URKUND

Urkund Analysis Result

Analysed Document:	propuesta_Huaraca_version_urkund20181112204352.docx (D43931423)
Submitted:	11/13/2018 1:29:00 PM
Submitted By:	dverap@unemi.edu.ec
Significance:	0 %

Sources included in the report:

Instances where selected sources appear:

0

INTRODUCCION

Las Herramientas CASE desde sus inicios en los años setenta empezaron con un simple procesador de palabras diseñado para generar y manipular documentación, continuando con

diagramas de flujo y métodos gráficos que con el tiempo era necesario realizar sus respectivos cambios debido a que estos eran muy complejos. A través del tiempo las herramientas CASE permitieron crear y modificar los diagramas, logrando incrementar

la calidad al desarrollar el software, que pronto se desarrollaron herramientas graficas integradas con diccionarios de base de datos, finalmente se desarrollaron Herramientas Case para ayudar e cada etapa del ciclo de desarrollo de software. El objetivo principal de las Herramientas CASE es el aumento en la productividad, debido a que esta permite generar de forma automática los códigos, logrando la mejora de calidad y rendimiento al desarrollar software. En la actualidad las Herramientas CASE permiten automatizar todo el ciclo de vida de desarrollo de software. En las organizaciones se presenta con una evolución tecnológica y de gran potencial, se debe tener claro las herramientas CASE tomando en consideración que se debe tener en cuenta las facilidades que presentan las Herramientas CASE al adquirirlas.

CAPITULO I

PROBLEMA DE INVESTIGACION

En la actualidad el objetivo principal de los desarrolladores de software es elaborar aplicaciones de calidad, con costos bajos, tiempo mínimo y permitiendo que las aplicaciones desarrolladas tengan compatibilidad con los diferentes sistemas operativos, bases de datos y lenguajes de programación. Para cumplirlo existen herramientas capaces de solucionar, consolidar, acoplar y unir los diversos requerimientos de los clientes. La principal es el uso de Herramientas CASE, la cual permite desarrollar aplicaciones en un entorno gráfico, manteniendo una metodología de programación ordenada y enfocada al desarrollo de sistemas basado en los estándares de la ingeniería de software, logrando desarrollar sistemas de alto nivel competitivo. El desarrollo de soluciones informáticas debe ser adaptable y no enfocarse en una sola base de datos y un lenguaje de programación específico, debe ser capaz de ejecutarse en diferentes arquitecturas sin la necesidad de realizar cambios extremos en su sistema maestro. Para entenderlo de mejor manera presento el siguiente ejemplo: Un cliente A necesita un sistema de contabilidad desarrollado en Visual Basic, Sql Server y Windows. El cliente B necesita el mismo sistema pero en un sistema operativo diferente y un tercer cliente necesita el mismo sistema pero usando Java y en Linux. Para satisfacer las demandas de los clientes tenemos dos opciones la primera contratar tres desarrolladores diferentes expertos en cada arquitectura requerida, la segunda opción es desarrollar un solo sistema con el uso de herramientas CASE que sea capaz de generar las aplicaciones cumpliendo los requerimientos mencionados. La solución más óptima es la segunda, pues permite elaborar un solo sistema o prototipo para cada arquitectura requerida, el uso de las herramientas CASE permiten generar aplicaciones multilenguaje, multiplataforma y en base de datos diferente.

OBJETIVO GENERAL Analizar la importancia de las Herramientas CASE en el desarrollo de software.

OBJETIVO ESPECIFICO • Evaluar como las herramientas CASE aumenta la productividad en las áreas de desarrollo de los sistemas informáticos. • Evaluar de qué manera inciden en la mejora de la calidad del software desarrollado. • Determinar como el desarrollo mediante herramientas CASE permite reducir tiempo y costos.

JUSTIFICACION El desarrollo de sistemas avanza a gran escala por tal motivo en la actualidad existen muchos lenguajes de programación, bases de datos y arquitecturas diferentes. Para facilitar el desarrollo de soluciones informáticas las herramientas CASE permiten automatizar los procesos en el desarrollo de software, dando como resultados sistemas compatibles en el mercado actual. Si se desea satisfacer la demanda actual de soluciones informáticas en tiempos cortos, reduciendo costos y con calidad, las herramientas CASE es una de las opciones más optimas porque permite realizar todo lo mencionado, es importante conocer los conceptos y formas de desarrollo convencional de software son de mucha importancia, los desarrolladores deben conocer sólidamente las bases al desarrollar, para tener una concepción adecuada sobre la automatización de los mismos por parte de las herramientas CASE.

CAPITULO II

MARCO TEORICO CONCEPTUAL

DEFINICION DE HERRAMIENTAS CASE

Una Herramienta CASE es un programa especializado en el control y desarrollo de aplicaciones informáticas, siguiendo alguna de las metodologías más extendidas como los diagramas de control de flujo de Yourdon o los diagramas Entidad / relación de P.P. Chen para la normalización de bases de datos. En grandes equipos de desarrollo, el sistema queda centralizado a través de un diccionario de datos o "repositorio" que coordina los desarrollos de todos los participantes CITATION MAR18 \l 3082 (MONTES, 2018).

EL PROCESO DE DESARROLLO DE SOFTWARE

El proceso de desarrollo de software consiste en una serie de pasos bien definidos, que seguidos adecuadamente, conducen a un software mantenible y bien diseñado, aun así, muchas organizaciones olvidan las fases de análisis y diseño a favor de comenzar inmediatamente la implementación de código. CITATION Mar10 \l 3082 (Rangel, 2010) Es más positivo pensar en el desarrollo de software, no como un proceso lineal, sino como un ciclo, aunque el paso de una fase a otra se realiza en un sentido, también pueden existir vueltas atrás en determinados momentos, especialmente cuando aparecen requerimientos de usuario ocultos en las primeras fases, a continuación se muestra cuáles son las principales facetas para desarrollar software. (Caceres & Cruz, 2006) Análisis de Requerimientos, Diseño de la Especificación (Prototipo), Implementación (Producción), Integración, Test y Documentación., Mantenimiento y Reingeniería.(Sommerville, 2011a)

CLASIFICACION DE LAS HERRAMIENTAS CASE

Las Herramientas CASE por su complejidad, no tienen una clasificación específica de sus tipos, varios autores las clasifican de diferente manera de acuerdo a la forma de ver e interpretar las cosas. Una de las más importantes clasificaciones de Herramientas CASE se observa en la figura 1.(Chavarría & Ocotilla, 2016) CLASIFICACION GENERAL DE LAS HERRAMIENTAS CASE

TOOLKIT WORKBENCH

POR SU AMPLITUD

UPPER CASE MIDDLE CASE LOWER CASE

POR LAS TAREAS QUE AUTOMATIZAN

Figura. 1 Clasificación general de las Herramientas CASE

POR SU AMPLITUD * TOOLKIT:

se defina a la colección de herramientas integradas en las que permiten mecanizar un conjunto de tareas en ciertas fases de un sistema informático.(Carrillo Sarabia, 2008) * WORKBENCH: estas son herramientas de soporte que permite la automatización del proceso completo en el desarrollo mediante una via informática(

Grado et al., 2015) POR LAS TAREAS QUE AUTOMATIZAN • UPPER CASE:

sirve como un planificador estratégico en las que efectúa

desarrollo funcional de planes corporativos.(Grado et al., 2015) • MIDDLE CASE: Análisis y diseño, como Designer de Oracle. CITATION Zuñ12 \l 3082 (Gallardo, 2012) • LOWER CASE: Generación de código, test e implantación, como Genexus de Artech.(Zapata & Chaverra, 2010)

OTROS TIPOS DE CASE Como se muestra en la figura 2 las Herramientas CASE también pueden clasificarse de la siguiente manera: (Universidad Pontificia de Valencia, 2010)

Figura. 2 Clasificación de las Herramientas CASE

POR LAS PLATAFORMAS QUE SOPORTAN: Case Uniplataforma y Case Multiplataforma.(Grado et al., 2015)

POR LAS FASES DEL CICLO DE VIDA QUE CUBREN

• CASE integrado: estas permiten abarcar las distintas fases del ciclo de sistemas, también conocidas como workbench CASE CITATION Mar11 \l 3082 (Salas, 2011) • TOOLKIT. Herramientas que cubren alguna de las partes del desarrollo.(Grado et al., 2015)

POR LA ARQUITECTURA DE APLICACIONES QUE GENERAN

Case Centralizadas, Case Cliente Servidor (Dos Capas) y Case Cliente Servidor (Multicapa). (Grado et al., 2015)

VENTAJAS Y DESVENTAJAS DEPENDIENDO DEL TIPO DE HERRAMIENTA CASE UTILIZADA TIPO VENTAJAS DESVENTAJAS

Upper CASE

Se utiliza en arquitecturas para PC y

es aplicable en diferentes entornos

Menor Costo

Mejora la calidad pero no la productividad.

Permite la integración del ciclo de vida.

Lower CASE

Mejora la

productividad a corto Plazo.

Buen soporte al mantenimiento.

No garantiza la persistencia en niveles corporativos.

No garantiza la eficiencia de análisis y diseño.

No permite la integración del ciclo de vida.

I – CASE

Integra el ciclo de Vida.

Mejora la productividad a mediano plazo.

Buen soporte de mantenimiento.

Mantiene la persistencia en niveles corporativos.

No es eficiente para niveles simples, sino para complejos.

Depende del hardware y software.

Costos elevados de implementacion. (Instituto, 1999)

CASE EN EL CICLO DE VIDA DE UN SISTEMA

En la figura 3 se puede observar qué partes del ciclo de vida pueden generar los diferentes tipos de Herramientas CASE. (ORTEGA & SANTA VILLA, 2012)

Programación y puesta apunto Mantenimiento Upper CASE Middle CASE Lower CASE Diseño detallado Ciclo de Vida Figura. 3 Tipos de Herramientas CASE y en qué Parte del Ciclo de Vida Actúan. Creación de dispositivo de estudio Análisis detallado Análisis global MODELO DE IMPLANTACIÓN MODELO DETALLADO MODELO GLOBAL Fase 4 PRODUCCIÓN Fase 3 DESARROLLO Fase 2 DECISIÓN Fase 1 PREVIA

Implementación

(Grado et al., 2015)

ESTRUCTURA DE LAS HERRAMIENTAS CASE

Es muy difícil especificar cuál es la estructura exacta de una Herramienta CASE, ya que no todas tienen las mismas características unas de otras, a continuación se presenta la estructura más general a manera de niveles o módulos: Módulo de Repositorio, Módulo de Diagramación y Modelamiento, Módulo de Prototipado, Módulo de Generación de Código y Módulo de Generación de Documentación. (Chavarría & Ocotilla, 2016)

MODULO DE REPOSITORIO O ENCICLOPEDIA En el contexto de las Herramientas CASE se entiende por enciclopedia a la base de datos que contiene toda la información relacionada con: las especificaciones, análisis y diseño de la aplicación, definiendo a cada objeto de la siguiente manera:(Universidad Pontificia de Valencia, 2010) •

Datos: Elementos, atributos (campos), asociaciones (relaciones), entidades (registros), almacenes de datos

y estructuras.(Pare & Casillas, 2005) • Procesos: Procesos, funciones y módulos.(Grado et al., 2015) • Gráficos: Diagramas de flujo de datos, diagramas entidad relación, diagramas de descomposición funcional, diagramas de estructura de árbol y diagramas de clases.(Caceres & Cruz, 2006) • Reglas: Reglas de gestión de métodos y comportamiento de datos y procesos. (Grado et al., 2015)

Las herramientas Case tiene como base de datos principal a un repositorio , debido a que este permite abarcar toda la información desde el inicio de la

vida de dicho sistema. En algunas referencias se le denomina Diccionario de Recursos de Información y en otros casos se le llama Base de Conocimiento, si a una Herramienta CASE se la considera como un Sistema Experto, el repositorio sería la fuente de alimentación de la información.(Carrillo Sarabia, 2008)

Debido a que la mayoría de H.C. tienen un repositorio,

es preferible trabajar con herramientas que tengan su propio repositorio ya que se adaptan a la tecnología propia de la herramienta y no tienen que heredar metodologías de otro fabricante.(Grado et al., 2015)

CARACTERISTICAS DE UN REPOSITORIO Tipo de Información: Nos permite formar: datos, gráficos, procesos, informes, modelos y reglas.(Grado et al., 2015) Tipo de Controles:

se encarga de incorporar modulos de gestión logrando su mantenimiento de versiones, información o acceso por claves.(Carrillo Sarabia, 2008) Tipo de Actualización: los cambios que se efectúen en losdiseños se verán reflejados en tiempo real en

el repositorio.(Grado et al., 2015) Reutilización de Módulos para Otros Diseños: la clave para poder localizar, identificar para extraer código

es mediante el repositorio. CITATION MAR18 \l 3082 (MONTES, 2018)

INTERFASES AUTOMÁTICAS CON OTROS REPOSITORIOS Y BASES DE DATOS EXTERNAS. Ayudan a la migración de aplicaciones, bases de datos, datos, y en si a la migración de toda la aplicación, ya sea como procesos de reingeniería o como procesos de cambios de versión de la herramienta, se toma como procesos de reingeniería a las aplicaciones y bases de datos realizadas con otras herramientas y se las acopla a la tecnología de una Herramienta CASE y los procesos de cambio de versión actúan cuando se quiere migrar aplicaciones de una versión más antigua a una versión más actual, sin que esto lleve a un cambio de fondo en la programación de la aplicación; en la figura 4. Se observa el módulo de repositorio.(Grado et al., 2015)

Paneles de trabajo Figura. 4 Diagrama de un Repositorio

Base de Datos Repositorio Base de Conocimiento Menús Procedimientos Reportes Transacciones

(Grado et al., 2015)

MODULO DE DIAGRAMACION Y MODELAMIENTO Es la parte del sistema, donde el programador diseña la realidad, dicho de otra manera diseña lo que desea ver propiamente en su programa, luego de esto la Herramienta CASE automáticamente genera los diferentes diagramas, como: Diagramas de flujo de datos, Diagramas entidad relación, Historia de vida de las entidades, Diagramas de estructura de datos, Diagrama de estructura de cuadros, Diagramas de estructura de árboles y Técnicas matriciales.(Martínez, 2013)

MODULO DE PROTOTIPADO Esta herramienta permite revelar al usuario desde el inicio del diseño, permitiendo

que el software cuente con la mayoría de las necesidades que pueda tener el usuario, antes de que la aplicación entre a producción.(Sommerville, 2011b) El programador diseñara el software conjuntamente con el usuario, haciéndole ver como se van a presentar en pantalla los criterios requeridos por él. En la figura 5 se observa el ciclo de vida de diseño, prototipo y producción. (Chavarría & Ocotilla, 2016)

Figura. 5 Ciclo de Vida: Diseño – Prototipo – Producción. Producción Prototipo Diseño

(Grado et al., 2015) La herramienta será más

efectiva cuando consiga la implicación del usuario final en el diseño de aplicación la construcción del prototipo.(Grado et al., 2015) Los prototipos fueron diseñados con la finalidad de utilizarse en el desarrollo de sistemas tradicionales tomando en cuenta que estos proporcionan una realimentación al instante ayudando de esta manera al sistema. Las HC están efectuadas para la creación de prototipos con seguridad y

con rapidez. En el ciclo de prototipo, el diseñador recorrerá repetidamente el bucle Diseño – Prototipo durante la fase de diseño, construyendo y probando sucesivos prototipos del modelo. CITATION Ern17 \l 3082 (Valle, 2017) En el ciclo de producción por el contrario, pasará con menos frecuencia el bucle Diseño Producción, ya que la generación del sistema se realiza solamente cuando el prototipo ha sido totalmente aprobado, o luego de haber instrumentado y probado algún cambio. (Lopez, Gonzalez, & Gallud, 2010) MODULO DE GENERADORES DE CODIGO

Estos son utilizados normalmente en ordenadores de estaciones de trabajo o personales.

Herramienta CASE genera código automáticamente en diferentes lenguajes.(Grado et al., 2015)

MODULO GENERADOR DE DOCUMENTACION

Este módulo se suman la ayuda en línea, los help o las ayudas del programa.(Instituto, 1999)

CAPITULO III

METODOLOGIA

El marco metodológico utilizado en la presente investigación permitió analizar y profundizar el tema propuesto, haciendo uso de la investigación descriptiva e investigación documental.

• Investigación descriptiva: Permite describir completamente la investigación realizada por medio de la información obtenida de las diferentes fuentes bibliográficas.

• Investigación Documental: Este tipo de investigación nos permitió recopilar datos e información a través del uso de técnicas e instrumentos de recolección con la finalidad de analizarlos e interpretarlos.

Las herramientas de investigación utilizadas principalmente el análisis de documentos nos permite obtener información amplia de nuestro problema de investigación, planteando lo más relevante del tema a investigar, para su posterior análisis y verificación.

CAPITULO IV

DESARROLLO DEL TEMA En la actualidad existen una gran variedad de Herramientas CASE una de las más completas y la cual propongo para el desarrollo de software es GENEXUS porque cubre todo el ciclo de vida en el desarrollo de software, la cual está dentro de las herramientas tipo WORKBENCH. Desarrollado por un equipo de investigadores Uruguayos, y está patentado por la empresa Uruguaya ARTECH, actualmente distribuido en Ecuador por GMS (Grupo

Microsistemas). El objetivo de Genexus, es ayudar a los desarrolladores a elaborar sus aplicaciones en el menor tiempo y con la mejor calidad posible, omitiendo ciertos pasos que son automatizables, y concentrándose en las partes verdaderamente complejas como son el análisis y diseño. Desde el punto de vista teórico, Genexus es más una metodología para desarrollar aplicaciones que una herramienta de software. Está relacionado con las metodologías tradicionales, pero aporta grandes y novedosos cambios a las mismas. Al comenzar las aplicaciones por el desarrollo de la interfase de usuario, revoluciona el mundo de la metodología tradicional, ya que el programador no tiene que preocuparse por la implementación física del sistema como: creación y normalización de las bases de datos, generación de programas, generación de diagramas, generación de documentación, etc. El punto de partida de Genexus, es “DESCRIBIR LAS VISIONES DE LOS USUARIOS O DISEÑO DE LA REALIDAD”; a partir de este modelo mediante una “síntesis de visiones” se construye el soporte computacional que es el eje de la aplicación. En la mayoría de metodologías tradicionales como: análisis estructurado, ingeniería de la información, análisis esencial, etc. Se separan el análisis de datos, de los procesos y programas; Genexus hace un análisis conceptual unificado que se lo conoce como “metodología incremental”. Cuando se comenzaron a utilizar verdaderos modelos corporativos que normalmente poseen cientos de tablas, se confirmó que no debía perderse más tiempo buscando algo que no existe: las bases de datos estables. Luego de varias investigaciones, se desarrolló una teoría la cual fue crear una herramienta que pudiera posibilita el desarrollo incremental y permitir la convivencia con las bases de datos reales, la figura.6 explica el crecimiento del modelo incremental.

Figura.6 Metodología Incremental Definición Inicial

La construcción automática de la base de datos y programas ejecutables a partir de una única especificación, permitirá a Genexus utilizar la metodología incremental; este proceso se realiza mediante “aproximaciones sucesivas”.

En cada momento se desarrolla la aplicación con el conocimiento que se tiene, y luego cuando se pasa a tener: más, menos o nuevo conocimiento; Genexus se ocupará de hacer automáticamente todas las nuevas adaptaciones en las bases de datos y programas.

Utilizando Genexus

la tarea básica del analista es la “descripción de la realidad”.

Solo el hombre podría realizar esta tarea, ya que solo él puede entender los problemas del usuario. Desde luego que esto modifica la actividad del analista e incluso su perfil supuestamente óptimo ya que lo transforma en un Busines Analyst o analista del negocio.CITATION Gen18 \l 3082 (Genexus, 2018)

Ahora trabaja en alto nivel, discutiendo problemas con el usuario y probando con el las especificaciones a nivel de prototipo, en vez de desarrollar su actividad en tareas de bajo nivel como: diseñar archivos, normalizar, diseñar programas, programar, buscar y eliminar errores de programación, etc.

DESARROLLO TRADICIONAL

Para empezar a analizar la estructura de las Herramientas CASE generadores de aplicaciones distribuidas, se tomará como patrón a JAVA, ya que es el lenguaje de programación pionero y más versátil para la realización de aplicaciones distribuidas. Nov 2018 Nov 2017

Change

Programming Language

Ratings

Change

1

1

Java 16.746%

+3.51%

2

2

C 14.396%

+5.10%

3

3

C++ 8.282%

+2.94%

4

4

Python 7.683%

+3.20%

5

7

Visual Basic .NET 6.490%

+3.58%

CITATION TIO18 \l 3082 (TIOBE, 2018)

JAVA es el estándar de la mayoría de plataformas de software, sin menospreciar a otras herramientas como la tecnología .NET de Microsoft y otras herramientas existentes en el mercado. Este estudio se va a realizar tomando como referencia a JAVA interactuando con la Herramienta CASE GENEXUS.

DESARROLLO DE APLICACIONES Al desarrollar en Genexus se debe tener claro su estructura posee objetos básicos generados por Genexus que sirven en todo el desarrollo del software: TRANSACCIONES: Las transacciones permiten definir los objetos de la realidad. La mayor parte de las transacciones pueden ser identificadas prestando atención a las entidades que el usuario menciona, como: clientes, proveedores, productos, etc. A partir de las transacciones, Genexus infiere el diseño de la base de datos. PROCEDIMIENTOS: Son los procesos no interactivos de actualización de la base de datos; son conocidos como procesos batch. REPORTES: Son aquellos que recuperan información a partir de los datos almacenados y no los alteran. A los reportes generalmente se los conoce como listados. PANELES DE TRABAJO Y WEB PANELS: Permiten definir consultas interactivas a las bases de datos, ya sea en modo windows o en modo web. MENUES: Son objetos organizadores del resto de objetos Genexus, los cuales sirven para realizar listas desplegables de selección de objetos.

REQUISITOS PARA ADQUIRIR HERRAMIENTAS CASE La primera fase que debe abordarse

de manera metódico dentro de la evolución de

adquisición, es el estudio de las micción existentes, que deberán ser satisfechas a través de la

creación del instrumento que se va a obtener. El comprador debe coincidir lo siguiente: CITATION gob18 \l 3082 (gobierno digital, 2018) • Los principales requisitos funcionales que debe cumplir la herramienta. CITATION gob18 \l 3082 (gobierno digital, 2018) • El tipo de facilidades de uso que debe prestar. CITATION gob18 \l 3082 (gobierno digital, 2018) •

Las limitaciones y restricciones que se derivan del entorno de operación previsto.

CITATION gob18 \l 3082 (gobierno digital, 2018)

REQUISITOS FUNCIONALES En función de los requisitos funcionales se podrá deducir qué tipo de herramienta es la más adecuada. Algunos factores a tener en cuenta y que son comunes en todas estas herramientas son:

CITATION gob18 \l 3082 (gobierno digital, 2018) •

Tipo(s) de plataforma(s) sobre las que deberá funcionar la herramienta, tanto desde el punto de vista del equipamiento lógico como del equipamiento físico. CITATION gob18 \l 3082 (gobierno digital, 2018) • Requisitos físicos (espacio en disco, memoria RAM, UCP, etc).

CITATION gob18 \l 3082 (gobierno digital, 2018) •

Necesidad de integración con otras herramientas de ayuda al desarrollo ya existentes.

CITATION gob18 \l 3082 (gobierno digital, 2018) •

Necesidad de acceso simultáneo para diferentes usuarios. Esto puede enfocar la elección hacia una herramienta que permita accesos compartidos a los datos y que cuente con una definición de perfiles de usuario para la protección de información.

CITATION gob18 \l 3082 (gobierno digital, 2018) •

Necesidad de compartir datos con aplicaciones externas. Se valorará más a aquella aplicación que permita exportar sus datos o que almacene la información en un formato de fácil acceso para otra aplicación.

CITATION gob18 \l 3082 (gobierno digital, 2018)

FACILIDADES DE USO DE UNA HERRAMIENTA CASE Funcionalidad Requerida: Es importante definir con el mayor grado de aproximación, cuáles son las funciones que se le van a pedir a la herramienta. Para ello, es necesario analizar si las necesidades son cubiertas con una herramienta integrada o con una orientada a alguna de las fases del ciclo de vida del desarrollo.

CITATION rep18 \l 3082 (repositorio.utn, 2018)

Metodología Soportada: Si en la organización ya existe una metodología y técnicas, la herramienta deberá soportar dicha metodología, así como las técnicas empleadas en cada fase. Si la Herramienta CASE va a servir precisamente para introducir un nuevo método de trabajo, habrá que asegurarse de que dicho método es el adecuado. En ocasiones, para adaptarse a una metodología, es preciso realizar desarrollos adicionales en la herramienta.

CITATION rep18 \l 3082 (repositorio.utn, 2018)

Generación Automática de Código: En algunos casos la necesidad predominante del usuario puede consistir en la generación automática de código fuente (programas), a partir de productos del diseño fuertemente formalizados (scripts, formatos, etc.). En tal caso, deberán conocerse los pormenores de tal necesidad, como lenguajes de programación admisibles como salida, generación en tiempo real o en un proceso por lotes, etc.

CITATION rep18 \l 3082 (repositorio.utn, 2018)

Capacidad de Integración en la Arquitectura Existente: Habrá que tener en cuenta la plataforma o plataformas diferentes, ordenadores que deberán soportar la Herramienta CASE, su tipología (fabricante, modelo y sistema operativo cuando menos) y las características de la red de interconexión cuando exista. Ello tendrá importancia a la hora de garantizar la compatibilidad del equipamiento existente con los nuevos productos que se van a adquirir. Lo mismo debe hacerse en relación con las herramientas lógicas previamente existentes en esas plataformas, siempre que deban integrarse en mayor o menor medida con los nuevos productos. Se deberá considerar cuáles son los recursos disponibles en el equipamiento existente para la implantación de la Herramienta CASE en cuestión. Deberán conocerse con el

mayor detalle, posibles cuestiones como memoria RAM y espacio en disco necesario, grado de utilización de la(s) UCP(s) en condiciones normales de operación y de picos de demanda de la nueva herramienta. Este punto es importante de cara a un posible redimensionamiento del equipamiento disponible. Estas mismas consideraciones también deben ser tomadas en cuenta no ya para la propia Herramienta CASE, sino para las aplicaciones desarrolladas con ayuda de dicha herramienta.

CITATION rep18 \l 3082 (repositorio.utn, 2018)

Modo de Funcionamiento: Será bueno conocer el modo de funcionamiento: monousuario o multiusuario, así como el grado deseable de centralización de los recursos y funciones asociadas con la administración y operación de la Herramienta CASE que se va a implantar. CITATION rep18 \l 3082 (repositorio.utn, 2018) Personalización del Entorno: Finalmente, deberán considerarse las necesidades o conveniencias de la personalización del sistema, en función de los diferentes perfiles de usuario de la herramienta.

CITATION rep18 \l 3082 (repositorio.utn, 2018)

CAPITULO V

CONCLUSION

La utilización de Herramientas CASE maximiza la calidad en todo el ciclo de vida al desarrollar aplicaciones informáticas mejorando la productividad y automatizando los procesos de creación, normalización y generación de la base de datos, generación de documentación, generación de diagramas, generación de código, generación de pantallas y generación de paneles de trabajo. Permite al programador trabajar bajo una metodología de diseño de software, dejando a un lado la programación de la aplicación, centrándose en el análisis y el diseño, la cual solo puede ser realizada por un humano. Las Herramientas CASE crea una cultura ordenada y sistemática de programación, esto permite reducir costos y tiempo al desarrollar; facilita la generación de software porque permite que cada aplicación generada no se tenga que programar nuevamente todo el software, sino solo realizar pequeñas modificaciones en la base de conocimiento, logrando de esta manera la reutilización del código ya generado, también permite migrar a diferentes lenguajes de programación, bases de datos y plataformas existentes en el mercado. Genexus posee un entorno grafico amigable e interactivo el cual facilita a la generación de software, cuenta con todas las ventajas de una Herramienta CASE WORKBENCH, la cual aporta con la generación de prototipos desde el principio de las etapas de desarrollo, permitiendo corregir errores desde el inicio del desarrollo de software.

20

Hit and source - focused comparison, Side by Side:

Left side: As student entered the text in the submitted document.
Right side: As the text appears in the source.
