Portada

Índice

Índice de tablas

Índices de figuras

Introducción

Objetivos

Justificación

Alcances

Limitaciones

# Capítulo I: Investigación Preliminar

## Marco Teórico

### Aplicación Web

(Luján Mora, 2002) Una aplicación web (web-based application) es un tipo especial de aplicación cliente/servidor, donde tanto el cliente (el navegador, explorador o visualizador) como el servidor (el servidor web) y el protocolo mediente el que se comunican (HTTP) están estandarizados y no han de ser creados por el programador de aplicaciones.

El protocolo HTTP forma parte de la familia de protocolos de comunicaciones TCP/IP, que son los empleados en Internet. Estos protocolos permiten la conexión de sistemas heterogéneos, lo que facilita el intercambio de información entre distintos ordenadores.

### Factibilidad

### Factibilidad técnica

### Factibilidad operativa

### Factibilidad económica

### Depreciación

### Valor Anual Neto (VAN)

### Tasa de Interés de Retorno (TIR)

### Herramienta de descripción del problema

### Árbol del problema

### Herramienta de definición del problema

### Pareto

### Entrevista

(Franklin Fincowsky, 2009) Esta herramienta consiste básicamente en reunirse con una persona con el fin de interrogarla en forma meticulosa para obtener información. Este medio es posiblemente el más usado y el más completo, pues el entrevistador, debido a que tiene un estrecho contacto con el entrevistado, además de obtener respuestas, puede percibir actitudes y recibir comentarios.

La entrevista debe dirigirse a directivos y empleados de una misma área o que intervienen en la misma clase de tareas, así como a clientes y/o usuarios, prestadores de servicios y proveedores que interactúan con la organización.

Para que una entrevista se desarrolle en forma positiva, es conveniente observar estos aspectos:

* Tener claro el objetivo. Se recomienda preparar previamente una guía de entrevista con los principales puntos que se desea captar, para que al término de la misma pueda verificarse si se ha obtenido la información prevista.
* Establecer anticipadamente la distribución del trabajo. Es conveniente asignar responsabilidades y determinar las áreas a investigar con el suficiente tiempo.
* Concertar previamente la cita. De esta forma el entrevistado estará debidamente preparado para proporcionar la información con el tiempo y tranquilidad necesarios para mantener la concentración en la entrevista, evitar interrupciones y posibles apreciaciones erróneas.
* Atender al compromiso. Es necesario acudir con anticipación a la hora convenida al área en que se llevará a cabo la entrevista programada. Para lograr la mayor eficacia en su desarrollo es conveniente observar los pasos siguientes:

1. Concentrarse de manera relajada.
2. Seguir un mapa mental para captar la información de manera lógica y consecuente.
3. Utilizar preguntas de terminación abierta en las cuales la respuesta correcta no sea tan obvia.
4. No hacer preguntas irrelevantes.
5. En la medida de lo posible escuchar, no hablar.
6. Asegurarse de no presionar al entrevistado.
7. Evitar el exceso de confianza en la memoria.
8. Dejar “abierta la puerta”

### Encuesta

### Unified Modeling Language (UML)

(Larman, 1999) El UML (Lenguaje Unificado para la Construcción de Modelos) se define como un “lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software…” [BJR97]. Es un sistema notacional (que, entre otras cosas, incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos.

El UML es un estándar incipiente de la industria para construir modelos orientados a objetos. Nació en 1993 por iniciativa de Grady Booch y Jim Rumbaugh para combinar sus dos famosos métodos: el de Booch y el OMT (Object Modeling Technique, Técnica de Modelado de Objetos). Más tarde se les unió Ivar Jacobson, creador del método OOSE (Object-Oriented Software Engineering, Ingeniería de Software Orientada a Objetos). En respuesta a una petición OMG (Object Management Group, asociación para fijar los estándares de la industria) para definir un lenguaje y una notación estándar del lenguaje de construcción de modelos, en 1997 propusieron el UML como candidato.

### Base de datos

(Silberschatz, Korth, & Sudarshan, 2002) Un sistema de bases de datos es una colección de archivos interrelacionados y un conjunto de programas que permitan a los usuarios acceder y modificar estos archivos.

Uno de los propósitos principales de un sistema de bases de datos es proporcionar a los usuarios una visión abstracta de los datos. Es decir, el sistema esconde ciertos detalles de cómo se almacenan y mantienen los datos.

### Pruebas

(Berzal, 2005) Errar es humano y la etapa de pruebas tiene como objetivo detectar los errores que se hayan podido cometer en las etapas anteriores del proyecto (y, eventualmente, corregirlos). Lo suyo, además, es hacerlo antes de que el usuario final del sistema los tenga que sufrir. De hecho, una prueba es un éxito cuando se detecta un error (y no al revés, como nos gustaría pensar).

La búsqueda de errores que se realiza en la etapa de pruebas puede adaptar distintas formas, en función del contexto y de la fase del proyecto en la que nos encontremos:

* Las pruebas de unidad sirven para comprobar el correcto funcionamiento de un componente concreto de nuestro sistema. Es este tipo de pruebas, el "probador" debe buscar situaciones límite que expongan las limitaciones de la implementación del componente, ya sea tratando éste como una caja negra ("pruebas de caja negra") o fijándonos en su estructura interna ("pruebas de caja blanca"). Resulta recomendable que, conforme vamos añadiéndole nueva funcionalidad a nuestras aplicaciones, vayamos creando nuevos test con los medir nuestro progreso y también repitamos los antiguos para comprobar que lo que antes funcionaba sigue funcionando (test de regresión).
* Las pruebas de integración son las que se realizan cuando vamos juntando los componentes que conforman nuestro sistema y sirven para detectar errores en sus interfaces.
* Una vez "finalizado" el sistema, se realizan pruebas alfa en el seno de la organización encargada del desarrollo del sistema. Estas pruebas, realizadas desde el punto de vista de un usuario final, pueden ayudar a pulir aspectos de la interfaz de usuario del sistema
* Cuando el sistema no es un producto a medida, sino que se venderá como un producto en el mercado, también se suelen realizar pruebas beta. Estas pruebas las hacen usuarios finales del sistema ajenos al equipo de desarrollo y pueden resultar vitales para que un producto tenga éxito en el mercado.
* En sistemas a medida, se suele realizar un test de aceptación que, si se supera con éxito, marcará oficialmente el final del proceso de desarrollo y el comienzo de la etapa de mantenimiento.

Por último, a lo largo de todo el ciclo de vida del software, se suelen hacer revisiones de todos los productos generados a lo largo del proyecto, desde el documento de especificación de requerimientos hasta el código de los distintos módulos de una aplicación. Estas revisiones, de carácter más o menos formal, ayuden a verificar la corrección del producto revisado y también a validarlo (comprobar que se ajusta a los requerimientos reales del sistema). Errar es humano y la etapa de pruebas tiene como objetivo detectar los errores que se hayan podido cometer en las etapas anteriores del proyecto (y, eventualmente, corregirlos). Lo suyo, además, es hacerlo antes de que el usuario final del sistema los tenga que sufrir. De hecho, una prueba es un éxito cuando se detecta un error (y no al revés, como nos gustaría pensar).

La búsqueda de errores que se realiza en la etapa de pruebas puede adaptar distintas formas, en función del contexto y de la fase del proyecto en la que nos encontremos:

* Las pruebas de unidad sirven para comprobar el correcto funcionamiento de un componente concreto de nuestro sistema. Es este tipo de pruebas, el "probador" debe buscar situaciones límite que expongan las limitaciones de la implementación del componente, ya sea tratando éste como una caja negra ("pruebas de caja negra") o fijándonos en su estructura interna ("pruebas de caja blanca"). Resulta recomendable que, conforme vamos añadiéndole nueva funcionalidad a nuestras aplicaciones, vayamos creando nuevos test con los medir nuestro progreso y también repitamos los antiguos para comprobar que lo que antes funcionaba sigue funcionando (test de regresión).
* Las pruebas de integración son las que se realizan cuando vamos juntando los componentes que conforman nuestro sistema y sirven para detectar errores en sus interfaces.
* Una vez "finalizado" el sistema, se realizan pruebas alfa en el seno de la organización encargada del desarrollo del sistema. Estas pruebas, realizadas desde el punto de vista de un usuario final, pueden ayudar a pulir aspectos de la interfaz de usuario del sistema
* Cuando el sistema no es un producto a medida, sino que se venderá como un producto en el mercado, también se suelen realizar pruebas beta. Estas pruebas las hacen usuarios finales del sistema ajenos al equipo de desarrollo y pueden resultar vitales para que un producto tenga éxito en el mercado.
* En sistemas a medida, se suele realizar un test de aceptación que, si se supera con éxito, marcará oficialmente el final del proceso de desarrollo y el comienzo de la etapa de mantenimiento.

Por último, a lo largo de todo el ciclo de vida del software, se suelen hacer revisiones de todos los productos generados a lo largo del proyecto, desde el documento de especificación de requerimientos hasta el código de los distintos módulos de una aplicación. Estas revisiones, de carácter más o menos formal, ayuden a verificar la corrección del producto revisado y también a validarlo (comprobar que se ajusta a los requerimientos reales del sistema).