Una vez leído el material:
1- Describa que entiende por proyecto y ciclo de vida.
2- Mencione los puntos de fusión de Albrecht y explique cada uno con sus palabras.

Fecha de entrega: 14/09/20

to, que se ve así calificado y comparado a lo ya existente y a las otras dos soluciones.

Los responsables, en función de criterios establecidos y de las posibles evoluciones, determinarán el argumento que deberá ser objeto de un estudio más profundo y que será la base del futuro desarrollo del producto.

2. ESTUDIO FUNCIONAL DEL SISTEMA DE INFORMACION Y CREACION DE PROTOTIPOS

Es a partir del argumento escogido en la fase precedente que va a orientarse el estudio del sistema de información (ver la diferencia con el sistema informático en el capítulo 1). El estudio de conveniencia/viabilidad no ha llevado al conjunto de especificaciones detalladas del sistema, por ello este estudio debe describir de forma precisa la solución deducida.

Es también la última etapa antes de la informática en el sentido técnico del término (hardware). Por esta razón pensamos que el maquetaje (sucesión de pantallas sin efectos sobre los datos) y el prototipaje (maquetaje al cual son añadidos efectos sobre algunos datos fundamentales) desempeñan aquí un papel esencial. En efecto, estas técnicas, ineludibles desde ahora, permiten al usuario validar a la vez el encadenamiento de diversas tareas previstas en el futuro programa y la ergonomía del diálogo hombre/máquina.

Sabemos que este último punto se ha convertido en algo primordial desde la aparición de herramientas tales como Windows o X-Windows, en la medida en que la gestión de interfaces de usuario exige ahora más desarrollos que el cuerpo de la aplicación propiamente dicho.

Podemos afirmar rotundamente, para que ciertos managers puedan aún apuntarse a la modernidad, que las técnicas de maquetaje y de prototipaje tendrán una cabida cada vez mayor en el marco de los métodos e instrumentos de la empresa.

Esta concisa descripción será la trama del pliego de condiciones del usuario, que es el eje central del proyecto. En caso de litigio, y sabemos

por experiencia que suele haber muchos y por múltiples causas, es a él a quien nos remitiremos.

Por lo que deberá tener en cuenta que nada está implícito en un documento de este tipo. Encontrará en él un verdadero contrato en el que usted, como realizador, se comprometerá a suministrar lo que le ha garantizado a su cliente, el usuario.

Tendrá entonces que realizar el pliego de condiciones, todo el pliego de condiciones y nada más que el pliego de condiciones.

Una vez este contrato ha sido aceptado por las dos partes, se pasará a otra fase también muy interesante: el estudio técnico.

3. DESARROLLO

La responsabilidades cambian de dueño, ahora el primer piloto es el informático y el segundo el usuario. Lo que no es de extrañar en vista de los acuerdos firmados en el pliego de condiciones.

Esta fase se va a efectuar en dos etapas: el estudio técnico del sistema informático por una parte, y por otra la programación; ambas serán traducciones sucesivas del lenguaje empleado en el pliego de condiciones en busca de un lenguaje comprensible para el ordenador que efectuará los tratamientos solicitados.

3.1. Concepción técnica del sistema informático

Se trata de retomar el pliego de condiciones del usuario para deducir un dossier de especificaciones técnicas, designadas como internas porque reflejan la arquitectura interna del programa.

Como para todos los documentos importantes de esta fase, se acompaña un plan tipo al final del capítulo.

Una vez facilitados estos documentos, no queda más que llevar a cabo la solución técnica.

3.2. Programación

Aunque muchos la consideran trivial, esta fase es muy importante, sobre todo si su programa debe tener interacciones con otras aplicaciones, en el mismo lugar o en lugares distintos.

Los que auguran desde hace veinte años la muerte de los programadores y la llegada de los generadores de programas se equivocan y aciertan a la vez. Aciertan porque es verdad que los servicios informáticos tienen cada vez menos necesidad de programadores que produzcan líneas de Cobol con las que ya no se sabe que hacer: para esto pueden utilizarse algunos generadores. Y se equivocan porque todo ha cambiado con la llegada de los micros que lleva a la búsqueda sistemática y a menudo contradictoria, de la optimización y sofisticación.

De igual modo, esta fase de programación puede revelarse como primordial si su aplicación evoluciona en un contexto técnico complicado con, por ejemplo, relaciones multi-puestos con materiales y redes heterogéneas.

Por lo general son necesarias dos etapas en esta fase: la primera es la codificación, es decir la transcripción de las instrucciones requeridas en el informe de las especificaciones internas a un lenguaje de programación adaptado; la segunda consta de una serie de tests destinados a demostrar que los programas escritos funcionan correctamente en los planos técnico, lógico y funcional.

Dichos tests se dividen generalmente en dos tipos: los unitarios y los de integración.

Los primeros deben mostrar que cada programa está conforme técnicamente con lo que se espera de él independientemente de los otros.

Los segundos permiten afirmar que todos los programas de aplicación ya testados funcionan juntos y se intercambian regularmente, y sin problemas, los datos esperados (parámetros, código respuesta o datos transformados).

4. FASE DE ENTRADA

Realizados estos tests, unitarios y globales, se pasará a la fase de entrada, en el curso de la cual se somete a la aprobación de los compradores el programa recién realizado.

Esta entrada se efectúa sobre una plataforma de integración que pone en juego un conjunto de materiales susceptibles de funcionar para el último usuario.

Es entonces cuando se pasan los programas que han sido testados anteriormente en el conjunto de las posibles configuraciones y se anotan los problemas para resolverlos rápidamente.

Esta fase llega a ser primordial por la multiplicación de los materiales de tipo micro y de los conjuntos tan heterogéneos que pueden derivarse (relacionar una carta con una red determinada administrada por el sistema de gestión de red, regulación de diferentes redes periféricas...)

Sólo se pondrá en marcha la versión del programa cuando los clientes den una opinión positiva al respecto.

5. PUESTA EN MARCHA

El producto es adoptado por el o los que lo han dirigido y funciona en el estudio. Ahora se trata de instalárselo al cliente para que funcione en un contexto real, después de generalizar su implantación en todos los lugares potenciales y antes de considerar que la línea de producto está en fase de servicio post-venta o "mantenimiento" en argot informático.

Esta instalación puede hacerse en dos tiempos: un primer lote de programas se analiza "en un contexto real", en lugares significativos y durante un período cuya duración depende de lo completas que estén las pruebas de integración, y tras el acuerdo de los usuarios y las observaciones de los compradores, que no tienen por qué ser forzosamente las mismas personas, se puede proceder a la generalización.

Esta etapa será más o menos larga según el número de puestos de trabajo a equipar, sin embargo la experiencia demuestra que dicha etapa

debe ser minuciosamente preparada ya que de su eficacia depende en parte la aceptación para la "base" de los aplicativos distribuídos.

En efecto, si bien las etapas precedentes han sido realizadas con personal motivado por la novedad o escogido, y por tanto valorado, para la puesta en marcha de la aplicación piloto; en esta fase se debe afrontar a menudo cierta resistencia a los cambios.

Sin embargo, si la comunicación ha sido bien realizada en torno al proyecto (ver capítulo 1 de la 3.ª parte), las cosas se sucederán sin demasiados problemas.

Cuando todos los puestos de trabajo han sido equipados con el nuevo programa, se llega a la fase de explotación por los usuarios y de mantenimiento por parte de los informáticos.

6. MANTENIMIENTO

En la introducción de las tareas de la puesta en marcha/explotación, recordamos el término de "servicio post-venta", que es en efecto lo mismo que se desea conseguir con un producto "clásico".

El programa instalado funcionará, pero como todo producto tendrá fallos que entrañarán acciones de tipo intervención *in situ*, cambio standard o reparación en el mismo lugar o en el centro informático. Es lo denominado por los informáticos "mantenimiento reparador" .

A veces las normas pueden evolucionar, las reglas de gestión o de organización pueden cambiar, lo que provocará modificaciones del programa: el" mantenimiento evolutivo".

Estas acciones deberán administrarse y no intervenir "siguiendo la corriente" (dejando aparte las reparaciones de problemas bloqueados), se privilegiará pues a los "trenes de mantenimiento" con el fin de no estar en perpetua fase de generalización de un fragmento de aplicación.

Después, como en todo producto, las necesidades de los usuarios evolucionarán de tal manera que, al final de un período más o menos largo, el

programa tendrá necesidad de ser reescrito. Es entonces cuando se entra en un nuevo ciclo de vida; lo que nos remite al principio de estos desarrollos.

En resumen, el ciclo de vida de un proyecto puede esquematizarse como se indica en la figura 3 con sus principales dossiers de final de etapa.

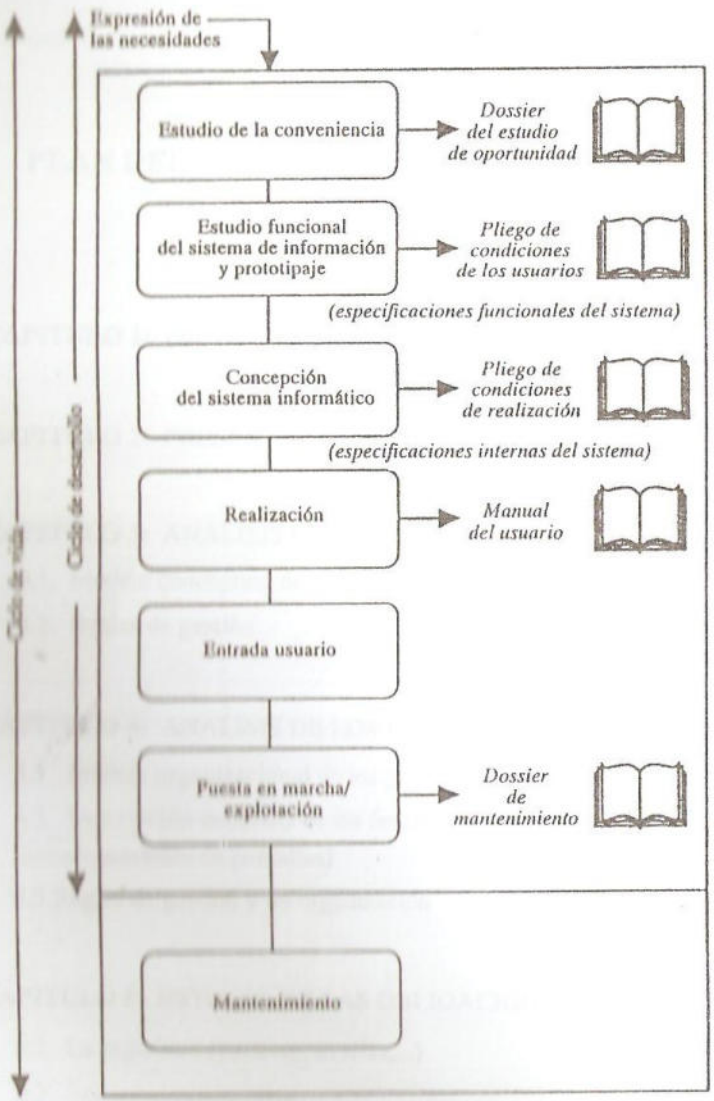

fig. 3 – El ciclo de vida

PLAN DEL DOSSIER

CAPITULO 1: OBJETIVO

CAPITULO 2: CONTEXTO, ENFOQUE DEL ESTUDIO

CAPITULO 3: ESTUDIO DE LO EXISTENTE

CAPITULO 4: ESCENARIOS

4.1. Descripción

4.2. Análisis Costes/Beneficios

- inversión
- rentabilidad
- plan de costes

ANEXOS

fig. 4 – Dossier del estudio de orientación y viabilidad

CAPITULO 5

ESTIMACION DE LOS COSTES

Existen numerosas técnicas de estimación de los costes, nuestro objetivo, en esta obra, no es hacer una descripción de todos, pero si de los más utilizados. Por esta razón presentaremos sucesivamente las técnicas de puntos de función de Albrecht y de Cocomo de Boehm en un orden que consideraremos como lógico. Al final de esta presentación especificaremos los pasos a seguir en lo concerniente a la estimación.

1. METODO DE LOS PUNTOS DE FUNCION

De todos los métodos de estimación, es probablemente el de los puntos de función de ALBRECHT (IBM) el de mayor aceptación. En efecto, la originalidad de este método proviene de que permite estimar los costes en función de la complejidad de los aspectos externos, o funciones, del futuro programa.

Esto debería permitir efectuar comparaciones entre proyectos antes de la utilización de métodos, instrumentos, o lenguajes diferentes en ordenadores distintos y entre equipos de proyectos también distintos.

Para ALBRECHT, existen cinco tipos de funciones:

- **Las entradas**: representadas por unas pantallas de entrada.
- **Las salidas**: pantallas o estados informáticos.
- **Los ficheros lógicos internos a la aplicación**: son el número de entidades, informes como el MERISE, o segmentos lógicos de datos próximos a la aplicación.
- **Los ficheros interfaces externos** compartidos con otras aplicaciones y que no están puestos al día por la aplicación.
- **Las consultas**: par de "entrada/salida" no puesto al día.

Con el fin de ayudarle en la utilización de los puntos de función, damos a continuación más detalles sobre el significado de cinco funciones para evitar cualquier mala interpretación del método.

1. Las entradas

Están catalogadas en esta categoría todas las potencialidades de comunicación de los usuarios con el programa. Encontraremos pues, entre otras, las pantallas, los mensajes provenientes de otros programas, las memorias de entrada y los soportes para lectura óptica.

Tres niveles de complejidad permiten caracterizarlos, los cuales denominaremos:

- SIMPLES cuando sólo comporten pocos tipos de datos o de grupos de datos lógicos o pocos factores humanos;
- COMPLEJOS cuando comporten numerosos tipos de datos o grupos de datos, factores humanos importantes y mandos complejos del cursor para visualización;
- MEDIOS cuando estén situados entre los dos precedentes.

Las entradas a considerar son o bien de formato distinto, o del mismo formato pero generando un proceso lógico diferente : un dato de entrada puesto al día se tiene que contar tres veces porque este proceso global necesitará tres procesos lógicos (uno para la creación, uno para la modificación y uno para la supresión).

2. Las salidas

Serán consideradas como salidas todas las posibilidades de comunicación del programa con el usuario. Encontraremos, entre otros, todos los estados de impresión, las pantallas y los mensajes hacia otras aplicaciones.

Se clasificarán las pantallas en función de criterios ya vistos para las entradas. En cuanto a las impresiones éstas podrán ser:

- SENCILLAS, si constan de pocas columnas y no necesitan más que una simple transformación de los datos;
- MEDIAS, si están constituidas de numerosas columnas con subtotales y generan transformaciones simples de datos múltiples;
- COMPLEJAS, si las transformaciones de datos múltiples implican interacciones complejas entre ellas cuando hacen referencia a numerosas bases de datos.

Las salidas a tener en cuenta son o bien de formato diferente o del mismo formato pero provenientes de un tratamiento lógico diferente (ver el ejemplo de la puesta al corriente en las entradas).

Las salidas de pantallas incluyen los mensajes de error, salvo si éstos no causan acción del usuario (acuse de recibo o error no bloqueante).

3. Los ficheros lógicos internos a la aplicación

En esta categoría encontraremos los principales ficheros lógicos desde el punto de vista del usuario, generados y mantenidos por el programa (pedidos, clientes, productos...). Estos son:

- SIMPLES si están sólo formados por un único tipo de registro y no son primordiales en materia de prestaciones o de mantenimiento;
- MEDIOS si se encuentran entre simples y complejos:
- COMPLEJOS si incluyen un gran número de tipo de registros, si tienen un impacto importante en materia de prestaciones o de copia de seguridad; toda particularidad importante del tipo de información distribuida se clasifica automáticamente en esta categoría.

Aquí sólo deben contarse los ficheros lógicos y no los físicos.

4. Los ficheros interface

Los ficheros que sirven de interface entre dos aplicaciones deben ser contabilizados en cada uno de los programas considerados.

5. Las consultas

Deben clasificarse aquí todas las entradas que provoquen una salida inmediata sin ocasionar la puesta al día de los datos. No obstante hay que asegurarse que éstas no hayan sido ya contabilizadas en las entradas o salidas.

Si un resultado de consulta se utiliza posteriormente para un proceso de puesta al día, habrá que calcular este dato en las consultas y en las entradas.

Para facilitarnos la búsqueda del nivel de complejidad, ALBRECHT nos proporciona unas tablas completas que presentamos a continuación:

Número de ficheros lógicos necesarios en línea	Número de campos en entrada		
	1 a 4	5 a 15	16 y más
0 o 1	Simple	Simple	Medio
2 o 3	Simple	Medio	Complejo
4 y más	Medio	Complejo	Complejo

fig. 12 – Complejidad de las entradas

Número de ficheros lógicos necesarios en línea	Número de campos en salida (o de mensaje)		
	1 a 5	6 a 19	20 y más
0 o 1	Simple	Simple	Medio
2 o 3	Simple	Medio	Complejo
4 y más	Medio	Complejo	Complejo

fig. 13 – Complejidad de las salidas

Número de tipos de registros lógicos (segmentos)	Número de campos		
	1 a 19	20 a 50	51 y más
1	Simple	Simple	Medio
2 a 5	Simple	Medio	Complejo
6 y más	Medio	Complejo	Complejo

fig. 14 – Complejidad de los ficheros