

Capítulo 2

Identificación de fases, tareas y entregables en un proyecto informático

2.1. INTRODUCCIÓN	2
2.2. DESCOMPOSICIÓN EN ACTIVIDADES DEL PROYECTO (WBS)	3
2.3. ENTREGABLES DE UN PROYECTO INFORMÁTICO	5
2.3.1. Entregables más usuales de un proyecto	6
2.4. DESCOMPOSICIÓN EN FASES DEL DESARROLLO DE UNA APLICACIÓN	8
2.5. DESCOMPOSICIÓN DEL DESARROLLO DE LA APLICACIÓN EN TAREAS	10
2.5.1. Enfoque de equipo en la identificación de tareas por actividades	10
2.5.2. Formas usuales de identificar tareas asociadas a un entregable	10
2.5.3. Tareas usuales de un proyecto informático	11
2.6. ALGUNAS REFLEXIONES SOBRE LA DESCOMPOSICIÓN DE UN PROYECTO EN TAREAS	19

2.1. Introducción

En este capítulo vamos a centrar nuestra atención en la descomposición del esfuerzo asignado a un proyecto. H. S. Geneen dijo: “Para leer un libro, se va del principio al fin. Para dirigir una empresa, se va exactamente al revés. Se empieza por los fines y luego se hace lo necesario para conseguirlos”. En nuestro caso podemos pensar de forma parecida, para realizar un proyecto, empezaremos por ver cuáles son los objetivos que queremos alcanzar y luego pensaremos qué cosas tenemos que hacer para alcanzar estos fines.

Esta descomposición pasará por identificar las fases de nuestro proyecto y el esfuerzo a aplicar en cada una de ellas. A su vez estas fases se descompondrán en tareas. También tendremos que marcar unos puntos (hitos) de control que nos permitan saber si el proyecto va de acuerdo a lo previsto.

Normalmente todas las fases y muchas tareas terminan en la generación de uno o varios documentos. A éstos se les llama entregables. Este nombre se debe a que pasan de manos del desarrollador a manos del controlador del proyecto o del cliente. En los proyectos informáticos se suele asociar los hitos a la consecución de un entregable.

De forma genérica ya habíamos identificado dos métodos complementarios que se pueden aplicar en la descomposición de proyectos:

- la descomposición del proceso (Análisis, Diseño, Codificación, ...), y
- la descomposición del producto (Contabilidad, Nómina, ...).

Lo normal en este punto del proyecto es que éste ya se haya enfocado hacia un único producto, es decir, los gestores de la empresa habrán identificado diferentes productos a desarrollar y solicitado el desarrollo de uno de ellos. Así será extraño que un proyecto consista en la nómina, contabilidad, facturación y gestión de la producción, todo de una pieza, por varias razones, tales como:

- El tamaño de un proyecto tiene una relación directa y superior a la lineal con el riesgo de fracaso.
- Los costes de coordinación suben tanto que la productividad media del personal baja, estando correlacionada de forma inversa al tamaño del proyecto.

- Las visiones actuales de desarrollo de software se aproximan cada vez más al desarrollo iterativo e incremental. Y éste consiste en implementar subsistemas hasta alcanzar el sistema completo.
- Dado que un proyecto de gran tamaño debería seccionarse para su implementación. Lo lógico es que sea la dirección estratégica de la empresa la que identifique los subsistemas más críticos, y primeros candidatos a ser desarrollados, y no que sean los desarrolladores de software los que tomen esta decisión.

Por todo lo anterior, podemos suponer que el proyecto se refiere a un solo producto, así pues veremos como primer paso una descomposición del proyecto en fases (procesos) y en un paso posterior refinaremos esta descomposición identificando las tareas.

Una vez conocidas las tareas a realizar se deberá programar (planificar), el proceso de desarrollo y asignarse los recursos, fundamentalmente, humanos. La programación de proyectos la llevaremos a cabo utilizando las técnicas matemáticas de la asignatura Investigación Operativa conocidas como PERT o CPM, que veremos más adelante.

2.2. Descomposición en actividades del proyecto (WBS)

Empezaremos por ver la herramienta que se utiliza a la hora de descomponer y documentar el trabajo de un proyecto, como un conjunto de tareas. Habitualmente se le conoce como WBS (Work Breakdown Structure) que literalmente significa estructura de descomposición del trabajo. Es un método para representar de forma jerárquica los componentes de un proceso o producto. Puede ser utilizado para documentar la descomposición de un proceso, la descomposición de un producto, o de forma híbrida.

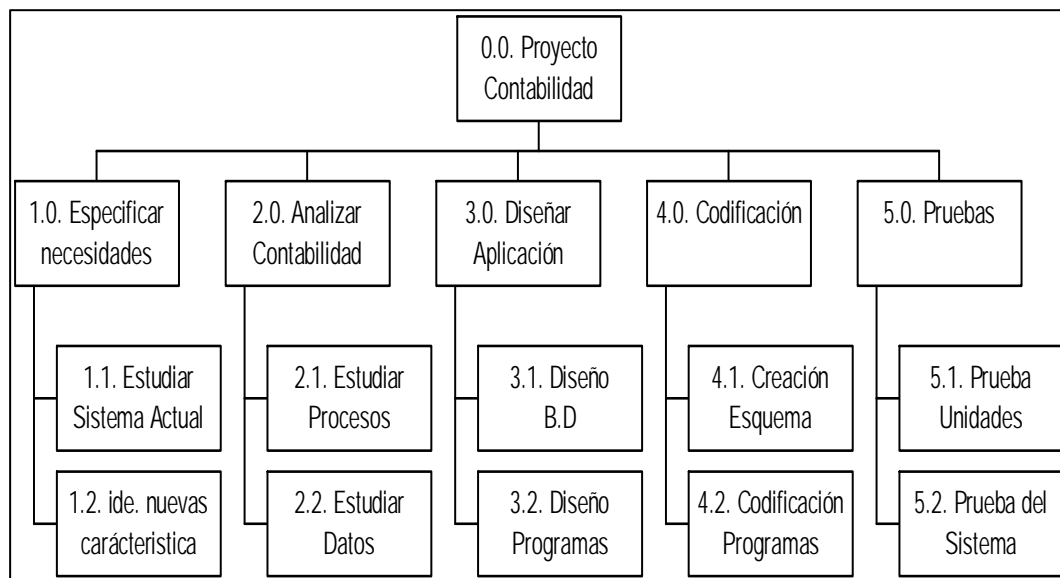


Figura 2.1.

Hay dos formas de representar un WBS. La primera es mediante una representación gráfica, en forma de árbol, como se muestra en la Figura 2.1. La segunda consiste en una lista indentada de tareas, como muestra la Figura 2.2.

En ambos casos se muestra la misma descomposición del trabajo. Los números se usan para etiquetar los nodos, de forma que dado un componente resulte fácil localizarlo en la estructura. Dado un nodo de la estructura decimos que contiene a todos sus descendientes, y está contenido en su antecesor. Así el nodo “2.0. Analizar la Contabilidad” está contenida en el nodo “0. Proyecto Contabilidad”, y contiene a los nodos “2.1. Estudiar Procesos” y “2.2. Estudiar Datos”.

Para crear un WBS empezaremos por clarificar la utilidad que se desea de esta estructuración. Nombramos un primer nodo con el nombre del proyecto. Identificamos componentes de este nodo, tenemos que intentar crear una estructura en la que cada nodo tenga del orden de 7 ± 2 componentes. Numeramos los nodos por niveles. Las tareas son los

nodos del nivel más bajo, las que no se descomponen más. Los nodos que se descomponen no indica que hay una tarea, sino el conjunto de tareas de las que se compone.

En los proyectos informáticos es importante generar una ficha para cada tarea identificada, donde anotaremos su número, nombre, una breve descripción y el esfuerzo estimado.

<ul style="list-style-type: none"> 0. Proyecto Contabilidad. <ul style="list-style-type: none"> 1. Especificar Necesidades. <ul style="list-style-type: none"> 1.1. Estudiar Sistema Actual. 1.2. Añadir Nuevas Características. 2. Analizar Contabilidad. <ul style="list-style-type: none"> 2.1. Estudiar Procesos. 2.2. Estudiar Datos. 3. Diseñar Aplicación. <ul style="list-style-type: none"> 3.1. Diseño B.D. 3.2. Diseño Programas. 4. Codificación. <ul style="list-style-type: none"> 4.1. Construcción del Esquema. 4.2. Codificación de los Programas. 5. Pruebas. <ul style="list-style-type: none"> 5.1. Prueba de Unidades. 5.2. Prueba del Sistema.
--

<p style="text-align: center;"><u>Especificación de tarea</u></p> <p>Número: 3.1.</p> <p>Nombre: Diseño B.D.</p> <p>Descripción: Se diseñará la base de datos, partiendo del modelo relacional propuesto en el análisis y con el objetivo de tener un sistema funcionando sobre MySQL.</p> <p>Esfuerzo Estimado: 2 semanas/hombre</p> <p>Entregables: Estructura de implementación de la B.D.</p>

Figura 2.2.

2.3. Entregables de un proyecto informático

Dado que el objetivo final del proyecto es la entrega de un subsistema informático (entregable), vamos a ver algunas definiciones y utilidades de los entregables. Los entregables los definiremos como: “Productos que, en un cierto estado, se intercambian entre los clientes y los desarrolladores a lo largo de la ejecución del proyecto informático”.

Los entregables los clasificamos como relativos al objetivo y relativos a la gestión del proyecto. Son entregables relativos al objetivo todos aquellos documentos que hacen referencia exclusivamente al sistema de información y al subsistema informático en desarrollo. Pertenecen a este conjunto los requisitos del sistema, la especificación del sistema, la documentación del diseño, el código fuente, los programas ejecutables, los manuales de usuario, etc. Los entregables relativos a la gestión del proyecto son aquellos documentos que se refieren a la situación en que se encuentra un proyecto, previsiones de costes, gastos realizados, informe sobre ambientes de trabajo, etc., siendo su objetivo poder controlar el proyecto. Pertenecen a esta clase la planificación del proyecto, los presupuestos, los documentos de control de la planificación o de la calidad, los estudios de riesgos durante el desarrollo, etc.

Se deberá definir de forma clara el conjunto mínimo de entregables necesarios para dar por terminada cada fase de desarrollo. Aunque algunos entregables se desarrollan a lo largo de varias tareas. Los entregables nos proveen de:

- Un conjunto de componentes que formarán el producto una vez finalizado el desarrollo.
- Los medios para medir el progreso y la calidad del producto en desarrollo.
- Los materiales necesarios para la siguiente etapa.

2.3.1. Entregables más usuales de un proyecto

Dado que como hemos visto los entregables juegan un papel central en el desarrollo de un subsistema informático, vamos a nombrar los más importantes. Basándonos en el Capítulo 4 de King tenemos:

- Estudio de viabilidad:
 - Descripción breve del sistema propuesto y sus características.
 - Descripción breve de las necesidades del negocio en el sistema propuesto.
 - Propuesta de organización del equipo de desarrollo y definición de responsabilidades.
 - Estudio de los costes, que contendrán estimaciones de la planificación y fechas tentativas de entrega de los productos.
 - Estudio de los beneficios que producirá el sistema.
- Análisis:
 - Captura de requisitos:
 - Análisis del sistema actual (si existe).

- Requisitos nuevos de los usuarios.
 - Descripción del sistema propuesto.
- Especificación del sistema:
 - Descripción del sistema.
 - Requisitos de datos.
 - Requisitos de telecomunicaciones.
 - Requisitos de hardware.
 - Plan de pruebas de integración.
- Diseño:
 - Descripción detallada del sistema:
 - Programas, módulos reutilizables y objetos.
 - Ficheros y bases de datos.
 - Transacciones.
 - Diccionario de datos.
 - Procedimientos.
 - Carga del sistema y tiempos de respuesta.
 - Interfaces, tanto humanos como de máquinas.
 - Descripción de los controles del sistema propuestos.
 - Diseños alternativos recomendados.
 - Estándares de programación y diseño de programas recomendados.
 - Técnicas de implementación recomendadas: codificación propia, compra de paquetes, contratación externa, etc.
 - Plan de pruebas de programas.
- Codificación:
 - Documentos del diseño final del sistema y de cada programa.
 - Diagramas definitivos del sistema y de los programas.
 - Descripción detallada de la lógica de cada programa.
 - Descripción de las Entradas y Salidas (ficheros, pantallas, listados, etc.).
 - Listado de los programas, conteniendo comentarios.
 - Cadenas de ejecución si es necesario (JCL, scripts, etc.).
 - Resultado de las pruebas de cada unidad.
 - Resultado de las pruebas de cada programa.
 - Resultado de las pruebas de la integración.

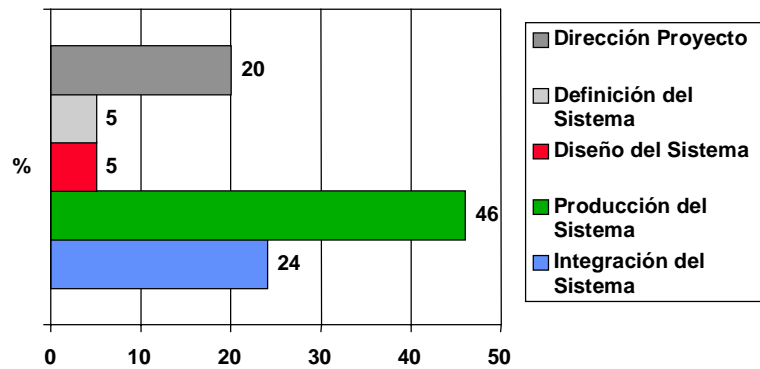
- Guía para los operadores del sistema.
- Programa de entrenamiento de los operadores.
- Manual de usuario del sistema.
- Pruebas:
 - Plan de pruebas del sistema (actualizado).
 - Informe de los resultados de las pruebas.
 - Descripción de las pruebas, el resultado esperado, resultado obtenido y acciones a tomar para corregir las desviaciones.
 - Resultados de las pruebas a la documentación.
- Instalación:
 - Planes detallados de contingencias de explotación, caídas del sistema y recuperación.
 - Plan de revisión post-instalación.
 - Informe de la instalación.
 - Carta de aceptación del sistema.
- Mantenimiento:
 - Listado de fallos detectados en el sistema.
 - Listado de mejoras solicitadas por los usuarios (sino dan lugar a nuevos proyectos).
 - Traza detallada de los cambios realizados en el sistema.
 - Actas de las revisiones regulares del sistema y aceptación de los niveles de soporte.

A todos estos documentos hay que añadir en todas las fases documentos con la estimación y planificación de la próxima fase y del resto del proyecto. También habrá que ir actualizando el índice de todo el material relacionado.

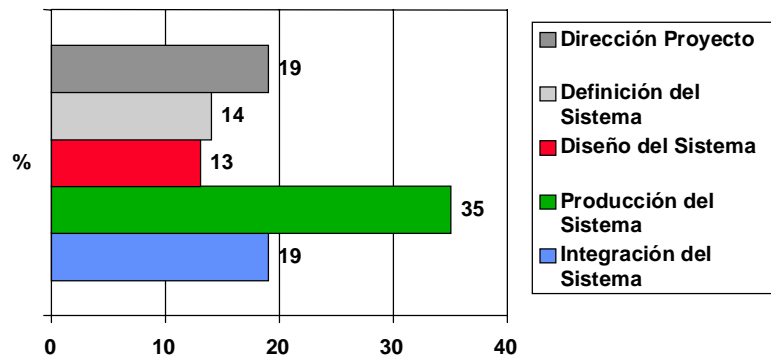
2.4. Descomposición en fases del desarrollo de una aplicación

La descomposición por fases (actividades) se basa en referencias históricas de la empresa que asocian una cantidad media de horas de trabajo a una actividad concreta, de modo que dado un proyecto concreto podemos estimar la cantidad de esfuerzo que se dedicará a esa actividad. En ésta se ha de tener en cuenta el tipo de proyecto, el lenguaje de desarrollo y la maduración de la organización. Martyn A. Ould, desde una perspectiva histórica, ofrece en las siguientes gráficas estos datos comparativos.

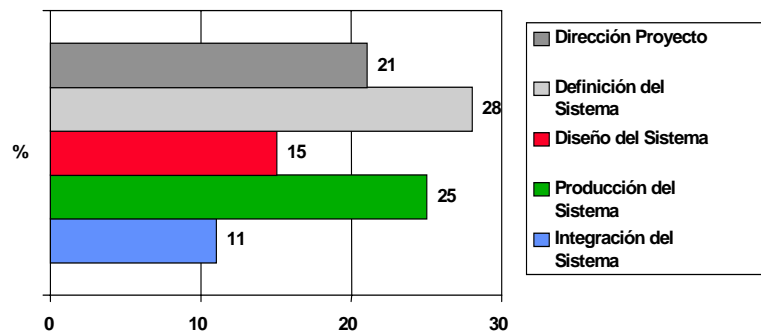
Reparto del Esfuerzo a mediados de los '70



Reparto del Esfuerzo a principios de los '80



Reparto del Esfuerzo a finales de los '80



Estos datos muestran la evolución de la empresa, pero también el cambio de lenguajes y entornos de trabajo.

Las empresas deben identificar las fases del ciclo de vida o actividades importantes de desarrollo de sus aplicaciones y almacenar el consumo de recursos (esfuerzo) aplicado en cada uno de éstas. Es aconsejable identificar aquellas componentes del desarrollo que supongan un consumo substancial de recursos.

2.5. Descomposición del desarrollo de la aplicación en tareas

Podemos plantear la descomposición desde el enfoque de entregables y asociar las tareas a la producción de un entregable concreto. Este enfoque tiene la ventaja de que la culminación de una tarea indica que ha concluido un producto y viceversa. Dado que, como veremos, no es aconsejable tener tareas que duren más de una semana, se plantean problemas con algunos entregables que cuestan más.

El planteamiento de descomponer por procesos o actividades puede resultar más natural en algunos casos. Es más fácil conseguir tareas acotadas en el tiempo. Tiene la desventaja de que el proyecto no será tan fácil de controlar, ya que en muchos casos será la palabra de los realizadores la única constancia de que la tarea está terminada o al “90%”.

En cualquier caso los proyectos se planifican con dos horizontes, el de la próxima fase y el del proyecto completo. El horizonte de la próxima fase se realiza con mayor nivel de detalle, mientras que según se alejan las fases se aplica un menor nivel de detalle.

2.5.1. Enfoque de equipo en la identificación de tareas por actividades

La descomposición del proyecto con mayor nivel de refinamiento no puede basarse en datos recogidos de forma analítica, sino que hace falta una aportación personal de los miembros del equipo de trabajo, tanto para identificar tareas como para asignarles esfuerzos. Se suele aconsejar el trabajo en grupo donde todos puedan aportar sus conocimientos y experiencias previas.

Hay que tener en cuenta que si identificamos las tareas y se las imponemos a los desarrolladores, éstos funcionarán en una situación de sumisión lo que puede tener efectos perniciosos tanto para los plazos de entrega como para la calidad del software. Por otra parte, dejar que sean los propios desarrolladores los que identifiquen tareas y recursos, dentro de un marco razonable les llevará a una situación de compromiso personal, pasando a interiorizar los objetivos y como consecuencia obtendremos mejores resultados.

2.5.2. Formas usuales de identificar tareas asociadas a un entregable

Hay que tener en cuenta que la tarea fundamental de los desarrolladores es escuchar a los clientes o usuarios y traducir sus requisitos a un lenguaje comprensible por la máquina, de

modo que el subsistema informático se adapte a las necesidades expresadas. Así para cualquier tarea podremos encontrar las siguientes subtareas:

- Documentarse, Buscar o Investigar,
- Organizar, Escribir Documentos,
- Verificar, Comprobar,
- Revisar, Actualizar Documentos,
- Entregar, Finalizar.

Además de lo anterior hay que tener en cuenta que al ir desarrollando el sistema obtenemos información que nos será útil a la hora de identificar nuevas tareas. Por ejemplo, el análisis estructurado nos provee de una descomposición del proyecto por productos: transacciones, archivos, entradas, salidas, etc. El diseño de programas descompone el sistema por módulos, el diseño de BD descompone por tablas, archivos, etc., y los diseños de interfaz por pantallas, listados, mensajes, etc.

2.5.3. Tareas usuales de un proyecto informático

Siguiendo la estructura de los entregables enunciados anteriormente y basándonos en el Capítulo 4 de King identificamos las siguientes tareas:

- Estudio de viabilidad:
 - Definir y documentar posibles tipos de sistemas.
 - Hacer un análisis de coste de sistemas similares.
 - Hacer una estimación del tamaño del sistema, la planificación y los costes (tener en cuenta los entregables más importantes).
 - Definir cualitativa y cuantitativamente los beneficios del sistema propuesto.
 - Realizar una planificación inicial del plazo de recuperación de la inversión.
 - Realización de una estimación detallada de costes, planificación, recursos, etc., de la siguiente fase (Análisis).
 - Asignar director del proyecto.
 - Componer el documento de estudio de viabilidad.
 - Presentar el documento de viabilidad a la dirección para su aprobación.
- Análisis:
 - Definir y documentar posibles tipos de sistemas.

- Captura de requisitos:
 - Definir el ámbito del sistema propuesto:
 - Funciones.
 - Dimensiones.
 - Usuarios.
 - Restricciones.
 - Entrevistar a todos los usuarios propuestos y actuales:
 - Determinar la utilización del sistema actual:
 - ◆ Deficiencias del sistema actual.
 - ◆ Requisitos nuevos del sistema.
 - Documentar:
 - ◆ Descripción del sistema actual.
 - ◆ Deficiencias del sistema actual.
 - Producir el documento de requisitos del nuevo sistema:
 - Requisitos del usuario priorizados.
 - Resoluciones sobre las deficiencias del sistema actual.
 - Producir una lista de los beneficios tangibles e intangibles (un refinamiento de la lista del estudio de viabilidad).
 - Realizar una estimación detallada de costes, planificación, recursos, etc., de la siguiente subfase (Especificación del sistema).
 - Producir una estimación revisada de costes, planificación, recursos, etc., para el resto del proyecto.
 - Producir el documento de definición de requisitos (esta tarea incluye la construcción de un prototipo).
 - Realizar una revisión final del documento de requisitos.
 - Tomar la decisión de continuar o no con el proyecto.
 - Definir las responsabilidades en la próxima subfase para el director, miembros del equipo de desarrollo y otros.
- Especificación del sistema:
 - Definir el tipo de sistema propuesto: Transformar las restricciones físicas, ambientales y operacionales a características del sistema; por ejemplo ¿sistema basado en transacciones?, ¿distribuido o centralizado?, ¿estaciones de trabajo o terminales?

- Esquematizar el sistema propuesto: Transformar los requisitos del usuario de la fase anterior en unas especificaciones funcionales (DFD, Organigramas, etc.).
 - Construir el diccionario de datos: Describir todos los elementos del DFD incluyendo funciones y datos; asegurarse de que todas las relaciones inter-funcionales y entre datos sean documentadas. Si existe DD de la empresa, hacerlo compatible.
 - Revisar y expandir el análisis de coste-beneficio: Actualizar con la información nueva; verificar que los beneficios esperados se mantienen y que el plazo de recuperación de la inversión sigue siendo aceptable.
 - Realizar una estimación detallada de costes, planificación, recursos, etc., de la siguiente fase (Diseño).
 - Producir una estimación revisada de costes, planificación, recursos, etc., para el resto del proyecto.
 - Producir el documento de especificación del sistema.
 - Realizar una revisión final del documento de especificación del sistema.
 - Tomar la decisión de continuar o no con el proyecto.
 - Definir las responsabilidades en la próxima fase para el director, miembros del equipo de desarrollo y otros.
- Diseño:
 - Producir el diseño global del sistema incluye:
 - Definir los programas y sus principales funciones.
 - Definir los principales flujos de datos entre programas y funciones.
 - Diseñar el esquema de datos lógico y físico.
 - Definir las fronteras con paquetes software, si existen.
 - Definir los entornos de hardware y software, proponiendo alternativas.
 - Documentar los diagramas de diseño alternativos.
 - Localizar paquetes software: Buscar paquetes software apropiados que puedan implementar parte, o toda la funcionalidad requerida del sistema de forma rentable y que, si se implementa, ofrezca un entorno compatible con los objetivos de la organización (puede realizarse antes del diseño, o de forma simultánea a la tarea anterior).
 - Desarrollar un diseño detallado del sistema para cada alternativa de diseño planteada:
 - Crear una descripción narrativa detallada del diseño para todo el sistema y cada una de sus partes (programas, funciones y datos).
 - Actualizar el diccionario de datos.

- Definir los componentes hardware específicos (capturadores de datos, sistemas de comunicación, etc.) y sus funciones.
- Validar el diseño con las especificaciones del sistema.
- Documentar el entorno hardware y software necesarios para esta alternativa.
- Revisar y expandir el análisis de coste-beneficio para cada alternativa:
 - Actualizar con la información nueva.
 - Verificar que los beneficios esperados se mantienen y que el plazo de recuperación de la inversión sigue siendo aceptable.
- Evaluar las alternativas de diseño y para cada alternativa documentar:
 - Requisitos de usuario que se alcanzan con dicha alternativa.
 - Nivel de aceptación esperado de los usuarios.
 - Realización de una estimación detallada de costes, planificación, recursos, etc., de la siguiente fase (Codificación) con esta alternativa.
 - Producir una estimación revisada de costes, planificación, recursos, etc., para el resto del proyecto.
 - Alternativa recomendada.
- Desarrollar un plan de test del sistema incluye:
 - Crear datos de entrada del test.
 - Producir el listado de los resultados esperados.
 - Producir el listado de los criterios de test.
 - Desarrollar la planificación de test del sistema.
- Desarrollar un plan de test diferenciado para cada alternativa.
- Identificar las necesidades de entrenamiento y documentación de los usuarios, es decir, definir las guías de:
 - Documentación completa de usuario.
 - Manuales de operador.
 - Documentos y planificación de formación para usuarios y operadores.
- Producir el documento de diseño del sistema.
- Realizar una revisión final del documento de diseño del sistema.
- Tomar la decisión de continuar o no con el proyecto.
- Recomendar una alternativa.
- Hacer recomendaciones sobre el nivel de compromiso, si los hay, de programadores subcontratados y otros.

- Definir las responsabilidades en la próxima fase para el director, miembros de los equipos de programación y test, así como de otros implicados.
- Codificación:
 - Producir un plan de trabajo:
 - Crear la lista detallada de tareas necesarias para realizar la codificación y las pruebas (test) de todos los componentes del sistema.
 - Producir una planificación para las tareas anteriores con las fechas más tempranas y más tardías, así como la asignación de responsabilidades.
 - Instaurar los procedimientos para recoger los progresos y estados del proyecto.
 - Instaurar los procedimientos para recoger tiempos, si resulta apropiado.
 - Obtener la aprobación del plan de trabajo por parte de la dirección.
 - Realizar el diseño detallado de cada programa:
 - Diseñar detalladamente los diagramas:
 - De estructura de los programas.
 - De estructura de los ficheros.
 - Pantallas, informes, y otras composiciones.
 - Esquemas de la base de datos.
 - Composición de las tablas y sus diseños.
 - Realizar pseudocódigo de la lógica del programa, que dependerá de los métodos de diseño utilizados.
 - Codificar, documentar y pasar los tests en cada programa:
 - Codificar el programa y los procedimientos de control (JCL).
 - Realizar las pruebas de unidad, hasta que los programas se adapten a las especificaciones descritas en las etapas anteriores.
 - Actualizar todo lo necesario en el sistema y en el DD de la organización.
 - Realizar el test de integración:
 - Poner todos los programas probados en la librería de pruebas de integración.
 - Realizar el test de integración de cada programa.
 - Documentar todos los resultados del test de integración.
 - Terminar los manuales de operador y usuario, así como los de formación.
 - Realizar una estimación detallada de costes, planificación, recursos, etc., de la siguiente fase (Pruebas).

- Producir una estimación revisada de costes, planificación, recursos, etc., para el resto del proyecto.
- Confeccionar el documento de diseño de programas y codificación.
- Realizar revisiones del documento de diseño de programas y codificación.
- Obtener los resultados finales de la integración completa del sistema y de las pruebas de integración.
- Definir las responsabilidades en la próxima fase para el director, miembros del equipo de test, así como de otros implicados.
- Pruebas:
 - Realizar el test del sistema:
 - Hacer el test de sistema de acuerdo al documento de test del sistema.
 - Verificar la operatividad de los manuales de usuario y operador, utilizándolos en los cursos de formación de los usuarios y operadores que realicen el test del sistema.
 - Verificar los documentos de entrenamiento de usuarios y operadores, utilizándolos en los cursos de formación de los usuarios y operadores que realicen el test del sistema.
 - Documentar completamente los resultados del test del sistema.
 - Revisar la planificación de instalación:
 - Disponibilidad de los recursos.
 - Revisión de los factores de contingencia que puedan afectar a la instalación.
 - Procesos especiales de final de mes y fin de año.
 - Vacaciones y fiestas.
 - Disponibilidad de soporte por parte de terceros vendedores.
 - Revisión final del calendario de instalación.
 - Esbozar el plan ante caídas:
 - Criterios para las caídas.
 - Identificación de recursos para contingencias.
 - Horario para recuperaciones o abandonos.
 - Desarrollar un acuerdo de nivel de servicio:
 - Criterios de rendimiento de usuario, precisión y volumen.
 - Criterios de apoyo de los vendedores.
 - Tiempo medio entre fallos.
 - Tiempo medio de reparación.

- Criterios de calidad del sistema.
 - Frecuencia de medición.
- Producir los documentos de test en la entrega.
- Revisar y aprobar los documentos de entrega.
- Aprobar la documentación del sistema:
 - Documentación de programas.
 - Manuales de operador.
 - Manuales de usuario.
 - Manuales de formación.
 - Documentación de ayuda.
- Aprobar el plan de instalación.
- Aprobar los planes de contingencia, recuperación y caídas.
- Finalizar el sistema completamente probado.
 - Acuerdo de finalización del desarrollo del sistema.
 - Acuerdo de finalización de los usuarios.
 - Acuerdo de finalización de aseguramiento de la calidad.
 - Acuerdo de finalización de finanzas.
- Instalación:
 - Instalar el hardware y el software nuevo.
 - Formar a los primeros usuarios y operadores.
 - Desarrollar los planes de contingencia, recuperación y caída.
 - Desarrollar los procedimientos de mantenimiento y versiones.
 - Establecer procedimientos para:
 - Versiones regulares.
 - Versiones de emergencia.
 - Versión por configuración (hardware o estaciones de trabajo).
 - Llevar a cabo cualquier conversión de datos necesaria.
 - Llevar a cabo la instalación del sistema nuevo a producción.
 - Instalación completa desde cero.
 - Instalación en paralelo.
 - Instalación por fases.
 - Comenzar el uso de los acuerdos de nivel de servicio.
 - Planificar y programar las revisiones post-instalación:

- Establecer los criterios de:
 - Rendimiento del sistema.
 - Calidad del sistema.
 - Satisfacción del usuario.
 - Calidad y facilidad de manejo de:
 - ♦ Manuales de usuario y operador.
 - ♦ Formación de usuarios y operadores.
 - ♦ Información y datos producidos.
 - Fluidez de la instalación.
 - Costes de desarrollo, instalación, operaciones y mantenimiento.
- Establecer la planificación y calendario para las revisiones.
- Asegurar la disponibilidad de:
 - Personal requerido.
 - Documentación requerida.
- Llevar a cabo las revisiones post-instalación:
 - Crear el informe de la revisión post-instalación.
 - Obtener la aprobación firmada de los informes de:
 - Usuarios finales del sistema.
 - Operadores del sistema.
 - Auditoria y aseguramiento de la calidad.
 - Desarrollo de sistemas.
 - Soporte de sistemas y mantenimiento.
 - Finanzas.
 - Obtener la carta de aprobación del sistema.
- Establecer el calendario para otras revisiones post-instalación, si es necesario.
- Mantenimiento:
 - Implementar los cambios del sistema:
 - Utilizar los procedimientos de implementación de versiones, o
 - Implementar versiones de emergencia y después utilizar los procedimientos de versiones formales de forma retroactiva.
 - Asegurarse de que el sistema continúa solucionando las necesidades de los usuarios:
 - Utilizar los acuerdos de niveles de soporte:
 - Revisiones regulares de requisitos del nivel de acuerdo.

- Revisiones regulares de cómo el sistema está alcanzando sus objetivos.
- Llevar a cabo revisiones regulares del sistema.
- Utilizar los procedimientos y contenido de las revisiones post-instalación.

Estas tareas se han enumerado a modo de lista de comprobación, de forma que serán los desarrolladores los encargados de identificar las tareas apropiadas a cada proyecto así como los recursos necesarios, teniendo en cuenta la estimación previa del esfuerzo.

2.6. Algunas reflexiones sobre la descomposición de un proyecto en tareas

Edward Yourdon sugiere algunas reglas a la hora de descomponer un proyecto en tareas, entre las que cabe recordar:

- a) Hacer las unidades de estimación tan pequeñas como se pueda, a ser posible que se aproximen a la semana.
- b) Que las tareas sean tan independientes como se pueda, es decir, no cortar procesos naturales como la codificación de un módulo en varias tareas.
- c) Tener en cuenta los factores de comunicación entre personas, haciéndolo sencillo.
- d) Tener en cuenta la posibilidad de reutilizar código, siendo conscientes de que también es trabajo buscarlo y adaptarse a este código.

REFERENCIAS

1. King, D., *Project Management Made Simple*, Prentice Hall.
2. O'Connell, F., *How to run successful projects*, Prentice Hall.
3. Ould, M.A., *Strategies for Software Engineering*, John Wiley.
4. Yourdon, E., *Análisis Estructurado Moderno*, Prentice Hall.