Este documento ha sido generado para facilitar la impresión de los contenidos.
Los enlaces a otras páginas no serán funcionales.

Caso práctico

*Como recordarás, **SINUNEURO** (**S**ervicios **IN**mobiliarios de la **UN**ión **EURO**pea) es una empresa dedicada a la venta y alquiler de locales, solares y viviendas. **Julia** y **María** están trabajando de forma conjunta en el desarrollo de una aplicación software para la gestión de la empresa. **Julia** representa al cliente, **SINUNEURO** y **María** a **SI Andalucía**, empresa que se va a dedicar a elaborar el proyecto.*

*Entre ambas mujeres tienen que realizar la gestión del proyecto de desarrollo del software que necesita la empresa **SINUNEURO.** Juntas van a compartir el cargo de Jefa del Proyecto, por lo que **María** hace a **Julia** una explicación de lo que consiste su trabajo para que tenga una visión global de todo el proceso, desde el significado de proyecto hasta la concreción de los objetivos que persiguen, pasando por la necesidad de planificar y descomponer el trabajo, sopesar los riesgos y establecer las pautas de control y supervisión.*

Introducción a la gestión de proyectos

Estarás de acuerdo con nosotros en que **gestionar** cualquier tipo de proyecto no es algo sencillo, sobre todo si se trata de proyectos complejos por su naturaleza, o por su tamaño, y existen técnicas que permiten facilitar esa gestión. Pero a nosotros no nos interesa cualquier tipo de proyecto, sino que estamos ocupados en aprender a **gestionar proyectos software**.

¿Existen **técnicas** que nos faciliten la gestión de nuestros proyectos software? Seguro que piensas que sí, y no andas equivocado. Existen, y en esta unidad vamos a tratar de aproximarte a ellas.

Podemos decir que el **método de trabajo** habitual en el Desarrollo del Software es mediante **proyectos**. La decisión de emprender un proyecto puede ser consecuencia de diferentes situaciones o circunstancias. Sin embargo cada proyecto depende de una gestión apropiada que le va a permitir alcanzar los objetivos marcados. Esta gestión viene condicionada por una serie de características específicas de cada proyecto.

Un proyecto consiste en un conjunto de etapas, actividades y tareas que tienen como finalidad alcanzar un objetivo, en un plazo, que se puede considerar, relativamente largo.

En general podemos decir que **un proyecto**...

- Tiene un principio y un final.
- Utiliza recursos finitos y cuenta con un presupuesto ajustado.

- Tiene actividades únicas y esencialmente no repetitivas.
- Tiene un objetivo, hacia el que se dirige todo el esfuerzo.
- Requiere de una persona responsable del proyecto y personal de desarrollo cuyos roles y estructura de equipo deben fijarse y desarrollarse.
- Tiene una planificación.
- Debe medir su progreso frente a esa planificación.
- Suele coexistir con otros proyectos y competir por los recursos.
- Posee agentes internos y externos, que deben ser identificados y tratados, porque suelen influir de una manera importante en el proyecto.

Precisamente es la **división en trabajos más sencillos** lo que va a permitir al personal del proyecto dominar la complejidad del proceso para desarrollar el software. En esta unidad abordaremos la gestión de proyectos de desarrollo de software a través de sus distintos aspectos:

Planificación

- **Previsiones.**
- **Objetivos a corto y largo plazo.**
- **Políticas organizativas.**
- **Procedimientos estándares.**
- **Análisis de tareas.**
- **Obtención y distribución de recursos.**

Organización

- **Descripciones del trabajo.**
- **Asignación de tareas a puestos de trabajo.**
- **Personal.**
- **Determinación de las relaciones. organizativas.**
- **Delegación de autoridad.**

Dirección

- **Motivación.**
- **Comunicación.**
- **Jefatura.**
- **Coordinación.**
- **Valoración de la ejecución.**
- **Resolución de conflictos.**
- **Mantenimiento de políticas de la empresa.**

Control

- **Supervisión.**
- **Comparación de la ejecución actual y la esperada.**
- **Decisión de acciones correctivas cuando sea preciso.**

El seguimiento adecuado en cada una de estas áreas determinará la toma de decisiones sobre la información de proyectos de software.

Para saber más

La Gestión de Proyectos sigue unas pautas preestablecidas que venían adaptándose continuamente y que cada vez más están estandarizadas. En el siguiente enlace puedes consultar algunas de las características que más se utilizan, así como un breve resumen de su importancia en el mundo empresarial.

Gestión de Proyectos **[Versión en caché]**

Resumen sobre la gestión de proyectos. Se trata de una especie de resumen de toda esta unidad. Aunque utiliza una terminología ligeramente distinta puede resultar muy interesante echarle un vistazo.

Técnicas de Programación **[Versión en caché]**

Planificación

*Una vez que **Julia** tiene una idea clara de lo que constituye el proyecto para el desarrollo del software, y fijadas las funciones del Jefe de Proyecto (en este caso jefas), **María** le inicia en algunas técnicas necesarias para realizar la planificación, como:*

- *La construcción del Plan de Proyecto, sobre el que deciden los puntos a incluir;*
- *La elaboración del Calendario del Proyecto con la organización del equipo de desarrollo;*
- *La identificación de los Riesgos y cómo reducirlos;*
- *El establecimiento de los compromisos con las diferentes técnicas necesarias como:*
  - *Diagramas de Hitos,*
  - *Diagramas de Gantt y*
  - *Redes de precedencia.*

*En este último apartado es donde **Julia** encuentra mayores problemas, ya que para entenderlas bien, ha tenido que aprender ciertos conceptos importantes, pero reconoce que ha conseguido una visión global del proyecto que le ha facilitado notablemente la planificación del mismo.*

***María** insiste que en cualquier caso esta planificación inicial debe ser revisada continuamente y comparar los resultados obtenidos en cada momento con los esperados, de modo que tengan el proyecto perfectamente controlado en cada una de sus fases.*

Introducción y conceptos generales

Seguro que no se te escapa la necesidad de hacer una buena **planificación** del proyecto, pero

- ¿Cuáles son los aspectos más importantes que debe cubrir la planificación?
- ¿Qué **objetivos** debe cubrir un buen plan?
- ¿Qué puntos debe incluir ese plan?

El área de **planificación de proyectos**, cubre dos aspectos importantes:

- La realización de un **plan de proyecto** por parte del jefe del mismo.
- La gestión de los **compromisos**.

El **jefe del proyecto** es la persona que tiene la responsabilidad de **planificar**, controlar y dirigir todas las actividades del proyecto. Muchas veces, estas responsabilidades suponen la **coordinación** e integración de actividades a través de distintas unidades organizativas y departamentos. Evidentemente la elección de un buen jefe debería ser el primer paso para iniciar el proyecto.

El primer cometido del jefe de proyecto es la realización del plan de proyecto. En este plan de proyecto debe:

- definir un conjunto de **tareas**,
- coordinadas en el **tiempo**,
- así como los **recursos** necesarios para cumplir los objetivos marcados a cada tarea con el propósito de satisfacer los compromisos adquiridos.
- Con independencia de su tamaño, todo proyecto debe incluir un **plan de trabajo** en el que se detalle de forma precisa lo que hay que hacer,
- cuándo y dónde hacerlo,

- así como quién lo debe realizar y el coste que implica, todo ello con el fin de garantizar el éxito del desarrollo.

Para obtener un plan de proyecto útil, éste debería facilitar los siguientes **objetivos**:

- Proporcionar a los directivos un **resumen del proyecto** que les ayude a tomar las decisiones adecuadas.
- Permitir en todo momento **supervisar** el progreso del proyecto a cualquiera de las partes implicadas (clientes y jefe del proyecto).
- Debe presentarse como un documento orientado al **cliente**.
- Debe ser el **documento base** del proyecto, aprobado por el cliente y actualizable a medida que avanza en su desarrollo.

Un plan de proyecto es único y puede variar considerablemente de uno a otro, pero se recomienda que incluya al menos los siguientes puntos:

- Un **resumen del proyecto** redactado de forma comprensible por cualquier persona. Debe especificar claramente los productos entregables, de modo que cuando se produzcan se compruebe que se ajustan al plan.
- Una lista de hitos que se pretenden alcanzar en un plazo determinado y sus responsables.
- Todos los **recursos** que se van a utilizar, ya sean humanos, materiales, herramientas software, etc., especificando el motivo de su utilización.
- Proceso de **revisión** de la planificación del proyecto, indicando quién, cómo y cuándo se realiza, así como con qué objeto.
- Vías de **comunicación** entre el equipo de desarrollo y el cliente.
- Un **diagrama de descomposición** del trabajo, mediante el cual el proyecto en cuestión será tratado por partes, más fáciles de manejar y que posteriormente deben ser integradas.
- Una lista del **personal** del proyecto y su asignación a las tareas del diagrama anterior.
- Una **red de actividades** que detalle la secuencia de tareas en el tiempo y su relación entre ellas. Especificando los responsables de cada una. Va a depender del modelo de ciclo de vida adoptado para el proyecto.
- Los **presupuestos** de esfuerzo y económicos, así como el calendario y plazos para todas las actividades, y por extensión para todo el proyecto.

Sobre el plan de proyecto existen **estándares**, como el IEEE-1998 para la estructura de planes de desarrollo de software, aunque este estándar se suele utilizar a modo de guía y cada empresa lo adapta a sus proyectos según sus propias conveniencias.

Selecciona el siguiente enlace para ver la estructura más o menos estándar de un plan de proyecto.

Estructura de un plan de proyecto

Para saber más

Hasta ahora has podido contrastar la importancia que tiene la Planificación en un proyecto de desarrollo, ya que de su adecuada realización va a depender, de forma muy importante, el éxito o fracaso de todo el proyecto. La Red nos ofrece muchos artículos sobre la planificación. El siguiente es un ejemplo pero te animamos a que busques algunos por tu cuenta.

Planificación de un proyecto **[Versión en caché]**

En el siguiente enlace puedes encontrar muchas más cosas sobre la planificación de proyectos.

Proyectos sobre sistemas **[Versión en caché]**

Actividades para la planificación de un proyecto (calendario)

Cualquier proyecto está sujeto a unos **plazos de ejecución**, y como imaginarás, los proyectos informáticos no son distintos en ese punto a cualquier otro. Pero hablar de plazos, supone hablar del calendario.

- ¿Qué debe incluir el calendario del proyecto?
- ¿Qué características debe tener?
- ¿Qué pasos y en qué orden debemos dar para elaborar un buen calendario?
- ¿Podremos elaborar un calendario sin tener en cuenta los recursos, los costes y las restricciones del proyecto?
- ¿Será necesario revisar y reajustar el calendario a lo largo de la ejecución del proyecto?

Éstas y otras preguntas son las que pretendemos contestarte en este apartado.

Uno de los principales **objetivos** del Plan de Proyecto, es la configuración del **calendario**, también denominado **programa de tiempos**. Básicamente consiste en una **representación gráfica** de todas las actividades del proyecto, que va a permitir al jefe del proyecto coordinar con efectividad a todo el equipo durante todo el desarrollo. Este **calendario** debería ser dinámico, admitiendo cambios a medida que se avanza, con el fin de ajustar plazos y tiempos ante las dificultades que pueden surgir durante el desarrollo.

Sin ese calendario, el **control del proyecto** es casi imposible. La dirección del proyecto puede ser extremadamente difícil si no han sido identificadas las actividades individuales y sus interrelaciones. El control del proyecto se basa en la supervisión periódica y en la comparación de los resultados con los previstos en el calendario. Si no existe calendario, es imposible estimar el estado del proyecto con certeza.

Para que un programa de tiempos (o calendario) sea **eficaz**, debe tener al menos las siguientes características:

- **Comprensible** por todos aquellos que van a utilizarlo.
- Suficientemente **detallado** para servir de base para medir y controlar el progreso del proyecto.
- Capaz de señalar las actividades críticas.
- **Flexible**, fácilmente modificable.
- Basado en estimaciones de tiempos coherentes según los compromisos adquiridos por los responsables de cada tarea.
- **Ajustable** a los recursos disponibles, de modo que si aumentan adecuadamente los recursos, mejora el calendario del proyecto.
- **Compatible** con los planes de otros proyectos con los que comparte los mismos recursos.

Pasos para elaborar el calendario de un proyecto

Para elaborar el **calendario** del proyecto, el jefe del mismo debe considerar una serie de pasos en el orden adecuado (como se indica en la tabla siguiente).

Pasos para elaborar un calendario del proyecto

1. Definición de los objetivos.
2. Descomposición de las actividades.
3. Relación entre las actividades.
4. Estimación de los tiempos y costes de las actividades.
5. Ajuste del programa de tiempos a las restricciones del proyecto.
6. Organización del equipo y asignación de recursos.
7. Revisión del calendario.

*CASO. A la hora de elaborar el calendario del proyecto, **Julia** no sabe por donde empezar, porque tiene grandes dificultades a la hora de plasmar los objetivos de forma clara y no olvidar nada importante. **María**, que tiene alguna experiencia en proyectos similares, le explica cómo deben ser definidos y qué es lo que se pretende con esa definición, y le pide a **Julia** que intente hacerlo por sí misma. Le explica que al principio ella lo hacía de forma progresiva, escribiendo inicialmente frases que después iba retocando hasta conseguir una definición óptima.*

*Tras un tiempo considerable Julia ha hecho una **definición de objetivos** basándose en el funcionamiento actual de la inmobiliaria y ha quedado bastante satisfecha:*

- *Consulta de Clientes.*
- *Consulta de Fincas.*
- *Generación de listados e informes.*
- *Alquiler de fincas a clientes.*
- *Venta de fincas a clientes.*
- *Contabilizar operaciones.*

***María** le comenta que ha hecho un gran trabajo pero que esos objetivos presentan algunas deficiencias, básicamente en el grado de detalle (es preciso definir las actividades a desarrollar para conseguir cada objetivo), en el establecimiento de un criterio de finalización claro así como la duración de cada una de las actividades y en la identificación de las metas a alcanzar. Entonces le muestra el borrador del calendario que ella está elaborando:*

***OBJETIVOS**:*

- *Gestión de alquiler y venta de fincas a clientes.*
- *Gestión contable y de personal de la inmobiliaria.*

***ACTIVIDADES**:*

- *Gestión de Fincas:*
  - *Alta de fincas.*
  - *Baja de fincas.*
  - *Modificación de datos de fincas.*
  - *Consultas. Listados e informes.*
  - *Permisos.*
- *Gestión de Clientes:*
  - *Alta de clientes.*
  - *Baja de clientes.*
  - *Modificación de datos de clientes.*
  - *Consultas. Listados e informes.*

- *Gestión de alquileres.*
  - *Elaboración de contratos.*
  - *Contabilización de operaciones.*
    - *Pagos por periodos fijos.*
  - *Finalización del alquiler.*
- *Gestión de Ventas.*
  - *Elaboración de contratos.*
  - *Contabilización de operaciones.*
- *Gestión de personal.*
  - *Altas.*
  - *Bajas.*
  - *Modificación de datos.*
  - *Consultas.*
  - *Gestión de Nóminas.*

DIAGRAMA DE DESCOMPOSICIÓN DE ACTIVIDADES (WBS):

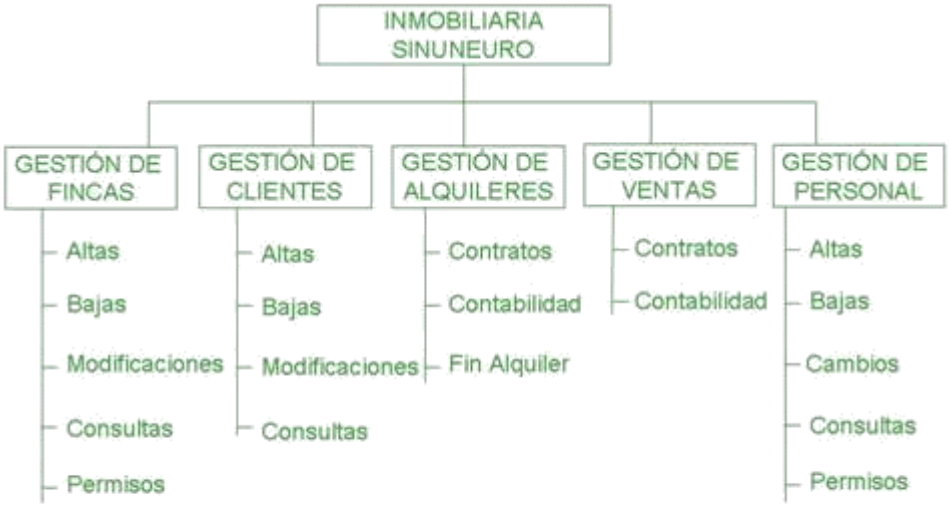


DIAGRAMA DE HITOS:

Actividades del Proyecto	Fecha prevista de finalización
Gestión de Fincas	30/09/06
Gestión de Clientes	06/10/06
Alquileres	09/11/06
Ventas	23/11/06
Gestión de Personal	15/12/06

DIAGRAMA DE GANTT

UNIDADES DE TIEMPO (semanas)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ACTIV. 1															
Tarea 1.1															
Tarea 1.2															
Tarea 1.3															
Tarea 1.4															
Tarea 1.5															
ACTIV. 2															
Tarea 2.1															
Tarea 2.2															
Tarea 2.3															
Tarea 2.4															
ACTIV. 3															
Tarea 3.1															
Tarea 3.2															
Tarea 3.3															
ACTIV. 4															
Tarea 4.1															
Tarea 4.2															
ACTIV. 5															
Tarea 5.1															
Tarea 5.2															
Tarea 5.3															
Tarea 5.4															
Tarea 5.5															

*Con este borrador **Julia** lo ve todo mucho más claro y parten del mismo para continuar su proyecto.*

Definición de los objetivos

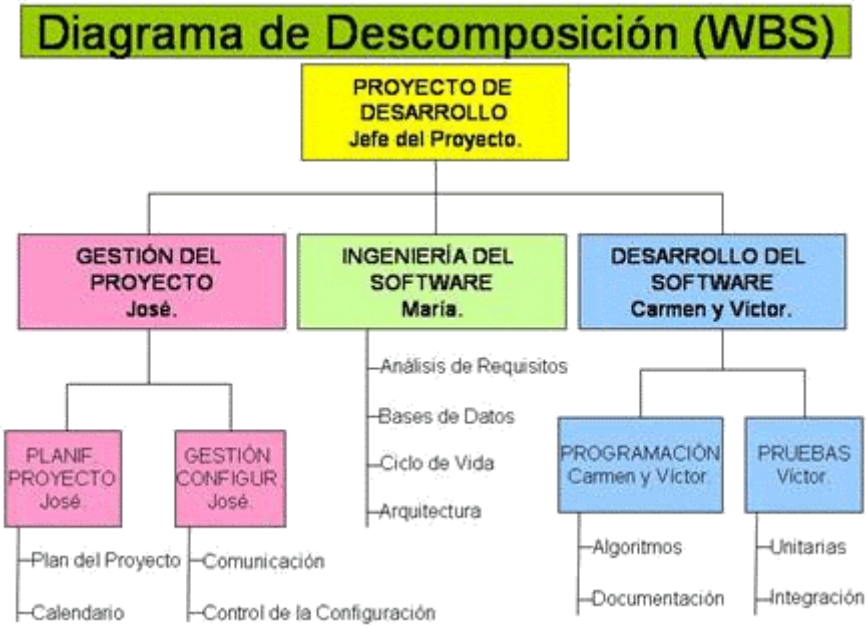
Los **objetivos** estarán definidos al comienzo para identificar las responsabilidades del equipo de desarrollo y serán revisados durante el proyecto para señalar los cambios que se apartan del alcance inicial. Un objetivo estará bien definido cuando es...

- **Asequible**. El objetivo identifica una meta y puede conseguirse con unos tiempos y unas restricciones dadas. Si la meta es demasiado ambiciosa, el objetivo puede perder credibilidad.
- **Definitivo**. Especifica concretamente qué es lo que se debe lograr y en qué grado de detalle. Sólo se consideran aquí los objetivos relacionados con el proyecto o metas organizativas. Las actividades rutinarias del proyecto no deben tomarse como objetivos.
- **Cuantificable**. Especifica un criterio de finalización.
- **De duración específica**. Define la duración de las actividades, lo que es imprescindible para evaluar el progreso del proyecto.

Descomposición de las actividades

El jefe del proyecto debe elaborar un diagrama de descomposición del trabajo (WBS- Working Breakdown Structure). Para ello en la **parte superior** se representa la actividad más general y a continuación, se subdivide en actividades más sencillas.

Inicialmente se identifican los paquetes de trabajo y después cada una de las tareas (y subtareas) que los producen. Con este diagrama el jefe de proyecto aumenta su capacidad de supervisión de los trabajos, y puede observar el seguimiento de actividades o tareas críticas que tienen una gran repercusión sobre el tiempo total del proyecto.



Relación entre las actividades

Normalmente el conjunto de **actividades** de un proyecto suelen estar relacionadas entre sí. La determinación del **calendario** del proyecto supone también la identificación de las interrelaciones entre las actividades que influyen en la secuencia que deben seguir las mismas y las repercusiones de lo que ocurra en una de ellas (por ejemplo un retraso) en las otras. Para representar esas interrelaciones se utilizan diferentes técnicas:

En proyectos sencillos ...

- **Diagramas de Hitos**
- **Diagramas de Gantt**
- **Diagramas full wall**

En proyectos grandes, técnicas basadas en Redes de Precedencia.

- **Redes PERT**
- **Redes CPM**

Algunas de estas técnicas se verán en detalle más adelante en esta misma unidad.

Estimación de tiempos y restricciones

Una vez que ha sido definida la secuencia entre las actividades, es necesario realizar una **estimación de los tiempos** (entre el comienzo y el final) **y los costes de las actividades**. Las estimaciones de tiempo se centran especialmente en el tiempo requerido para finalizar una

actividad y suelen estar basadas en la experiencia en proyectos similares. También es habitual que se contemplen los retrasos normales.

Ajuste del programa de tiempos a las restricciones del proyecto

Ajuste del programa de tiempos a las restricciones del proyecto. Básicamente es preciso calcular la **duración total** del proyecto partiendo de la duración correspondiente a cada una de las actividades que lo forman, para ello es imprescindible identificar y analizar las actividades críticas. Normalmente para llevar a cabo este tipo de tareas es necesario utilizar Redes de Precedencia.

Organización del equipo y asignación de recursos

Los **recursos disponibles** para un determinado proyecto, son muy **limitados**, por lo que se hace imprescindible organizar al equipo con una asignación de recursos ajustada en el tiempo a determinadas actividades. También es necesario que los costes proyectados se ajusten al **presupuesto** inicial.

Revisión del Calendario

Una vez elaborado el proyecto es preciso hacer una **revisión del calendario** para determinar si es **realista**. Se deben comprobar si los criterios adoptados (tanto en la estimación de tiempos como de costes) son razonables y realistas. Además debe contemplar la previsión de algunos retrasos e incluso debe tener cierta flexibilidad para adaptarse a imprevistos.

Para saber más

En este enlace se muestra un estupendo ejemplo de descomposición WBS con un proyecto de Geografía muy sencillo e intuitivo que puede aclarar muchas ideas.

WBS

Gestión de riesgos

¿Qué entendemos por **riesgo** en un proyecto? ¿Podemos controlar los riesgos? ¿Podemos eliminarlos por completo o nos tenemos que contentar con reducirlos a niveles razonables? ¿Son muy variados los riesgos que pueden presentarse en el desarrollo de un proyecto?

El tratamiento de **riesgos** del software en la realización de un proyecto, es un factor importante, ya que **podemos definir riesgo como cualquier elemento potencial que provoca resultados insatisfactorios en el proyecto**. Un Jefe de Proyecto debe conocer y controlar en todo momento el riesgo de aquellas áreas del proyecto que puedan provocar esos resultados, así como los riesgos **externos** que pueden afectar a su proyecto. También interesa establecer las circunstancias en que se producen los riesgos, los factores o causas desencadenantes y los efectos que pueden producir esas situaciones.

El **jefe del proyecto** debe dominar técnicas de **identificación de riesgos**, valoración de su impacto, seguimiento y control. El conocimiento de estas técnicas permitirá aumentar las probabilidades de éxito del proyecto.

Los **tipos de riesgos típicos** que debe tratar un jefe del proyecto son los siguientes:

- Riesgos **Estratégicos**. Relacionados con las pérdidas o beneficios, las inversiones, la imagen empresarial, etc.
- Riesgos **Comerciales**. Aquellos relacionados con la venta del proyecto, el precio, las actualizaciones, el seguimiento, etc.
- Riesgos **Contractuales y Financieros**. Reflejados en los términos del contrato; penalizaciones, obligaciones, calendario de pagos, etc.
- Riesgos de **Gestión**. Relacionados con la organización del proyecto; gestión de recursos, calendarios, estimaciones, etc.
- Riesgos de **Proyecto**. Causados por los aspectos técnicos del desarrollo del software; especificación, diseño, integración, validación, etc.
- Riesgos de **Explotación** y **Mantenimiento**. Se producen una vez que el producto es entregado y el cliente comienza a utilizarlo y se refiere a situaciones o acciones que tienen como consecuencia accidentes (ya sean físicos o de pérdida de datos).

Una vez identificados los riesgos, el objetivo es reducirlos hasta niveles aceptables para el jefe del proyecto.

Gestión de compromisos

En todo proyecto, como puedes suponer, intervienen varias **personas**, que pueden ser incluso muchas. El jefe del proyecto va estableciendo **tareas y pautas** que deben ir cumpliendo todas las personas implicadas en el proyecto, pero

- ¿Piensas que se conseguiría avanzar en el proyecto si esas personas no aceptasen como compromiso cada una de las actividades que le corresponde?
- ¿Será conveniente para ello imponer los compromisos, o por el contrario se obtendrán mejores resultados negociándolos?
- ¿Qué pasa si se fijan compromisos poco realistas?
- ¿Es conveniente cambiar los compromisos adquiridos una vez que se ha iniciado el proyecto, o deben mantenerse hasta el final? Este tipo de cuestiones es las que debe abordar la gestión de compromisos.

Es un aspecto **crucial** dentro de la realización de un proyecto software. Su objetivo es negociar, establecer y gestionar los compromisos adquiridos por todas las partes implicadas en el desarrollo del proyecto.

Recordemos que un equipo de desarrollo está **jerarquizado y perfectamente organizado**, y a cada uno de los responsables de cada fase o sección se les piden resultados (que exijan un esfuerzo) con un plazo concreto de finalización. Es conveniente que la **negociación** se haga contando con un compromiso negociado con cada participante o responsable de cada parte del proyecto. De este modo se consigue un **ambiente de colaboración** y trabajo en equipo que motiva a los participantes para cumplir y asumir su responsabilidad, y de paso facilitar la gestión de posibles cambios durante el proyecto.

A veces ocurre que los directivos se comprometen a cumplir objetivos **poco factibles** sin tener en cuenta la opinión de los técnicos del grupo de desarrollo, que son realmente los encargados de llevar a cabo el proyecto. Por este motivo el jefe del proyecto además, debe ayudar a quienes adoptan los compromisos, una vez que ha recabado las opiniones de los expertos.

Técnicas. Diagramas de Hitos

Anteriormente hemos hablado de la importancia de elaborar un buen calendario, pero como casi todas las cosas que son complicadas, disponemos de técnicas que nos ayudan a conseguirlo.

¿Quieres conocer algunas? Pues vamos a comenzar con los **diagramas de hitos**.

¿Y qué son los **hitos**?

Son las actividades del proyecto que marcamos, con fecha de comienzo y plazo de finalización, a lo largo del "largo camino" que nos lleva a culminar nuestro proyecto.

El **Diagrama de Hitos** es el método más sencillo para determinar el calendario.

Consiste en una tabla de dos columnas, en la que se reflejan las diferentes actividades del proyecto y la fecha correspondiente a su finalización. La consecución de cada una de las actividades en el tiempo previsto, se considera la consecución de un hito marcado.

Actividades del Proyecto	Fecha prevista de finalización
Inicio	23/06/06
Análisis	03/07/06
Diseño	15/07/06
Desarrollo	29/07/06
Pruebas de módulo	05/08/06
Instalación	08/08/06
Pruebas de Integración	11/08/06
Documentación	16/08/06
Validación	25/08/06

Las **ventajas** que presenta esta técnica son:

- **Es muy intuitiva y fácil de usar.**
- **El coste de preparación es mínimo.**

Y las **desventajas**:

- **Incertidumbre sobre las fechas** de comienzo de las actividades.
- **Imposibilidad de reflejar las interrelaciones** entre las actividades.

Este diagrama también se suele utilizar a modo de resumen de calendarios complejos que tienen muchas tareas.

Técnicas. Diagramas de Gantt

Como habrás imaginado, los diagramas de hitos no son la única técnica disponible, así que vamos a continuar presentándote alguna que otra más. La siguiente son los **Diagramas de Gantt**.

Se utiliza principalmente en **proyectos pequeños** (sobre las veinticinco actividades) y supera algunos de los inconvenientes de los diagramas de hitos. **Probablemente este diagrama sea el más utilizado** porque a primera vista parece más comprensible que las redes de precedencia.

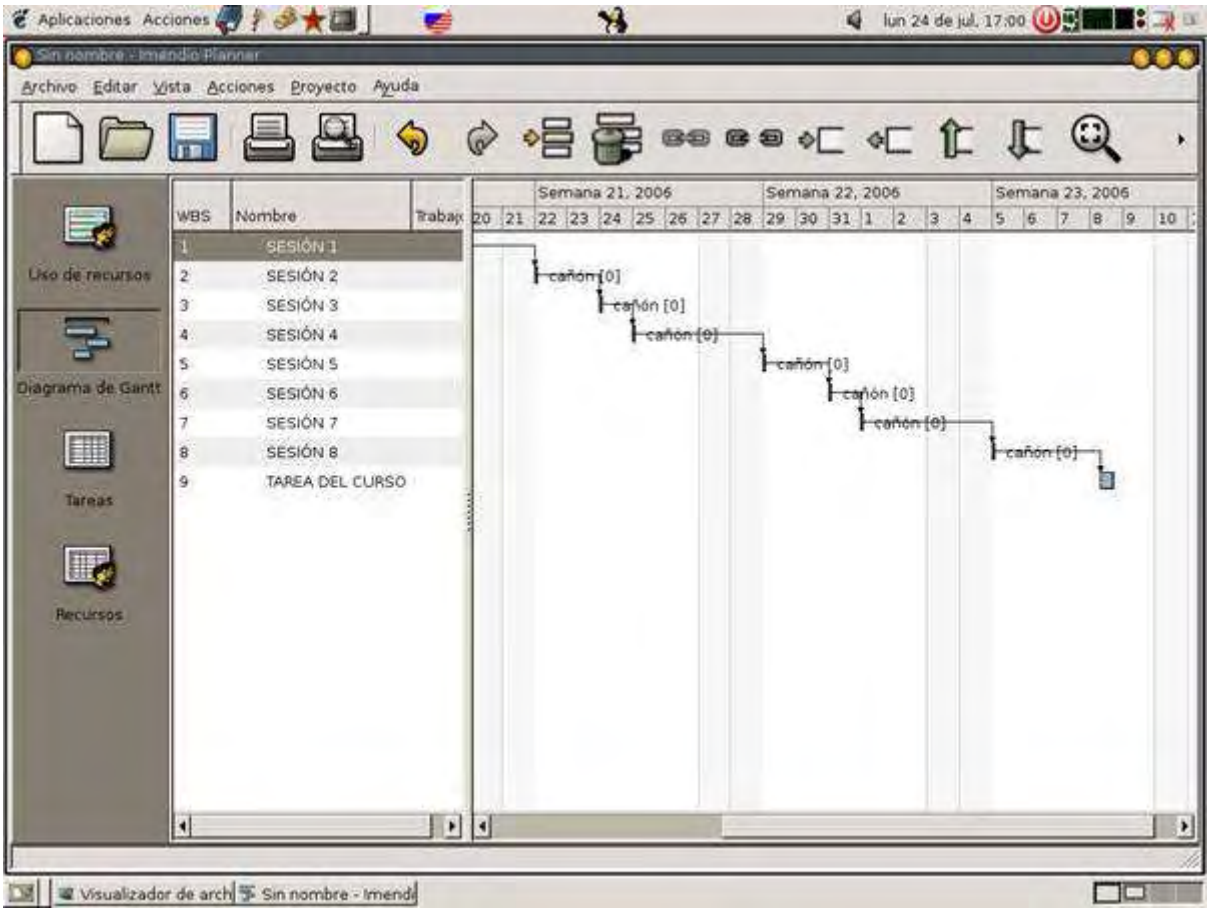
Consiste en un **diagrama de barras** en forma de tabla donde se hace una referencia cruzada entre las tareas (filas) y las unidades de tiempo (columnas) empleadas para medir la duración de las mismas.

TAREAS	UNIDADES DE TIEMPO									
	1	2	3	4	5	6	7	8	9	10
Tarea 1.1										
Tarea 1.2										
Tarea 1.3										
Tarea 2.1										
Tarea 2.2										
Tarea 2.3										
Tarea 3.1										

Dentro de este tipo de diagramas, se pueden incluir **fases** que engloben diferentes tareas. La **duración** de cada fase es la necesaria para concluir esas tareas. Estos diagramas se pueden utilizar para estimar los recursos y el presupuesto en función del tiempo, totalizando los costes de las diferentes actividades del proyecto.

	UNIDADES DE TIEMPO									
	1	2	3	4	5	6	7	8	9	10
FASE 1										
Tarea 1.1										
Tarea 1.2										
Tarea 1.3										
FASE 2										
Tarea 2.1										
Tarea 2.2										
Tarea 2.3										
FASE 3										
Tarea 3.1										

En la siguiente imagen puedes ver un ejemplo **de Diagrama de Gantt** para la planificación de un curso de ocho sesiones, con un proyector multimedia como recurso. Está desarrollado con Planner bajo el sistema operativo Guadalinex.



Para saber más

Artículo muy interesante para la ampliación de los diagramas de Gantt, en el que se pueden encontrar también algunos ejemplos de redes de precedencia.

Diagramas de Gantt **[Versión en caché]**

Técnicas. Redes de Precedencia, Redes Pert

__Julia__ hizo un curso de investigación operativa en el que aprendió a usar __redes PERT__ para minimizar los costes en determinados problemas. Lo que aprendió le resultó muy interesante, pero nunca había puesto en práctica sus conocimientos. __María__ le explica que justamente es una de las herramientas que se usa para __planificar los proyectos__, y que su uso permite asignar los recursos de forma eficiente a las distintas tareas de que consta un proyecto, reduciendo los tiempos de desarrollo y detectando de forma preventiva las tareas críticas que harán acumular retrasos en todo el proyecto si no se terminan según lo planificado.

También permiten detectar __otras tareas__ para las que el tiempo planificado admite ciertos márgenes de demora. Esto nos permitirá por un lado evitar retrasos dedicando los recursos necesarios a las tareas críticas, y por otro lado, reasignar recursos desde las tareas que tienen "holgura" hacia las tareas críticas, para cumplir con todos los plazos, o incluso para reducir el tiempo total de desarrollo del proyecto.

__Julia__ lo entiende perfectamente, y está dispuesta a ser ella la que se encargue de elaborar el diagrama PERT del proyecto, que tan útil les resultará a la hora de hacer una buena planificación de su proyecto. De todas formas, __María__ que ya tiene más experiencia

*elaborando y aplicando estas técnicas a proyectos informáticos, le ayudará en su elaboración, e incluso **José** le hará una revisión al final para comprobar que todo está en orden.*

Introducción a las redes de precedencia

Como seguro que ya has notado, la siguiente técnica de la que nos vamos a beneficiar en la planificación de nuestros proyectos son las redes de precedencia, o redes PERT. ¿Crees que es una **técnica nueva** aplicable solo a proyectos informáticos? Nada de eso. Las redes PERT se idearon y usaron por primera vez para mejorar la planificación del proyecto "Polaris", en Estados Unidos. Su uso permitió reducir el tiempo total del proyecto en un año.

A finales de los años cincuenta aparecieron técnicas basadas en grafos para la planificación de proyectos, las redes de precedencia, que tratan la relación entre el coste y la duración de las actividades. Es lo que se conoce como **planificación de proyectos con coste mínimo**.

Estas técnicas son adecuadas cuando se cumplen las siguientes afirmaciones en el proyecto:

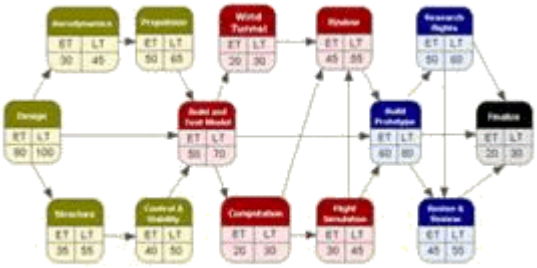
- Tiene todas sus actividades bien **definidas**.
- Las **actividades** se pueden comenzar, interrumpir y realizar de forma separada dentro de una secuencia dada.
- Pueden existir **relaciones** entre actividades.
- Las actividades están bien ordenadas, de modo que se puede seguir una **secuencia**.
- Una vez comenzada una actividad, debe continuar sin interrupción hasta su conclusión.

Con estas **redes** se pretende establecer la **secuencia** de sucesos clave en un proyecto, permitiendo de este modo mostrar el camino crítico, que será la base de la planificación y el control del proyecto.

Para disminuir el **tiempo total**, hay que reducir los tiempos de las actividades incluidas en el camino crítico, y eso, habitualmente supone un aumento de costes. Lo normal es que los ajustes de tiempos aparezcan a medida que se avanza y eso puede llevar a retrasos inevitables, lo que significa que el **camino crítico** previsto inicialmente no determina la duración real del proyecto y por tanto pueden aparecer nuevos caminos críticos. El jefe del proyecto debe supervisar continuamente aquellas actividades susceptibles de provocar retrasos y valorar el impacto de los posibles cambios, para evitar la aparición de nuevos caminos críticos y por consiguiente los retrasos en el proyecto.

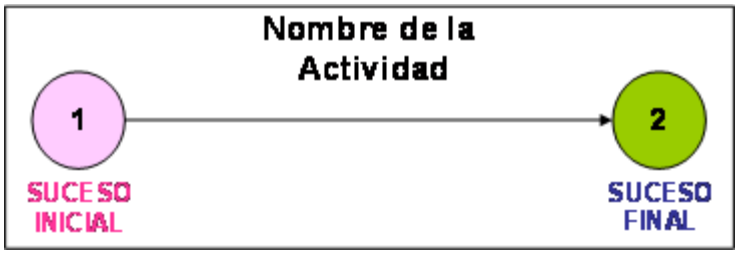
También se representan con las **redes de precedencia**, las actividades que no son críticas, lo que permite tener una visión global del control del proyecto y facilitar al jefe del mismo la reordenación de tareas o reasignación de los recursos disponibles.

Representación de diagramas PERT



La técnica PERT parte de la descomposición del proyecto en actividades. Las actividades ocurren entre dos sucesos; inicial y final. La representación se realiza mediante un grafo en el que **las actividades se muestran con flechas** y **los sucesos con nodos**.

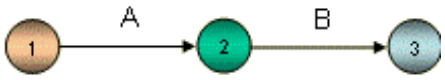
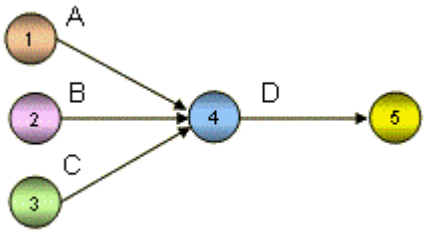
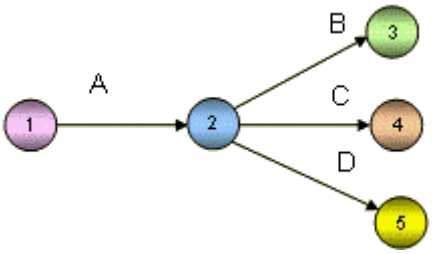
Importante: **La longitud de la flecha NO tiene nada que ver con la duración de la actividad.**



Para cada actividad es preciso estudiar las relaciones de precedencia, es decir, las actividades que deben estar finalizadas antes del comienzo de la actividad en cuestión.

Hay tres tipos de **relaciones de precedencia** entre actividades, que te mostramos en la siguiente tabla:

Tipos de relaciones de precedencia

Tipo de relación	Representación gráfica	Descripción
Lineales		Antes de iniciar la actividad B es preciso haber concluido la actividad A. En este caso el nodo 2 es suceso final de la actividad A y es suceso inicial de la actividad B.
Convergentes		Para iniciar la actividad D es preciso haber concluido las actividades A, B y C.
Divergentes		Para poder iniciar cualquiera de las actividades B, C o D, es preciso que haya terminado la actividad A.

Ejemplo de elaboración de un diagrama PERT

El movimiento se demuestra andando, así que veamos un **ejemplo** de representación de la precedencia de actividades con redes Pert, que te ayudará a entender todos los conceptos:

Vamos a realizar el análisis de un trabajo que se puede realizar en un taller de reparación de ordenadores. Las actividades que deben realizarse para la reparación del equipo en cuestión son las siguientes:

A. *Tomar datos del equipo y su propietario, para la recepción y elaborar nota para reparación. (tiempo estimado de duración, 1 hora)*
B. *Diagnosticar los problemas y elaborar presupuesto de reparación. (3 horas)*
C. *Pasar el presupuesto al cliente. El cliente confirma el presupuesto y damos la orden de reparación. (2 horas)*
D. *Cambio o reparación de los componentes defectuosos y prueba exhaustiva del equipo para comprobar su funcionamiento tras la reparación. (5 horas)*
E. *Confección de la factura o documento de reparación y entrega del equipo reparado al cliente. (2 horas)*
F. *Un equipo que está perfectamente identificado y en garantía, presenta problemas. (2 horas)*
G. *El cliente no confirma el presupuesto y se lleva el equipo sin reparar. (1 hora)*
H. *Los problemas diagnosticados no tienen reparación y el cliente se lo lleva sin coste. (1 hora)*

Para facilitar la comprensión de los ejemplos posteriores, evitamos las fracciones de hora en la duración de las actividades (normalmente se trata de actividades más complejas, y la duración de las tareas suele hacerse en número de días, aunque evidentemente depende de cada proyecto). Y lo resumimos en el siguiente cuadro:

Actividades	A	B	C	D	E	F	G	H
Duración (horas)	**1**	**3**	**2**	**5**	**2**	**2**	**1**	**1**

Las relaciones entre las actividades son las siguientes:

- A precede a B y H
- B precede a C y G
- D precede a E
- C y F preceden a D

Hay dos formas (igualmente efectivas) de representar este conjunto de relaciones:

- La matriz de encadenamientos.

		ACTIVIDADES PRECEDENTES j = A .. H							
		A	B	C	D	E	F	G	H
ACTIVIDADES SIGUIENTES	A								
	B	X							
i = A .. H	C		X						
	D			X			X		
	E				X				

F		
G		**X**
H	**X**	

- El cuadro de relaciones de precedencia.

Actividades del Proyecto	Actividades Precedentes
A	-
B	A
C	B
D	C y F
E	D
F	-
G	B
H	A

Con la ayuda de estas dos herramientas, estamos en disposición de construir el grafo representativo de las Redes Pert, en el que la finalización de cada actividad nos lleva a un estado del proyecto.

[D]

El diagrama nos indica el orden de precedencia de las diferentes actividades a realizar. Cualquier empleado del taller de reparación sólo tiene que seguir el diagrama para atender correctamente a un cliente.

A cada una de las actividades del proyecto debemos asignar un tiempo. Pert considera que la duración de cada actividad es un valor que sigue una distribución estadística. Para la **asignación de tiempo** a una determinada actividad, debemos considerar tres variables:

- **Estimación del tiempo pesimista (t_p).** Tiempo máximo en que podría finalizarse la actividad si suceden todas las previsiones más negativas.
- **Estimación del tiempo normal (t_n).** Tiempo más probable en que es posible finalizar la actividad, teniendo en cuenta que pueden aparecer algunos problemas, pero no todos.

- **Estimación del tiempo más optimista (t_o).** Tiempo mínimo en que se puede finalizar la actividad si no aparece ningún problema.

Con estas tres variables, en una red vamos a trabajar con el Tiempo PERT (T_{ij}), que podemos calcular siguiendo la fórmula siguiente:

[D]

Esta fórmula viene a considerar que la probabilidad de que se presenten los inconvenientes más negativos y la probabilidad de que no se presente ningún inconveniente son **cuatro veces menores** que la probabilidad de que haya algún problema, pero no todos al mismo tiempo.

El Tiempo PERT se calcula para cada una de las actividades, pero el grafo representa además diferentes estados, para los que es necesario calcular...

- **El tiempo Early** (**TE** o más temprano posible o ausencia de demora). Representa el tiempo mínimo que debe transcurrir desde el comienzo del proyecto para alcanzar un determinado estado o suceso.
- **El tiempo Last** (**TL** o más tardío posible o mayor demora posible). Representa el tiempo máximo que podemos tardar en alcanzar un determinado estado o suceso de forma que el proyecto no se retrase.

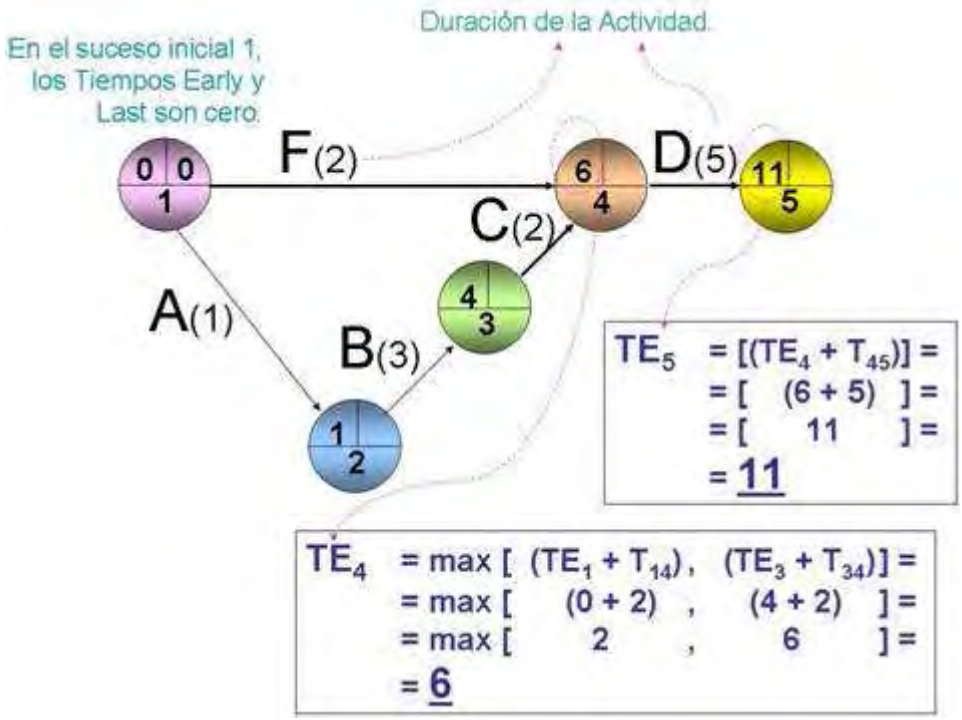
Cálculo de los Tiempos Early

Como ves, esta técnica no es sólo una representación gráfica, sino que tiene también toda una **teoría**, con definiciones que hay que asimilar, y **métodos** que hay que aprender a realizar para calcular algunos valores. Vamos a empezar viendo cómo se **calculan los tiemposEarly** de cada uno de los sucesos de nuestro proyecto.

El Tiempo Early del suceso j (TE_j) se calcula como el valor máximo que se obtiene al sumar el Tiempo Early de los sucesos iniciales de todas las actividades que finalizan en el suceso j y la duración de cada una de estas actividades.

Hay que tener en cuenta que el primer tiempo (**TE_1**) es cero, porque la primera actividad se puede iniciar nada más comenzar el proyecto. Veamos cómo se calcula el Tiempo Early de algunos sucesos del ejemplo anterior, según la tabla de duración de actividades obtenida (en ella las actividades E, F y H no aparecen en negrita para indicar que no tienen actividades que les sigan, y por tanto todas ellas terminan en un nodo final de proyecto):

Actividades	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*
Duración	*1*	*3*	*2*	*5*	*2*	*2*	*1*	*1*



Cálculo de los Tiempos Last

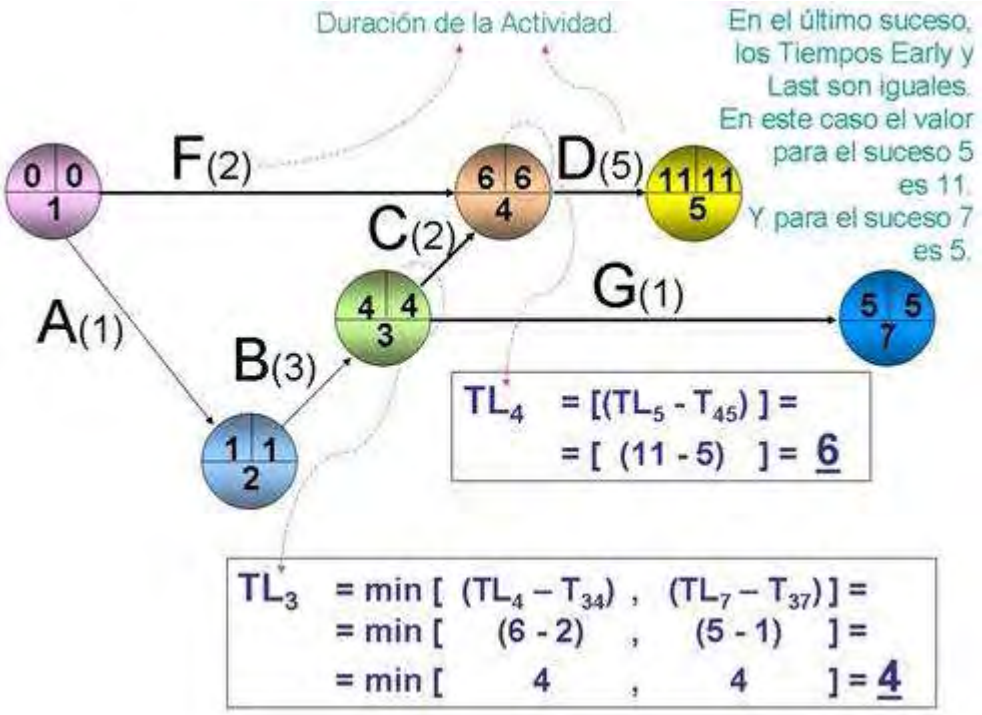
Y evidentemente, como supongo que esperarías, ahora le llega el turno al cálculo de los tiempos Last.

Hay que tener en cuenta que el Tiempo Last del último suceso, coincide con su Tiempo Early.

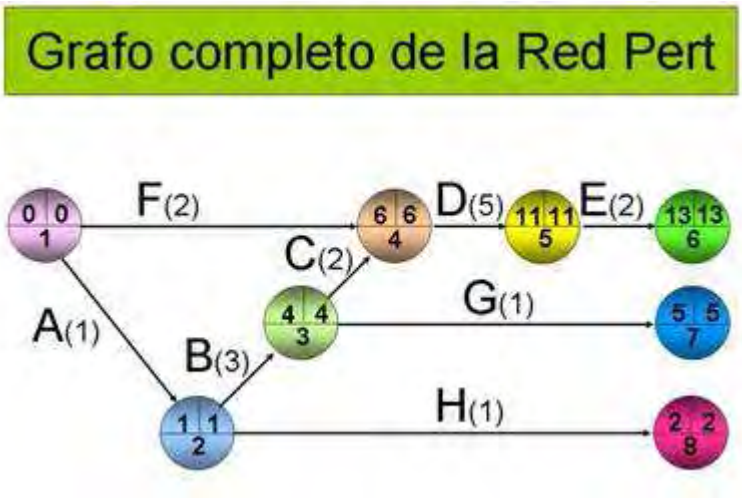
Se trata de establecer que si lo más pronto posible que se puede llegar a alcanzar el último suceso es su tiempo Early, también debe ser igual su tiempo Last, puesto que no estamos dispuestos a retrasar el proyecto. Es decir, si podemos terminar nuestro proyecto como muy pronto en 100 días, por ejemplo, es normal que queramos terminarlo en esos 100 días, ya que es

posible. Por tanto, consideraremos que el final del proyecto, como muy tarde, debe alcanzarse a los 100 días.

Con los **tiempos Last**, lo que pretendemos determinar por tanto es el **tiempo máximo** que podemos tardar en alcanzar ese suceso sin que se retrase el resto del proyecto. Teniendo eso en cuenta, para un suceso i el Tiempo Last se calcula como la menor diferencia entre los Tiempos Last de todos los sucesos en los que terminan las actividades que parten del suceso i, y la duración de las actividades en cuestión.



De este modo podemos concretar que la representación de nuestro ejemplo mediante el grafo de Redes Pert queda como sigue:



Haz clic sobre la siguiente imagen para ver una demo de cómo se representaría mediante Microsoft Project.

Holguras de sucesos y actividades: camino crítico

¿Y para qué sirve tanto cálculo de tiempos Last y tiempos Early?

Básicamente nos permite **calcular la holgura de cada suceso** y actividad, que a su vez nos permite detectar actividades que admiten cierto retraso sin poner en peligro los plazos de ejecución, y aquellas otras actividades "críticas" que de retrasarse, retrasarán inevitablemente todo el proyecto.

Seguro que ves claramente lo que eso significa: Con esa información, podremos distribuir los recursos asignados a cada actividad de la manera más eficiente, para evitar incumplir los plazos, o incluso para acortarlos. ¡El tiempo es oro!

El concepto de **Holgura de un suceso**, indica el número de unidades de tiempo que éste se puede retrasar sin que afecte la duración total del proyecto. Y se calcula como la diferencia entre el Tiempo Last y el Tiempo Early. De este modo la **Holgura del suceso i** se calcula como:

$$H_i = TL_i - TE_i$$

Por **ejemplo** para el suceso 4 del grafo anterior, sería...

$$H_4 = TL_4 - TE_4 = 6 - 6 = 0$$

En este grafo ninguno de los sucesos presenta **Holgura**, lo que significa que tenemos que alcanzar todos los sucesos en el tiempo mínimo que es posible hacerlo, si no queremos que se vea **retrasado** el proyecto. No obstante, en muchos casos, sí que es posible encontrar holguras distintas de cero para algunos sucesos del grafo, lo que significaría que es posible alcanzarlos más tarde del tiempo mínimo establecido, y que aún así no se vea retrasado el resto del proyecto.

Asociado a este concepto aparece también el de **Holgura Total de una Actividad**, como el **número de unidades de tiempo que puede retrasarse su realización** sin que aumente la duración total del proyecto. Y se calcula como la diferencia entre el Tiempo Last del suceso final de la actividad, menos el Tiempo Early del suceso inicial, menos la duración de la actividad. Así para la actividad que comienza en el suceso i para finalizar en el suceso j:

$$H^T_{ij} = TL_j - TE_i - T_{ij}$$

Por **ejemplo** para la actividad F del grafo anterior, sería...

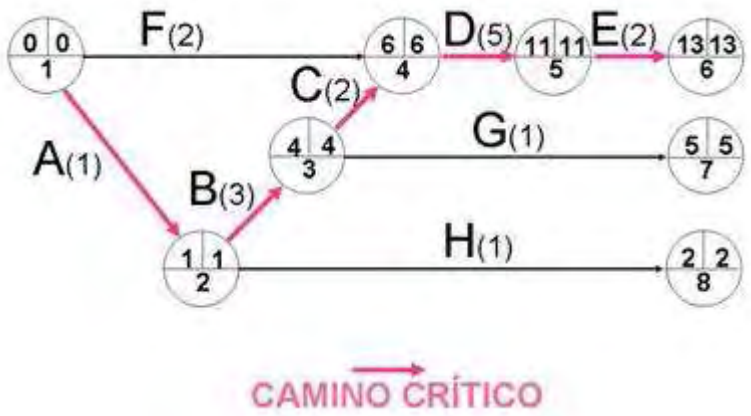
$$H^T_{14} = TL_4 - TE_1 - T_{14} = 6 - 0 - 2 = 4$$

Y para la actividad G, sería...

$$H^T_{37} = TL_7 - TE_3 - T_{37} = 5 - 4 - 1 = 0$$

Las actividades que tienen una Holgura Total igual a cero se denominan **Actividades Críticas**. Uniendo todas las actividades críticas desde el suceso inicial hasta el suceso final del proyecto, obtenemos un camino, que recibe el nombre de **camino crítico**.

Cualquier retraso en alguna actividad del camino crítico conlleva un inevitable retraso similar del proyecto.



Por último podemos definir la **Holgura Libre de la actividad ij**, que representa la parte de la Holgura Total que puede consumirse sin afectar a las siguientes actividades, como la diferencia entre el tiempo Early del suceso final menos el tiempo Early del suceso inicial menos el tiempo de la actividad:

$$H^L_{ij} = TE_j - TE_i - T_{ij}$$

Por **ejemplo** para la actividad F del grafo anterior, sería...

$$H^L_{14} = TE_4 - TE_1 - T_{14} = 6 - 0 - 2 = 4$$

Y para la actividad G...

$$H^L_{37} = TE_7 - TE_3 - T_{37} = 5 - 4 - 1 = 0$$

Hay que hacer notar que algunos autores cambian estos nombres por **Margen de Demora Total** (Holgura Total) y **Demora Permisible** (Holgura Libre) y así podemos encontrarlos en algunas aplicaciones destinadas a la gestión de proyectos.

*Y como **resumen** final, piensa lo que nos permite toda esta teoría:*

- *Determinar las actividades críticas, y el camino crítico, y así poder...*
- *Destinar los recursos suficientes a esas actividades para que no se retrasen, y para ello,...*
- *Podremos reasignar recursos desde otras actividades cuya holgura libre nos de cierto margen de maniobra.*
- *Conseguiremos así hacer nuestro proyecto en un tiempo mínimo, e incluso...*
- *Podremos identificar las actividades sobre las que habrá que incidir si se quiere reducir el tiempo de ejecución del proyecto.*

Para saber más

Amplio artículo sobre las redes Pert que puede ser un magnífico complemento a los contenidos de este apartado.

Redes Pert **[Versión en caché]**

Estimación de recursos y costes

Cuando se trabaja para un cliente que ha contratado nuestros servicios, las principales preocupaciones del mismo sobre el producto son:

- *¿**cuánto** le va a costar?, y*
- *¿**cuándo** estará disponible?*

*Por ello, una vez realizada la planificación del proyecto, **María** y **Julia** se disponen a hacer una previsión de los plazos, así como de los recursos y costes.*

***María** ha tenido ya alguna experiencia en la **estimación de costes** en proyectos similares, por lo que orienta a **Julia** que se encuentra ante su primer proyecto. En primer lugar le explica que cualquier estimación debe basarse en determinados **modelos** y que desde luego no debe ser un esquema rígido al que ajustar el proyecto, más bien se trata de una guía dinámica que irá adaptándose y "aprendiendo" durante el desarrollo del proyecto. En primer lugar realizan una concreción de los recursos necesarios y disponibles, para después aplicar una de las técnicas más conocidas de estimación de costes, llamada método COCOMO, basada en modelos empíricos y con claras fórmulas matemáticas, que **Julia** no tarda en comprender.*

Las estimaciones suelen ser valoraciones del esfuerzo esperado para el desarrollo del proyecto y de los plazos de tiempo requeridos para completarlo, teniendo en cuenta cierto grado de error. Se asume que **los gastos del proyecto están dominados por los gastos del personal**, por lo que la principal unidad de medición suele ser **el número de salarios mensuales o anuales del equipo de desarrollo**, por lo que la unidad de estimación suele ser **personas-mes** o **personas-año**.

Para la realización de una buena **estimación** de los recursos y costes durante el desarrollo de un proyecto software es necesario **disponer** de...

- **Experiencia mínima en estimaciones**. Haber participado en algunas anteriormente.
- **Conocimientos sobre la complejidad y tamaño del proyecto**. Es imprescindible tener cierto conocimiento de estos aspectos del proyecto, ya que suelen ser determinantes a la hora de hacer estimaciones.
- **Informaciones históricas**. Especialmente de proyectos similares o de similares características.

La estimación se realiza sobre **recursos, costes y plazos de tiempo** necesarios, al principio del proyecto y deben ser actualizadas constantemente durante el desarrollo del mismo, indicando el grado de cumplimiento de cada uno de ellos y su repercusión en el resultado final.

Estimación de recursos

Cuando hablamos de los **recursos** necesarios para llevar a cabo el proyecto planteado, nos referimos tanto a recursos humanos, materiales e incluso a la información que se debe tratar o utilizar, por ello es habitual encontrar una clasificación de recursos según el siguiente esquema:

- **Herramientas hardware**. Es preciso indicar el número de unidades de cada dispositivo, para qué va a ser utilizado y detallar las características deseables de cada uno. En ocasiones se hace inevitable que varios proyectos compartan determinadas herramientas, por lo que es necesario hacer una distribución de uso.
- **Herramientas software**. Debe ser recogido el modo de uso de todo el software y bajo qué condiciones actuará en el proyecto, por ejemplo si se utilizará de forma individual o como recurso compartido en red, si tendrá diferentes niveles de acceso y las posibilidades de cada nivel, etc.
- **Personal**. Detalle de los diferentes puestos organizativos dentro del proyecto y las funciones o actividades asignadas a cada uno de ellos, estableciendo claramente los protocolos de actuación y seguridad durante todo el proyecto.

Como resumen de lo anterior, podemos concretar que cada recurso debe quedar especificado mediante cuatro características:

- **Descripción del recurso** (hardware o software) o **habilidad requerida** (del personal).
- **Disponibilidad**.
- **Fecha de comienzo**.
- **Duración del uso** (hardware o software.) o **duración de la tarea** (del personal).

Estimación de costes

Como ya has visto a lo largo de las unidades anteriores, incluso a lo largo de las unidades del módulo de PLE, si también lo has cursado, **el software es el elemento más caro de los sistemas informáticos.** Por eso, entenderás que pongamos tanto empeño en conseguir **reducir sus costes,** tanto de desarrollo como de mantenimiento. Por eso también ponemos especial empeño en estimar convenientemente esos costes al iniciar el proyecto.

¿Qué pasaría si abordamos un proyecto y el coste resulta inasumible para el cliente?

Seguramente habrás pensado, con razón, que habremos estado perdiendo el tiempo y trabajando inútilmente, porque el proyecto no podrá terminarse por falta de **financiación**. Y no sólo eso. Tenemos que estimar los **beneficios** que el proyecto aportará al cliente una vez que esté terminado. Si el cliente hace una inversión, es normal que quiera rentabilizarla, por lo que el resultado del proyecto debe suponer una mejora para el cliente, que debe ser también cuantificable.

Un **error** en la estimación de coste puede ser la diferencia entre los beneficios y las pérdidas. **Sobrepasarse en el coste puede ser desastroso para el equipo de desarrollo.**

A pesar de que estamos hablando del término "**estimación de coste**" para proyectos de software, los valores obtenidos no se miden directamente en unidades monetarias. Las estimaciones suelen ser valoraciones, con un cierto error (por ejemplo, se suele asumir un 20%) del esfuerzo esperado para el desarrollo del proyecto y de los plazos de tiempo requeridos para completarlo.

Ya que el software es un producto sin existencia física propia y cuyo coste principal reside en su **desarrollo o diseño** (no en su fabricación o replicación a partir de la primera copia), es lógico que se asuma que el coste de su producción está dominado por los **gastos de personal**, como hemos indicado anteriormente. Por eso, la principal unidad de medición de coste del proyecto (**esfuerzo de desarrollo**) suele ser el número de salarios mensuales o anuales que deben pagarse. El esfuerzo de desarrollo se mide por tanto en **personas-mes** o **personas-año**.

Para conseguir una buena previsión de costes, es necesario invertir en la implantación de

mecanismos para la recogida de datos que sirvan para mejorar las predicciones futuras así como en el desarrollo de métodos apropiados de estimación.

Métodos de estimación de costes

¿Es fácil hacer una buena estimación de costes? Si a veces nos cuesta trabajo cuadrar las cuentas a final de mes, y eso que suelen ser bastantes parecidas todos los meses, o al menos todos los años, imagina lo **complicado** que puede ser estimar los costes de los proyectos, donde cada uno es distinto del anterior, y donde la cantidad de variables a tener en cuenta es enorme.

- ¿Será el coste igual si intervienen programadores y analistas expertos que si son novatos?
- ¿Influirá el hecho de que el proyecto se haga con un nuevo lenguaje de programación?
- ¿Son importantes las herramientas de desarrollo disponibles?
- ¿El coste de proyectos similares en complejidad y tamaño, será similar?
- ¿Vale de algo la experiencia a la hora de hacer las estimaciones de costes?
- ¿Existen fórmulas que puedan aplicarse sin más para obtener el coste de un proyecto dado?

Poco a poco iremos despejando esas incógnitas.

Podemos decir que existen **cuatro tipos de métodos** para la estimación de costes de proyectos software:

1. **Opinión de expertos**. Consiste en un "**estimación basada en la experiencia**". Se suele emplear la opinión de más de un experto para obtener una mayor fiabilidad en la estimación. En algunos casos, simplemente se calcula la media de los valores ofrecidos por las distintas personas. Esto tiene el riesgo de que la estimación obtenida puede resentirse mucho a causa de uno o dos valores extremos. También se puede efectuar una **reunión** tan larga como sea necesaria hasta llegar a un consenso, pero se corre el riesgo de que las personas más influyentes, por su capacidad o por su poder de convencimiento, sean las que determinen el resultado final.
2. **Estimación por analogía**. Se compara el proyecto a estimar con otros. En función de las **similitudes y diferencias** con ellos se deduce el coste del nuevo proyecto. La principal ventaja de esta técnica es que está basada en la experiencia real (no sólo en la subjetiva) de los proyectos. El inconveniente esencial es que es difícil conocer realmente el grado de similitud del proyecto que se estima con el elegido.
3. **La descomposición**. Consiste en **descomponer el proyecto** en sus funciones principales y éstas a su vez hasta conseguir un nivel de detalle tal que se pueda estimar directamente cada una de sus unidades elementales. El **coste total** del proyecto será la suma de todas las estimaciones individuales. Por **ejemplo**, existen técnicas en las que tras la descomposición el responsable de cada componente del software que hay que construir estima el coste de desarrollo de ese componente. La estimación para el proyecto completo se calcula mediante la **suma de las cantidades parciales**. Para poder aplicar esta técnica con una mínima eficacia es preciso disponer de un diagrama de descomposición del producto (**WBS del producto**), que suele crearse en la planificación del proyecto y que representa la jerarquía del producto. La técnica suele complementarse con un **diagrama de composición** de actividades del proyecto (**WBS del trabajo**) que indica la jerarquía de tareas del proyecto. De esta manera se asegura que actividades como la integración de componentes o la gestión de la configuración queden reflejadas en la estimación del coste.
4. **Ecuaciones o modelos empíricos**. Son fórmulas matemáticas que relacionan diversos parámetros del proyecto con el coste o esfuerzo requerido. Como ejemplo de estos métodos existe el **modelo COCOMO** que veremos en el siguiente apartado, pero también SLIM y la estimación con puntos de función.

A modo de resumen, la presentación siguiente te ofrece un esquema e estos tipos de métodos de estimación de costes. Pulsa sobre la imagen para activarla.

Existen **herramientas automatizadas** de estimación tanto para los modelos de descomposición como para los empíricos.

Modelo COCOMO

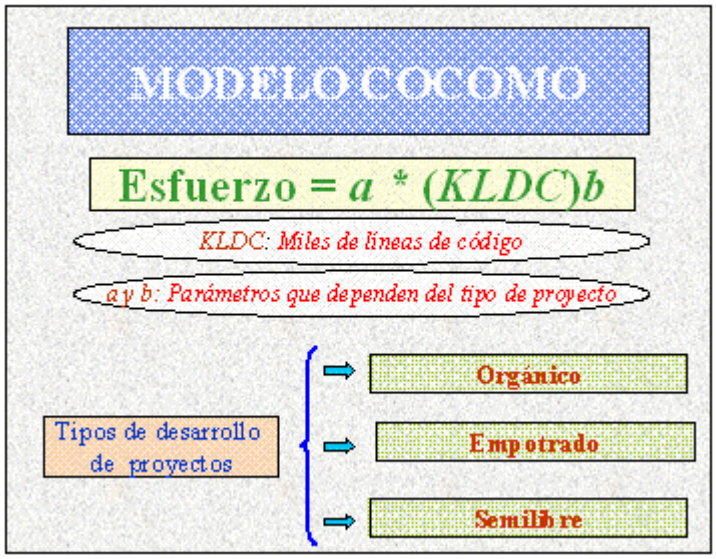
Seguro que te ha sonado bien la idea que planteábamos de conseguir **una fórmula** o ecuación que pudiéramos aplicar sin más para calcular el **coste de un proyecto**. De hecho, suena tan bien, que ha habido investigadores que la han desarrollado.

Pero antes de que empieces a lanzar las campanas al vuelo, no olvides que seguimos dentro del campo de la "**estimación**" de costes, y en esa fórmula, no todas las incógnitas tienen valores claros y conocidos.

El **modelo COCOMO** es de este tipo, pero la primera variable que destaca en la fórmula es el tamaño en miles de líneas de código de nuestro proyecto. ¿Cómo lo vamos a saber, si precisamente estamos viendo si merece la pena programarlo y todavía no hemos escrito una sola línea de código? Probablemente, una vez más tendremos que recurrir a los **expertos** en proyectos similares para que esta primera estimación sea lo más precisa posible

El modelo COCOMO (acrónimo de COnstructive COst MOdel) es seguramente el **modelo más conocido** de estimación de esfuerzo. Se apoya en una estimación previa del tamaño del software en LDC o también en KLDC. Este dato sirve como parámetro de las ecuaciones de cálculo de cuya forma general es la siguiente:

$$Esfuerzo = a * (KLDC)^b$$



El **esfuerzo** se mide en personas-mes y en la ecuación, *KLDC* es el tamaño en miles de líneas de código, mientras que *a* y *b* son parámetros de ajuste según el tipo de desarrollo del proyecto, que puede ser:

- **Orgánico.** Desarrollo en un entorno estable, con poca innovación técnica, con pocas presiones de tiempo y tamaño relativamente pequeño (< 50 KLDC).

- **Empotrado.** Desarrollo de software con requisitos muy restrictivos, con requisitos poco claros, complejo y en un entorno con gran innovación técnica.
- **Semi-libre.** Situaciones entre el modo orgánico y el empotrado.

Las **ecuaciones**, tanto para el cálculo de esfuerzo como de tiempo de desarrollo para cada uno de los modos, se ofrecen en la tabla siguiente:

Modo de desarrollo	a	b	Personas-mes (nominal)	Tiempo de desarrollo (nominal)
Orgánico	3,2	1,05	$PM = 3.2 \times KLDC^{1.05}$	$TD = 2.5 \times PM^{0.38}$
Semi-libre	3,0	1,12	$PM = 3.0 \times KLDC^{1.12}$	$TD = 2.5 \times PM^{0.35}$
Empotrado	2,8	1,20	$PM = 2.8 \times KLDC^{1.20}$	$TD = 2.5 \times PM^{0.32}$

A partir de estas ecuaciones básicas, COCOMO distingue **tres modelos** distintos que se corresponden con las diferentes cantidades de información disponible en las distintas etapas del ciclo de vida:

- **COCOMO básico,** para estimaciones **iniciales** moderadamente precisas al inicio del proyecto cuando no se dispone de detalles (por ejemplo, para empezar a negociar el contrato). Consiste en aplicar la ecuación básica antes presentada.
- **COCOMO intermedio,** cuando tenemos identificados los **principales componentes** del sistema (por ejemplo, cuando se dispone de una especificación de requisitos más o menos terminada). Se emplea para estimar el coste de dichos componentes y consiste, primeramente en aplicar la **ecuación básica** para el esfuerzo o el tiempo de desarrollo denominado nominal, ya que no está adaptado a las características del entorno de desarrollo. Después, este esfuerzo nominal se ajusta incorporando la influencia de 15 factores de coste:
  1. Fiabilidad requerida
  2. Tamaño de la Base de Datos
  3. Complejidad del software
  4. Restricciones de tiempo de ejecución
  5. Restricciones de memoria
  6. Volatidad del hardware
  7. Restricciones de tiempo de respuesta
  8. Calidad de los analistas
  9. Experiencia con el tipo de aplicación
  10. Experiencia con el hardware
  11. Experiencia con el lenguaje de programación
  12. Calidad de los programadores
  13. Técnicas modernas de programación
  14. Empleo de herramientas
  15. Restricciones a la duración del proyecto

A cada uno de esos factores se le pueden asignar valores, a elegir entre 6 posibles, que son Muy Bajo, Bajo, Medio, Alto, Muy Alto y Extra Alto, según la tabla elaborada por Boehm en 1981 y que se expone más abajo.

- **COCOMO detallado,** cuando están identificados los componentes individuales del sistema (por **ejemplo**, cuando se dispone de una especificación de requisitos totalmente acabada o cuando el diseño general está bien definido). En este caso, el modelo **COCOMO** proporciona **tablas** para poder distribuir las cantidades, ajustadas al entorno, del esfuerzo y del tiempo de desarrollo del proyecto, a lo largo de las distintas fases del mismo. Incluso se permite **refinar el**

ajuste de los factores para adaptarlo a las peculiaridades de cada etapa del proyecto. COCOMO permite estimar también el coste del mantenimiento del software.

Para la aplicación del modelo COCOMO se necesita

- conocer el tamaño del software que se va a construir en KLDC y
- debemos valorar cada uno de los **factores de coste** sobre una escala ordinal de seis niveles que va desde "muy bajo" hasta "extra alto" (en importancia de valor).
- Cuando un factor se valora como **nominal** o medio, su valor asignado es siempre 1, con lo que no influye en el coste.

Boehm elaboró una serie de criterios para poder valorar el nivel de cada uno de los factores (por **ejemplo**, hay un factor que se refiere a la experiencia en el lenguaje de programación donde los niveles abarcan desde el mes de experiencia en el más bajo a más de tres años en el más alto). Además, en el modelo COCOMO se asume que cada uno de los factores es independiente de los demás (no influyen unos en otros), aunque esto puede ser discutible.

TABLA DE FACTORES (O CRITERIOS) DE COSTE ELABORADA POR BOEHM EN 1981

Tipo de factor	Factores de coste (cost drivers)	Valores de los factores					
		Muy Bajo	Bajo	Medio	Alto	Muy Alto	Extra Alto
PRODUCTO	1. Fiabilidad requerida	0.75	0.88	1.00	1.15	1.40	------
	2. Tamaño de la Base de Datos	------	0.94	1.00	1.08	1.16	------
	3. Complejidad del software	0.70	0.85	1.00	1.15	1.30	1.65
HARDWARE	4. Restricciones de tiempo de ejecución	------	------	1.00	1.11	1.30	1.66
	5. Restricciones de memoria	------	------	1.00	1.06	1.21	1.56
	6. Volatidad del hardware	------	0.87	1.00	1.15	1.30	------
	7. Restricciones de tiempo de respuesta	------	0.87	1.00	1.07	1.15	------
PERSONAL	8. Calidad de los analistas	1.46	1.19	1.00	0.86	0.71	------
	9. Experiencia con el tipo de aplicación	1.29	1.13	1.00	0.91	0.82	------
	10. Experiencia con el hardware	1.21	1.10	1.00	0.90	------	------
	11. Experiencia con el leng. programación	1.14	1.07	1.00	0.95	------	------
	12. Calidad de los programadores	1.42	1.17	1.00	0.86	0.70	------
PROYECTO	13. Técnicas modernas de programación	1.24	1.10	1.00	0.91	0.82	------
	14. Empleo de herramientas	1.24	1.10	1.00	0.91	0.83	------
	15. Restricciones a la						

duración del proyecto 1.23 1.08 1.00 1.04 1.10 ------

Veamos a continuación un ejemplo del uso del modelo COCOMO para hacer una estimación del coste de un proyecto, utilizando los valores de la tabla.

Aquí tienes un ejemplo del modelo cocomo

Para saber más

Monográfico sobre el modelo COCOMO de estimación de costes, describe los diferentes modelos y obtiene conclusiones que ayudan a entender el modelo.

El Modelo COCOMO **[Versión en caché]**

Otra Web del modelo COCOMO en la que podemos apreciar otra visión de este modelo de estimación.

Análisis de Sistemas

El modelo de estimación COCOMO presenta muchas variantes según su ámbito de aplicación, el siguiente enlace puede ser una magnífica piedra de toque para conocer las características del COCOMO II.

COCOMO II **[Versión en caché]**

En el siguiente enlace puedes encontrar un PDF que describe el uso de una de las herramientas de trabajo (CocomoTool) para el modelo COCOMO.

COCOMOTool **[Versión en caché]**

Críticas a estos modelos de estimación de costes

Naturalmente ningún modelo de estimación de costes es **perfecto**, pero al menos nos proporcionan métodos para obtener una primera aproximación, y unos márgenes de error, que nos den ciertas garantías.

Sin embargo, ¿Cuáles son las principales críticas que podemos hacer de estos modelos? ¿En qué fallan?

- Los modelos que usan el número de LDC tienen el inconveniente de que hay que calcular de alguna manera este parámetro, y esto no es siempre sencillo.
- Todos los modelos han surgido de datos estadísticos de proyectos. El problema es que la cantidad y la representatividad de los proyectos no es tan amplia como sería deseable.
- Los modelos pierden precisión si se aplican en entornos distintos a los que se crearon. Hay que adaptarlos constantemente al entorno.
- Los errores en las estimaciones son de (25%.

Seguimiento y supervisión del proyecto software

*El proyecto **SINUNEURO** está en marcha y lo único que les falta a **María** y **Julia** es establecer los medios para el **seguimiento** y control del propio proyecto, con el fin de evitar*

obtener un producto final diferente del planteado inicialmente, y que por tanto sería inservible para la empresa.

Estos medios de seguimiento incluirán la previsión de situaciones en las que no se correspondan los resultados obtenidos con los esperados (especialmente cuando se trata de compromisos adquiridos o hitos no alcanzados), y las necesarias medidas correctoras de esas situaciones.

***María** explica que aunque contemplen algunas de estas situaciones es muy probable que a lo largo del desarrollo del proyecto, aparezcan algunas que habrá que ir solucionando sobre la marcha, por inesperadas y novedosas.*

El seguimiento y la supervisión del proyecto software implica seguir, revisar y comparar los logros y los resultados obtenidos frente a las estimaciones, los compromisos y los planes del proyecto, actualizándolos en función de estos resultados obtenidos en cada momento.

Además, el seguimiento y la supervisión del proyecto software puede detectar, a veces, que no se sigue el plan. Esto es muy útil, ya que el reconocimiento temprano de los problemas es el primer paso para resolverlos con garantía de éxito.

Por lo tanto, se puede decir que los **objetivos** que pretende conseguir el seguimiento y la supervisión del proyecto software, llevados a cabo por el **jefe del proyecto**, son:

1. Comparar los resultados actuales con los planes previstos (supervisión del progreso del proyecto).
2. Tomar acciones correctivas cuando existan desviaciones significativas de los planes previstos.
3. Acordar compromisos con el personal afectado por las acciones correctivas.

Supervisión de los resultados

¿Cuáles son las **actividades** que deberemos desarrollar para supervisar los resultados de nuestros proyectos? ¿Qué **criterios** deberemos adoptar para no separarnos demasiado del plan previsto, y para facilitar la comprobación de posibles desviaciones respecto a ese plan?

Para llevar a cabo la supervisión de los resultados de un proyecto, es preciso desarrollar una serie de **actividades**, tales como las siguientes:

- Establecer las **condiciones o medidas** que deben cumplirse cuando se realicen correctamente las diferentes tareas del proyecto. Esto incluirá documentos, terminología, comportamiento del software y cualquier otra cosa que indique a los miembros del equipo de desarrollo, los resultados esperados tras la correcta realización de cada tarea, de modo que si los resultados no se ajustan, no podemos decir que dicha tarea ha sido realizada correctamente.
- Establecer **sistemas de supervisión** y de informes, para lo cual hay que determinar: los datos necesarios, quién los recibe y cuándo se reciben.
- Medir los **resultados**, lo que permite determinar la consecución o la desviación de los objetivos y previsiones.

Teniendo en cuenta estas actividades, algunos de los **criterios necesarios para controlar los proyectos de acuerdo al plan** van a ser:

- **Planificación detallada del proyecto**. Desarrollar un plan detallado del proyecto implicando a todo el personal clave, definiendo el trabajo específico a realizar, el tiempo, los recursos y las responsabilidades.
- **Descomponer el proyecto global en actividades y tareas**. Utilizar la WBS como técnica de descomposición.
- **Resultados y entregables**. Definir los objetivos y los requisitos del proyecto en términos de especificaciones, planificación, recursos y productos entregables del proyecto.
- **Fijar Hitos**. Definir los hitos y los puntos de verificación del proyecto. Además, se pueden definir los resultados, los entregables y las medidas técnicas específicas frente al calendario y al presupuesto.
- **Establecimiento de Compromisos**. Obtener el compromiso de todo el personal clave (equipo de proyecto y dirección) en cuanto al plan del proyecto, sus medidas y sus resultados.
- **Seguimiento del proyecto**. Definir e implementar un sistema de seguimiento del proyecto que capture y procese los datos del proyecto obtenidos en las revisiones y acciones de la dirección.
- **Medición**. Asegurar que se realizan medidas fiables de los datos del proyecto, especialmente del progreso técnico frente al calendario y al presupuesto.
- **Revisiones periódicas**. Revisar los proyectos de forma regular tanto a nivel de subsistemas como a nivel de proyecto global.

Acciones correctivas

Como bien sabrás por experiencia propia, el refrán que dice "El hombre propone y Dios dispone" es tan cierto como la vida misma. Seguro que tú mismo has trazado planes en más de una ocasión que has tenidos que **modificar por imprevistos**, o que sencillamente no has podido cumplir por que eran poco realistas, o porque las circunstancias que tuviste en cuenta se han modificado de forma inesperada, etc. Pero eso nos plantea una cuestión: ¿Qué **acciones correctivas** tenemos que tomar cuando detectamos que el plan no se está cumpliendo según lo previsto?

El **progreso** se determina, sobre todo, mediante la comparación de los planes previstos, con el tamaño del software, con el esfuerzo, con el coste y con el tiempo empleado en finalizar los diferentes productos.

¿Y cuando debemos hacer esas comparaciones?

Parece razonable que deberán hacerse **cada vez que se alcance uno de los hitos marcados en el proyecto**.

Cuando no se cumplen los planes, se ejecutan diversas **acciones correctivas** que pueden incluir desde la revisión del plan de desarrollo del software para reflejar los logros reales hasta la replanificación de todo el trabajo.

El **objetivo** para el seguimiento del estado y del progreso, es que los problemas puedan ser detectados y resueltos con prontitud, cuando los resultados reales se desvían significativamente de los planes.

En general, las **acciones correctivas** pueden ser:

- **Añadir o reasignar personal para mejorar la eficiencia.**
- **Ajustar el número de horas extra.**
- **Reducir el alcance o el contenido de una entrega.**

- **Alargar o retrasar el calendario.**

En algunos casos, el problema es bastante **pequeño**, la acción correctiva está en manos de la dirección y no se requiere la revisión o la aprobación del cliente para efectuar cambios. En otros casos, la solución del problema puede ser bastante complicada y, por ello, hay que buscar la aprobación del cliente, puesto que puede significar cambios significativos de las condiciones pactadas (coste y plazos de ejecución).

Un aspecto clave del software que resulta tan frustrante para los directores es que es intangible. Su evolución sólo puede verse a través de mediciones significativas.

La necesidad desesperada de algunos directores para ver el progreso del software, que se concreta en una fuerte presión para acabar la planificación y el diseño del software y para empezar inmediatamente con la codificación, puede ser perjudicial.