de derecho, el software se instala localmente en el equipo del desarrollador y permite la elaboración de diagramas UML, entidad relación, DFD, con accesorios o características especiales que permiten la generación del código para crear los objetos que son diseñados con esta plataforma, también proporciona como valor agregado la posibilidad de aplicar la ingeniería inversa es decir a partir de un producto o un código obtener su modelado y diseño

DB Schema: Es una herramienta de diseño y modelado de bases de datos, la cual permite crear los esquemas o diagramas de los modelos entidad relación, e incluso ir más allá con diagramas de bases de datos orientadas a objetos, es una herramienta comercial con un costo de licencia de uso y una cantidad de equipos a compartir restringida, otra de sus capacidades es la de poder generar el código SQL que crea la base de datos física que es diseñada gráficamente.

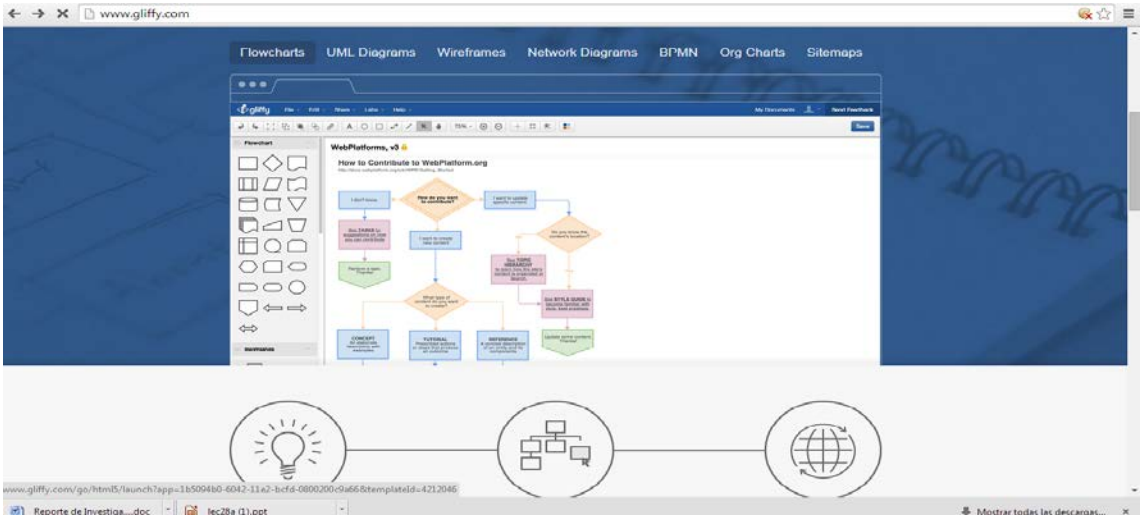

Fig. 3. Pantalla de inicio de la tecnología en la nube de gliffy.com

Para el nivel de lower o la generación de código, es decir el último nivel consistente en la codificación, las herramientas que existen permiten el auxilio en la gestión del código y módulos del software, en este apartado se encuentran las siguientes características deseables

Generación de código: los frameworks permiten que a partir de un modelado (MVC) deben contener un generador de código y de formularios para los procesos que ya se encuentran definidos, así mismo los IDE que mediante herramientas de arrastre drag & drop permiten modelar formularios de software de manera automática

Manejo de versiones y repositorios: Cuando la codificación se está realizando entre varios integrantes de un equipo que comparten módulos, los conceptos de cohesión y acoplamiento permiten que un equipo pueda tener el código general en un repositorio central, con el propósito de permitir ir haciendo modificaciones a las versiones de manera ordenada y sincronizada

Pruebas: Las herramientas de beta testing son funcionales cuando el software se encuentra en su fase final de desarrollo ya que permiten generar circunstancias extremas para la prueba de las condiciones del diseño, pruebas como caja negra, caja blanca, debugging, eficiencia, complejidad son o deben ser incorporadas a la herramienta, por tanto una característica deseable de esta fase es que el software contenga diversos tipos de pruebas en una sola interfaz.

Las herramientas disponibles para este nivel son:

Framework Yii: Es un framework especializado en el MVC que permite al desarrollador mediante un diseño de base de datos construir modelos que serán utilizados por controladores y vistas generadas automáticamente, el framework permite incorporar incluso tecnologías como Bootstrap o Jquery así como su uso de manera nativa con PHP, el software es de código abierto y es de tipo local.

Net Beans: Es un entorno de desarrollo integrado IDE que permite la gestión de desarrollos de software asi como la compilación de programas, para las herramientas como java, ofrece una interfaz de diseño grafica que permite arrastrar componentes y generar automáticamente el código de los objetos con el fin de que el programador solo tenga que programar el código de las acciones.

Git: Es un software de repositorio que permite crear en un nodo o servidor central un origen donde se encuentra el código que podrá ser editado por los desarrolladores accediendo mediante una terminal previamente configurada puede tener acceso vía bash o línea de comandos UNIX y mediante un cliente que puede ser accediro localmente o por navegador web (smartgit).

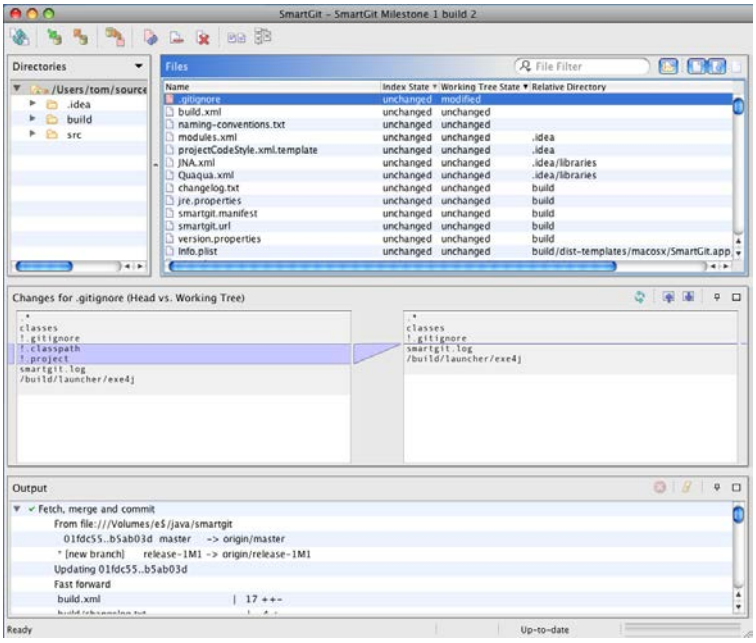

Fig. 4. Pantalla administración de Smart git con una conexión hacia un repositorio

Git: Es un software de repositorio que permite crear en un nodo o servidor central un origen donde se encuentra el código que podrá ser editado por los desarrolladores accediendo mediante una terminal previamente configurada puede tener acceso via bash o línea de comandos unix y mediante un cliente que puede ser accediro localmente o por navegador web (smartgit).

The Test Environment Toolkit: Es un software de pruebas o beta testing, de código abierto que permite al equipo de desarrollo implementar un plan general de debugging, permite generar datos aleatorios, sobrecargar base de datos, hacer pruebas de complejidad a los algoritmos y de caja blanca a los formularios, agrupa varias características de prueba y benchmarking de procesos.

Conclusiones

El conjunto de herramientas CASE es muy extenso, con el evolucionar de las mejores prácticas en la ingeniería del software la oferta se incrementado, proporcionando al desarrollador o ingeniero de software ventajas competitivas que son apreciables incluso en los productos terminados, el uso de las tecnologías complementado con una metodología de desarrollo basada en un enfoque de calidad permite el ir incrementando el nivel de madurez en el proceso o desarrollo de las habilidades de gestores de proyectos de software.

El uso de las herramientas se debe inculcar en la práctica docente a los estudiantes desde su etapa formativa y analizar el beneficio que representa para ellos que potencialmente, serán gestores de proyectos de software e incluso realizar prácticas de laboratorio donde el estudiante pueda tener interacción con el software y proponer nuevas tecnologías.

Bibliografía

[1] IBM Rational Unified Process Reference and Certification Guide: Solution Designer, Ahmad K. Shuja and Jochen Krebs, IBM Press, USA 2008

[2] Ingeniería del software 7ª Edición, Roger Pressman, Mc Grawhill, USA,2010.

[3] A Software Project Management for Small to Medium Sized Project. Rakos, John J.Ed. Prentice Hall, USA 1990.

[4] The Unified Software Development Process, Jacobson, Booch, Rumbaugh, Addison-Wesley, USA 1999

Referencias Web:

[1]http://www.microsoft.com/es-mx/movil/software/agreement/

[2]http://www.gnu.org/licenses/licenses.es.html

[3]http://www.gnu.org/licenses/old-licenses/gpl-2.0.html