EL MODELO DE IMPLANTACION DEL USUARIO

Este modelo contiene una descripción completa de lo que el sistema debe hacer para satisfacer al usuario. Específicamente, el modelo esencial describe:

- La política, o lógica, esencial de las funciones que se requiere realizar.
- El contenido esencial de los datos que almacena el sistema, y que se mueven a través de él.
- El comportamiento esencial dependiente del tiempo que el sistema debe exhibir para manejar señales e interrupciones del ambiente exterior.

En prácticamente todos los proyectos de desarrollo de sistemas, el usuario insistirá en proporcionar alguna información adicional. Esta información adicional involucrar cuestiones de implantación que son suficientemente importantes, es decir, tienen el suficiente impacto sobre la capacidad del usuario para usar el sistema.

Recientemente, varias opciones y posibilidades han aumentado la importancia de las cuestiones de implantación:

- ¿Qué partes del modelo esencial se asignaran a la PC (bajo el control del usuario) y que partes a la computadora principal? ¿Qué parte de los datos se asignaran a la PC y cuál a la principal? ¿Qué formato tendrán las entradas que el usuario proporciona a la PC? ¿Qué actividades adicionales de apoyo hay que proporcionar para asegurar que el usuario no dañe inadvertidamente los datos almacenados en la PC o en la computadora principal?
- Decidir que partes del sistema se implantaran utilizando lenguajes de cuarta generación y cuales usando lenguajes convencionales de tercera generación.
- El usuario y el analista podrían hacer un prototipo de porciones del sistema utilizando un lenguaje de cuarta generación para la exploración y experimentación con formatos de entrada, diálogos en línea y diálogos de salida para pantallas o reportes.

Selección y compra de software bajo licencia. En este caso las mismas cuestiones de implantación son de importancia para el usuario.

DETERMINACION DE LA FRONTERA DE AUTOMATIZACIÓN

La frontera de automatización es casi irrelevante en el modelo esencial pues, aunque el usuario quiere que se desarrolle un sistema automatizado, también necesita una declaración bien hecha de los requerimientos de las funciones y los datos que queden justo fuera de la frontera de automatización.

Hay tres casos extremos:

- Al usuario pudiera no importarle dónde esta la frontera de automatización.
- El usuario podría escoger un sistema totalmente automatizado.
- El usuario podría optar por un sistema completamente manual.

Normalmente estas opciones extremas no ocurren; basándose en interacciones entre el usuario, el analista y el equipo de implantación, se automatizara parte de las actividades del modelo esencial y otras se identificaran como funciones manuales.

No es labor ni del analista ni del equipo de implantación escoger la frontera de automatización, sino responsabilidad del usuario. Una vez elegida la frontera de automatización, el analista pudiera darse el lujo de pensar en eliminar procesos y datos manuales.

En caso general, el analista debe reconocer que las actividades manuales son parte del nuevo sistema.

DETERMINACIÓN DE LA INTERFAZ HUMANA

Involucra cuatro asuntos relacionados:

1. La elección de los dispositivos de entrada y salida.
2. El formato de todas las entradas que fluyen desde los terminadores hasta el sistema.
3. El formato de todas las salidas que fluyen desde el sistema hacia los terminadores.
4. La secuencia y los tiempos de entradas y salidas en un sistema en línea.

Dispositivos de entrada y salida. La elección puede estar determinada por los terminadores fuera del sistema; por ejemplo, si el sistema produce reportes de salida para el gobierno, tal vez no haya otra opción para producirlos en papel.

Los dispositivos que se usan típicamente para proporcionar entradas al sistema incluyen los siguientes: tarjetas perforadas, cinta, magnética, discos flexibles, terminales y computadoras personales, lectores ópticos y lectores de código de barras, teléfono, voz.

Los medios comúnmente usados para las salidas son los siguientes: salidas impresas, tarjetas perforadas, salida de voz, graficador, terminal, cinta magnética o disco y COM (Microforma para Salidas de Computadora).

Formatos de entrada y salida. En el caso de sistemas de información computarizados, las siguientes reglas ayudaran a desarrollar una interfaz amable con el usuario en la mayor parte de casos:

1. El sistema debe pedir entradas y producir salidas en forma consistente.
2. Pida información con una secuencia lógica.
3. Haga obvio al usuario el tipo de error que ha cometido y dónde.
4. Distinga entre edición de campos y ediciones de pantalla.
5. Haga la edición y la revisión de errores dependientes del usuario.
6. Permita que el usuario pueda (a) cancelar parte de la transacción y (b) cancelarla toda.
7. Proporcione un mecanismo de “ayuda” conveniente.
8. Distinga entre sistemas guiados por menús y sistemas dirigidos por órdenes; si es el apropiado, déle a escoger al usuario.
9. Si el sistema está realizando un proceso largo, despliegue un mensaje al usuario para que no crea que se detuvo.
10. Proporcione alternativas por omisión para las entradas estándar.
11. Aproveche el color y el sonido, pero no abuse.

Diseño de las formas. El diseño de los formatos de entrada y salida de un sistema tradicionalmente se conoce como formas. Aunque hay muchos estilos diferentes para las formas, todas deben contener la siguiente información básica:

- Título, para distinguirla de cualquier otra.
- Instrucciones, para decir al usuario cómo poner la información necesaria en la forma.
- Cuerpo, es decir, la parte principal de la forma, donde se ingresan los datos.

Dependiendo de la aplicación el analista puede diseñar formas individuales o de especialidad. Las primeras suelen imprimirse en hojas sencillas y son adecuadas para la

gran mayoría de las situaciones, Las formas de especialidad son más complejas y se crean con la ayuda de un diseñador de formas con experiencia.
Las formas de especialidad son mucho más caras que las sencillas; por lo que se debe tener cuidado de que no representen un costo importante en el sistema.
Códigos de entrada y salida. Como parte de la labor de especificar formatos de entrada y salida, el analista muchas veces debe especificar códigos, es decir, abreviaciones de la información que sería difícil y tardado describir con detalle. Ejemplos de estos son los números de seguro social , códigos postales, números de ISBN para libros publicados y números de identificación de empleados que se asignan a las compañías en sus declaraciones de impuestos.
Los ejemplos anteriores representan códigos externos para la mayoría de nosotros, es decir, sin importar el tipo de sistema tenemos que usar códigos desarrollados por gobierno, correos y el seguro social.
Las técnicas de codificación son un área especializada y deben ser:

- Expandible, debe proporcionar espacio para entradas adicionales que pudieran requerirse.
- Preciso, debe identificar el artículo específico.
- Conciso, debe ser breve pero describir adecuadamente al artículo.
- Conveniente, debe ser fácil de codificar y decodificar.
- Con significado, debe ser útil para quienes lo manejan, de ser posible debe indicar algunas de las características del artículo.
- Operable, debe ser compatible con los métodos presentes y anticipados de proceso de datos, manual o a maquina.

Es más común el código de clasificación, que usa grupos de dígitos (o letras) dentro del código para identificar clasificaciones mayores, intermedias o menores dentro del artículo que se describe.
Los códigos alfabéticos también se usan comúnmente en sistemas de información. Muchos códigos alfabéticos con intentos mnemónicos o auxilios de memoria que el usuario podrá recordar fácilmente.
Algunos códigos se autoverifican; es decir, contienen información adicional (redundante) que puede usarse par verificar que se haya ingresado correctamente.

IDENTIFICACIÓN DE LAS ACTIVIDADES DE APOYO MANUAL ADICIONAL
Tenemos que ocuparnos por la posibilidad de la tecnología defectuosa en cuatro áreas principales:

- Ingreso de datos al sistema.
- Realización de los cálculos.
- Almacenamiento a largo plazo de los datos.
- Salida de datos del sistema.

¿Qué debe hacerse al respecto de estas áreas posibles de tecnología defectuosa? , depende en gran medida de 1) el nivel estimado de confiabilidad del hardware y el software que se usa; 2) la naturaleza de la aplicación del usuario y, 3) los costos y cargos asociados con entradas y salidas defectuosas.

- Entradas o salidas redundantes.
- Tecnología de procesador redundante.
- Redundancia interna.
- Controles por lote (batch)

- Verificaciones secuenciales.

ESPECIFICACION DE RESTRICCIONES OPERACIONALES

Finalmente el equipo de implantación tendrá que decidir la combinación de hardware, sistema operativo, equipo de telecomunicaciones, lenguaje de programación y estrategia de diseño para implantar mejor los requerimientos. Para que todo esto se pueda lograr bebe realizar una declaración e restricciones operativas, en el que las cuestiones son:

- Volumen de los datos.
- Tiempo de respuesta de las diversas entradas.
- Restricciones políticas sobre las modalidades de implantación.
- Restricciones ambientales.
- Restricciones de seguridad y confiabilidad.
- Restricciones de seguridad.

CONCLUSION PERSONAL

Mas que concluir con la construccion de un modelo de implantacion del usuario por analistas, usuarios, diseñadores y por suspuesto programadores es de gran importancia cumplir con los resultados esperados por todos desde la fase de analisis de proyecto, la aprobación del diseño y la implementación se denben de incluir puntos tan necesarios como la identificación del problema, soluciones alternativas, vista global del odelo, costos, beneficios, la factibilidad y demas estimaciones que estan incluidas dentro del proyecto.

PASANDO AL DISEÑO

Una vez completado el modelo de implantación del usuario concluye oficialmente la labor de análisis de sistemas. Por ello es importante entender el proceso que enfrenta el diseñador cuando el analista termina su labor.

LAS ETAPAS DEL DISEÑO

La actividad de diseño involucra el desarrollo de una serie de modelos. Los modelos más importantes para el diseñador son el modelo de implantación de sistemas y el modelo de implantación de programas. El modelo de implantación de sistemas se divide en un modelo del procesador y uno de tareas.

El modelo del procesador. El diseñador del sistema trata principalmente de decidir cómo se asignara el modelo esencial a los distintos procesadores (CPU) y como deben comunicarse entre sí. Existe típicamente una variedad de opciones:

- El modelo esencial completo se le puede asignar a un solo procesador (solución de computadora principal).
- Cada burbuja de la figura del DFD del modelo esencial se puede asignar a un procesador distinto (solución distribuida).
- Se puede escoger una combinación de computadoras principales, minis y micros para minimizar costos, maximizar confiabilidad o lograr algún otro objetivo.

Así como se deben asignar procesos a los componentes de hardware , los almacenes de datos se deben igualmente asignar. Dado que la mayor parte de los almacenes se

comparten entre muchos procesos, también sebe decidir si se asignan copias del almacén del a diferentes procesadores.
Algunas opciones disponibles al diseñador del sistema para comunicación de procesador a procesador son:

- Conexión directa entre procesadores.
- Enlace de telecomunicaciones entre procesadores.
- Enlace directo entre procesadores.

El diseñador debe tomar en cuanta varios factores al hacer estas asignaciones, las cuestiones principales son:

- Costo, dependiendo de la naturaleza del sistema, pudiera ser o no ser más barata un implantación de un solo procesador.
- Eficiencia, el diseñador de sistemas generalmente se preocupara por el tiempo de respuesta de los sistemas en línea y por la longitud del ciclo para los sistemas de cómputo por lote,
- Seguridad, el usuario final podría tener requerimientos de seguridad que dicten que algunos (o todos) los procesadores y/o datos delicados se coloquen en lugares protegidos.
- Confiabilidad, el usuario final típicamente especifica los requerimientos de confiabilidad para un nuevo sistema.
- Restricciones políticas y operacionales. La configuración de hardware puede verse influenciada también por restricciones políticas impuestas directamente por el usuario final, por otros niveles de administración dentro de la organización o por el departamento de operaciones a cargo del mantenimiento y operación de todos los sistemas de cómputo.

El modelo de tareas. Una vez que se han asignado procesos y almacenes a los procesadores, el diseñador debe, procesador a procesador, asignar procesos y almacenes al alas tareas individuales de cada uno. El diseñador generalmente trata de mantener los procesos con mayor volumen de comunicación dentro de la misma tarea. El diseñador puede considerar cada tarea como una actividad independiente no sincronizada.
El modelo de implantación de programas. Finalmente llegamos al nivel de una tarea individual; hasta aquí el diseñador ya logro completar dos niveles de asignación de procesos de almacenamiento de datos. Dentro de una tarea individual, la computadora opera de una manera no sincronizada; sólo se puede llevar a acabo una actividad a la vez. El modelo más común de organización de la actividad es una sola unidad sincronizada es el diagrama de estructura, que muestra la organización jerárquica de módulos dentro de una tarea.

METAS Y OBJETIVOS DEL DISEÑO

Además De lograr los objetivos que se especifican en el modelo de implantación del usuario, el diseñador también se ocupa de la calidad global del diseño.
El campo del diseño estructurado ofrece guías para ayuda al diseñador a determinar los módulos y sus interconexiones, que mejor realizarán los requerimientos especificados por el analista. Las dos reglas más importantes son las referentes al acoplamiento y la cohesión; a continuación se discuten éstas y algunas otras reglas comunes.

- Cohesión, grado en el cual los componentes de un módulo son necesarios y suficientes para llevar a cabo una sola función bien definida.
- Acoplamiento, grado en el cual los módulos se interconectan o se relacionan entre ellos. Entre más fuerte sea el acoplamiento entre módulos de un sistema, más difícil es implantarlo y mantenerlo, pues entonces se necesitará un estudio cuidadoso para la modificación o cambio y modificación de algún módulo o módulos.
- Tamaño del módulo, de ser posible, cada módulo debe ser lo suficientemente pequeño como para caber en una sola página (o para que pueda desplegarse en una sola pantalla).
- Alcance del control, el número de subordinados inmediatos que un módulo administrador puede llamar se conoce como el alcance del control.
- Alcance del efecto/alcance del control, esta regla sugiere que cualquier módulo afectado por el resultado de alguna decisión debe ser subordinado del módulo que toma la decisión.

CONCLUSION PERSONAL

Primero que nada conviene cumplir o solucionar de la mejor manera posible los requerimientos del usuario con la configuracion de procesadores y posteriormente dentro de cada procesador; de aqui que el diseñador decidirá la asignacion de procesos y datos a las tareas que se hayan identificado. Ya por ultimo, este mismo debe de organizar los procesos dentro de cada tarea en modulos usando el diagrama de estructura como herramienta con la cual sera de grana yuda para obtener una solucion que quizas no se la mejor pero si estara mas cerca de ella.

PROGRAMACIÓN Y PRUEBA

La programación y prueba comienzan cuando termina la actividad de diseño: La fase de programación o implantación de un proyecto típico involucra la escritura de instrucciones en un lenguaje de programación para implantar lo que el analista y el diseñador ha organizado en módulos. La prueba, involucra ejercitar el sistema para asegurar que produzca las salidas apropiadas y exhiba el comportamiento adecuado para una gama amplia de entradas.

EL PAPEL DEL ANALISTA EN LA PROGRAMACIÓN Y LA PRUEBA

Hay algunas razones por las cuales el analista puede requerir seguir involucrado en el proyecto al comenzar la actividad de programación:

- El analista es el líder del proyecto y esta a cargo de los programadores.
- El titulo del analista es programador/analista o analista programador.
- El analista forma parte de un grupo que escribe casos de prueba que se usarán para ejercitar los programas que los programadores escriben.
- El analista puede verse involucrado en el desarrollo de manuales de usuario, preparación de los usuarios o en la planeación de la instalación del nuevo sistema y conversión de datos desde el otro sistema.
- Tal vez los programadores no comprenden las especificaciones del analista de sistemas.

- Puede ser que los usuarios hayan comenzado a cambiar de opinión con respecto a los requerimientos, incluso cuando los programadores están implantando los que decían querer.

EL IMPACTO DEL ANALISIS, LA PROGRAMACIÓN Y LA PRUEBA SOBRE LA ESTRUCTURA ORGANIZACIONAL

Quienes tienen el título de analistas usualmente son personas relativamente maduros, con varios años de experiencia, por lo tanto, la labor de los programadores consiste básicamente en traducir las especificaciones del analista de sistemas a un lenguaje de programación.

Se puede permitir a la gente madura hacer las actividades de nivel superior del proyecto y a los novatos jóvenes todas las actividades detalladas de nivel inferior.

La ventaja para los programadores es que les toca hacer el trabajo creativo de escribir las especificaciones de proceso y tienen el placer de traducir sus propias especificaciones a código. También se tiene la ventaja de mantener a la gente madura en contacto con la tecnología, al forzarlos a continuar realizando alguna labor de diseño y programación.

IMPLANTACIÓN DESCENDENTE Y SEGUIMIENTO RAPIDO

¿Por qué consideraría un administrador del proyecto seguir el enfoque radical? ¿Por qué podría alguien comenzar la albor de diseño y programación antes de concluir el análisis? Existen muchas razones, de las cuales las más importantes son las siguientes:

- Como la labor de análisis, diseño y programación se realiza de manera concurrente, usualmente se tiene la oportunidad de acortar dramáticamente el tiempo total necesario para un proyecto.
- La labor de desarrollo concurrente puede usarse como una forma de hacer el prototipo: permite al equipo del proyecto mostrar al usuario una versión esquelética del sistema antes del concluir la labor detallada de análisis.
- Comenzar la labor de programación pronto, suele evitar diversos problemas referentes a la demanda de recursos, tales como tiempo de computo, que de otra manera se convertirían en un obstáculo.

PROGRAMACION Y LENGUAJES DE PROGRAMACION

Si aún esta involucrado en el proyecto durante la etapa de implantación, debe tener en cuéntalas técnicas de programación:

- Cuatro generaciones de lenguajes de programación.
- Asuntos importantes en la programación.
- Cosas que debe buscar si examina la codificación de los programadores.

Las cuatro generaciones de lenguajes de programación. Es conveniente agrupar los distintos lenguajes de programación en cuatro generaciones distintas:

- Lenguajes de primera generación, fueron los lenguajes de maquina que se usaron en los años 50; los programadores que intentaban que la computadora hiciera algo útil codificaban sus instrucciones con unos y ceros binarios.
- Lenguajes de segunda generación, son los sucesores del lenguaje maquina; generalmente se conocen como lenguajes de ensamble o ensambladores, estos lenguajes son de bajo nivel en el sentido de que el programador tiene que

escribir una declaración por cada instrucción de maquina. La principal desventaja de estos lenguajes es que en lugar de pensar en términos del problema que requiere resolver, el programador debe pensar en términos de la máquina.

- Lenguajes de tercera generación, son la norma actual, incluyen BASIC, FORTRAN, Pascal, C, Ada y muchos más. Son de alto nivel en el sentido de que una sola declaración usualmente representa cinco o diez declaraciones de lenguaje ensamblador, además permiten al programador expresar pensamientos en una forma un tanto más compatible con el área del problema que el que se está trabajando. Los lenguajes de tercera generación también se caracterizan como lenguajes guiados por procedimientos. Requieren que el programador piense con cuidado la secuencia de los cálculos o procedimientos necesarios para lograr una acción.
- Lenguajes de cuarta generación, o 4GLs son la moda actual y muchos consultores de computación los consideran el desarrollo más importante en el campo del software en los últimos 20 años. Algunos ejemplos son: FOCUS, IDEAL, MARK IV, MANTIS, MAPPER, Rbase-500. La mayor parte tienen las características de programación estructurada.

Cuestiones importantes de programación. Como analista debe estar familiarizado con cuestiones comunes que los programadores enfrentan:

- Productividad, escribir más software, más rápidamente.
- Eficiencia.
- Corrección, si el programa no funciona correctamente, no importa que tan eficiente sea.
- Portabilidad, el usuario puede desear ejecutar el mismo sistema en varios tipos distintos de computadoras.
- Mantenibilidad, los sistemas viven durante mucho tiempo, por lo que el software debe mantenerse.

Cosas de las que hay que tener cuidado. Como analista, puede tener la oportunidad de observar el trabajo que realizan los programadores del proyecto. ¿Pero cómo se logran las metas?, las siguientes son cuestiones clave en la programación:

- La programación estructurada, debe seguirse un enfoque de programación estructurada en el que la lógica del programa se organiza en combinaciones anidadas de construcciones SI-ENTONCES y HACER-MIENTRAS.
- Módulos pequeños, es esencial que los programas se organicen en pequeños módulos para que la lógica de programación quepa en una sola página de listado de programa.
- Sencillez de estilo, escritura de programas sencillo, es decir, programas que el programador promedio puede entender y que se le pueden pasar al programador de mantenimiento.

PRUEBAS

Es probable que el proceso de probar el sistema tome tanto como la mitad del tiempo programado para su desarrollo, dependiendo de qué tan cuidadosamente se hayan hecho las actividades iníciales de análisis, diseño y programación. Si se realizo un trabajo impecable se debe hacer algún esfuerzo para verificar que no haya errores, si se hizo un trabajo imperfecto, entonces la prueba se vuelve iterativa; la primera tanda

de pruebas muestra la presencia de errores, y las posteriores verifican si los programas corregidos funcionan correctamente.

Tipos de pruebas. Hay distintas estrategias de prueba; las más comunes se conocen como prueba ascendente y descendente.

El enfoque ascendente empieza por probar módulos individuales pequeños separadamente, luego los módulos individuales se combinan para formar unidades cada vez más grandes que se probarán en masa, esto se conoce como prueba de subsistemas. Finalmente todos los componentes del sistema se combinan para probarse, esto se conoce como prueba del sistema y suele estar seguido de las pruebas de aceptación donde se permite al usuario usar sus propios casos de prueba para verificar que el sistema esté trabajando de manera correcta.

El enfoque de prueba descendente empieza con un esqueleto del sistema; es decir, la estrategia de prueba supone que se han desarrollado los módulos ejecutivos de alto nivel del sistema, pero que los de bajo nivel existen sólo como módulos vacios.

Además de estos conceptos básicos, debería de estar familiarizado con los siguientes tipos de prueba:

- Prueba funcional, su propósito es asegurar que el sistema realiza sus funciones normales de manera correcta.
- Prueba de recuperación, el propósito de este tipo de prueba es asegurar que el sistema pueda recuperarse adecuadamente de diversos tipos de fallas.
- Prueba de desempeño, el propósito de este tipo de prueba es asegurar que el sistema pueda manejar el volumen de los datos y transacciones de entrada especificados en el modelo de implantación del usuario, además de asegurar que tenga el tiempo de respuesta requerido.

Debemos estar preparados para un volumen grande de pruebas, para realizar esto de manera efectiva, el equipo que desarrolla el sistema necesita tres cosas: planes de prueba, descripciones de pruebas y procedimientos de prueba.

Un plan de prueba es exactamente lo que parece: un documento organizado que describe las actividades de prueba.

Un documento de planeación de pruebas típico debe contener la siguiente información:

- Propósito de la prueba.
- Localización y horario de prueba.
- Descripciones de la prueba.
- Procedimientos de prueba.

Conceptos relacionados. La mayor parte de las organizaciones llevan a cabo pruebas, pueden usar para aumentar el proceso estándar los siguientes conceptos:

- Recorridos y revisiones, son una forma de supervisión hecha pro un grupo revisor de productos técnicos, revisan diagramas de flujo de datos, diagramas de estructura además de programas, su objetivo es descubrir posibles errores en el sistema.
- Inspecciones, son similares a los recorridos, tienen un plan más formal de puntos a examinar en el programa antes de que se pueda probar.
- Pruebas de corrección.

¿QUE OCURRE DESPUES DE LAS PRUEBAS?

Alguien debe llevar a cabo las actividades finales en un proyecto de desarrollo de sistemas:

1. Conversión
2. Instalación.
3. Capacitación.

La conversión es la tarea de traducir los archivos, formas, bases de datos actuales del usuario al formato que el nuevo sistema requiere.

La instalación del nuevo sistema puede ser un asunto instantáneo, pero a menudo puede ser necesario instalar el sistema por etapas.

La capacitación es la tarea final del equipo de desarrollo del sistema: la capacitación de los usuarios, demás de la preparación personal de operaciones, los programadores de mantenimiento y varios niveles de administración.

CONCLUSIONES

Aun despues de haber analizado e identificado los problemas que queremos solucionar o mejorar con determinado sistema, posteriormente planteado un modelo que lo resuelve es de gran importancia considerar la forma, plataforma, tiempo, costo, entre otros puntos para poder llevar a cabo la debida, si no es que excelente programación durante el desarrollo del proyecto, ademas de considerar la implantación del sistema y capacitación a los usuarios comenzar a obtener los resultados en los diferentes tipos de pruebas ya que serán los que determinarán la calidad, funcionalidad, originalidad, seguridad y rendimiento de dicho sistema.