UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Planificación y seguimiento en proyectos de desarrollo y mantenimiento de software dirigido por la gestión de tiempos

Autor: María Isabel Marante Estellés

Director: Dr. Patricio Letelier Torres

Diciembre del 2009

Tesis presentada para cumplir con los requisitos finales para la obtención del título de Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información, de la Universidad Politécnica de Valencia, 2009.

Agradecimientos

Estos tres últimos años han sido para mi, los más importantes e intensos de mi trayectoría profesional. En este tiempo he tenido la enorme suerte de conocer y trabajar con personas que me han ayudado (de una forma u otra) en la consecución de la presente tesis de máster, y motivado a seguir adelante con mis objetivos. A todas ellas mil gracias, pero en especial quisiera hacer mención de agradecimiento a las siguientes personas.

En primer lugar quisiera mencionar a mi director de tesis Patricio Letelier, al cual admiro como persona y como profesional. Me gustaría agradecerle la gran oportunidad que me ha dado y la confianza que ha depositado en mí. Estoy francamente agradecida por todo lo que me ha enseñado en estos años, por sus sabios consejos, por su fuente de motivación, y sobretodo, por el tiempo y el esfuerzo que me ha dedicado, gracias Patricio.

También me gustaría agradecer a los socios de ADD Informática el apoyo que he recibido en el desarrollo de este trabajo, y por haber confiado en mí para llevarlo a cabo. Gracias.

A mis padres, mi hermano y familia les agradezco que me hayan ayudado, apoyado y motivado en los momentos más duros. Sobretodo a mis padres, gracias por haberme escuchado y aconsejado, por estar ahí simpre.

Y en último lugar y más importante,quisiera especialmente darle las gracias a mi pareja, compañero, y sobretodo mi mejor amigo, Juanvi. Él ha sido mi mayor apoyo en todo este tiempo, sabe cómo hacerme reir cuando más lo necesito. Muchas gracias por toda tu comprensión, por entenderme, por tus consejos, por animarme, por estar ahí, gracias.

Tabla de Contenidos

Capítulo 1. Introducción

1.1. Motivación

Un proyecto de desarrollo y/o mantenimiento de software conlleva todas las dificultades de un proyecto de ingeniería, pero además incluye los particulares retos que tiene la construcción de un producto software; complejidad de la implementación, requisitos volátiles, desafíos tecnológicos, desarrollo colaborativo, dificultad para asegurar la calidad, etc. Además, el mercado cada vez exige plazos de entrega más reducidos y presupuestos más ajustados. Las empresas de desarrollo de software buscan dar respuestas a estos retos mediante procesos que se centren en mejorar la productividad y calidad del desarrollo, de forma que ayuden a cumplir con sus compromisos en los plazos de entrega establecidos.

En la última década hemos visto un creciente interés por metodologías y estándares asociados al proceso de desarrollo de software. El modelo de referencia CMMI [10] y los estándares SPICE [25] e ISO/IEC 90003 [26] han logrado reconocimiento y atraído el interés de empresas desarrolladoras de software. Metodologías Ágiles (p.ej. XP [5] y Scrum [43]) y Tradicionales (p.ej. RUP [29] y Métrica 3 [31]) han animado una interesante discusión respecto de las estrategias para desarrollar software y su efectividad en diferentes contextos, confirmándose que cada proyecto requiere un proceso ajustado a sus necesidades, es decir, deben considerarse: composición del equipo, envergadura del proyecto, características del dominio de aplicación, tecnología utilizada, etc.

Los proyectos de desarrollo de software suelen enfrentarse a un ámbito de trabajo muy cambiante (especialmente en cuanto a las especificaciones del producto y las prioridades de las características solicitadas) e intensivo en comunicación (entre el equipo y con el cliente). Las técnicas y herramientas genéricas para seguimiento de proyectos resultan claramente ineficaces para enfrentar estos desafíos. Las metodologías ágiles destacan esta situación pero la resuelven de una manera excesivamente simplista, utilizando roles muy genéricos (reduciendo así la comunicación necesaria entre diferentes roles) y confiando en la habilidad de cada uno de los miembros del equipo para que, verbalmente y/o con soportes no informatizados [19], realicen el seguimiento continuo del proyecto.

TUNE-UP es una metodología nacida en el trabajo día a día en una PYME de desarrollo de software y con una vocación de mejora continua del proceso. En tres años de aplicación y evolución TUNE-UP ha conseguido una madurez suficiente para ser presentada como una alternativa interesante, al menos en contextos de desarrollo con equipos pequeños. TUNE-UP incorpora elementos de metodologías ágiles y también del ámbito más tradicional. Una de las características clave de TUNE-UP es la planificación y seguimiento del proyecto centrada en la gestión de tiempos. TUNE-UP se inspira en la esencia de PSP (Personal Software Process) [45], donde se destaca que la base del éxito radica en una disciplina de trabajo y productividad individual centrada en la gestión de los compromisos. TUNE-UP ayuda en cada momento del proyecto a responder a las siguientes preguntas: ¿conseguiré cumplir con los plazos de entrega de mis tareas? o ¿seremos capaces de cumplir con los plazos de entrega al cliente? Estas simples preguntas inquietan a cualquier gestor o participante de un proyecto de desarrollo de software. No contar con información que ofrezca una respuesta acertada y sobre todo oportuna conlleva en la mayoría de los casos graves complicaciones en el proyecto.

Este trabajo de tesis se ha realizado en el contexto de un convenio universidad-empresa durante 3 años. La PYME es una empresa de desarrollo de software que tiene un ERP perteneciente al sector socio-sanitario y cuenta con más de 700 clientes. La autora de la tesis ha participado en la mejora de la metodología mediante su trabajo de investigación relativa a la gestión de tiempos, y en la implementación de los módulos de la herramienta de apoyo a dicha metodología (TUNE-UP Process Tool) relacionados con la gestión del tiempo. Tanto la metodología como la herramienta se están desarrollando y utilizando en la PYME desde hace 3 años.

El objetivo de este trabajo es ilustrar cómo TUNE-UP aborda la gestión de tiempos contribuyendo a que todos los agentes conozcan el estado de su trabajo ayudándoles a priorizar y planificar sus tareas o compromisos, y en especial a que el jefe del proyecto tenga mayor control respecto de los plazos y compromisos en un proyecto de desarrollo o mantenimiento de software.

1.2. Organización de la memoria

A continuación se presenta la estructura de la tesis de máster:

El capítulo 1 incluye una introducción de la motivación de nuestro trabajo y las preguntas que queremos responder con nuestra propuesta. Esta tesis tiene como fin dar respuesta a ¿conseguiré cumplir con los plazos de entrega de mis tareas? o ¿seremos capaces de cumplir con los plazos de entrega al cliente?

En el capítulo 2 se ha realizado un estudio de los diferentes modelos, guías y estándares más reconocidos en la gestión de proyectos, en concreto, cómo abordan la gestión del tiempo.

El capítulo 3 resume el estudio realizado sobre un conjunto de herramientas de gestión de proyectos existentes en el mercado, clasificadas en: herramientas genéricas, herramientas CASE junto a metodologías tradicionales, y herramientas de gestión de proyectos para metodologías ágiles. El fin de este capítulo es conocer cómo abordan la gestión del tiempo cada conjunto de herramientas.

El capítulo 4 explica los aspectos básicos de la metodología TUNE-UP.

En el capítulo 5 se completa la metodología TUNE-UP con nuestra propuesta, detallando de forma precisa el control de tiempos, y la planificación y seguimiento de proyectos software centrados en la gestión del tiempo.

El capítulo 6 muestra la herramienta TUNE-UP Process Tool, la cual apoya todas las prácticas propuestas por la metodología TUNE-UP, en concreto, las relativas a la gestión del tiempo.

El capítulo 7 concluye este trabajo mediante un resumen comparativo de los estudios realizados, las conclusiones sacadas a partir de nuestra propuesta, los impedimentos con los que nos hemos encontrado, y el trabajo futuro que ha surgido a raiz de esta tesis de máster.

Capítulo 2. Propuestas para la planificación y seguimiento de proyectos

De acuerdo con Project Management Institute (PMI) [38], la gestión de proyectos consiste en aplicar conocimientos, habilidades, herramientas y técnicas a las actividades de los proyectos a fin de satisfacer las necesidades de las partes interesadas. Es el arte de dirigir y coordinar los recursos humanos y materiales en la vida de un proyecto para lograr los objetivos del proyecto dentro de los límites especificados. El objetivo de la gestión de proyectos software es mantener el proyecto dentro de los objetivos de costo, calidad y duración mediante la gestión de las interacciones entre la planificación, control, desarrollo técnico, aseguramiento de la calidad, etc. La relación entre estos objetivos se explica mediante la analogía del Triángulo de Hierro [3], el cual demuestra que en los proyectos software existen 3 variables directamente relacionadas (tiempo, costo y alcance), y otra que permanece estática (calidad). La variable tiempo indica el tiempo que tardará el proyecto en completarse, la variable coste hace referencia a los recursos humanos, materiales, etc. que se dedicarán al proyecto, y el alcance hace mención a las características o tareas que se van a realizar. Para mantener la calidad de cualquier proyecto software es necesario que estas tres variables sean cuantitativamente proporcionales, en otro caso, si por ejemplo aumenta el número de características a desarrollar, se deberán de incorporar más recursos y consecuentemente, el coste del proyecto aumentará. Estas tres variables deben estar presentes en tres de las funciones principales dentro de la gestión de proyectos software: estimación, planificación y seguimiento del estado del proyecto.

Berander et al. (2005) [6] muestra que existen 4 atributos orientados a la gestión de proyectos que pueden ser predecibles, medidos, evaluados y mejorados para evitar que los proyectos software se desvíen de los compromisos iniciales planificados. El atributo **tiempo** indica la cantidad de tiempo requerido para completar el trabajo, el cual se suele medir en semanas meses o años. Este atributo está directamente relacionado con el "time-to-market" y se considera el más importante dentro de la planificación de los proyectos. Estimar la cantidad de tiempo en completar un trabajo es beneficioso para poder comparar el tiempo estimado con respecto al actual, y decidir si nos desviamos con respecto a lo estimado. El **esfuerzo** representa la "mano de obra" necesaria para realizar un trabajo medido en horas, meses-hombre, o en términos monetarios. Estimar

y evaluar el esfuerzo que se invierte en un trabajo resulta productivo cara a determinar el esfuerzo en futuros trabajos. El atributo **contenido** representa el tamaño del contenido funcional desarrollado en un proyecto, medido en función del desempeño, disponibilidad, usabilidad, etc. Según Putnam y Myers (1992) [40], los jefes de proyecto tienden a utilizar uno de los siguientes enfoques para predecir el tamaño de un producto:

- Un enfoque conservativo donde los jefes de proyectos dan vagas declaraciones sobre el tiempo que llevará en completarlo.
- El jefe de proyecto presenta una estimación del plazo de fin del proyecto, que no sabe si se cumplirá.
- El jefe de proyecto subestima a propósito, para poder seguir adelante con el proyecto o ganar el contrato externo que la empresa necesita.

Obviamente, no se recomienda ninguno de estos enfoques. Sin embargo existen otras propuestas como Líneas de Código (LoC), Puntos de Función (FP) o COCOMO, cuya medición es mucho más específica que los enfoques anteriores. El atributo **productividad** hace referencia al conjunto de características desarrolladas a una cierta velocidad. Este atributo solo se puede estimar a partir de estimaciones anteriores de características similares a la del objeto en cuestión. Dadas unas características a desarrollar y una estimación de la productividad del equipo de desarrollo, podemos determinar el tiempo o esfuerzo que les llevará en completar las características, y tomar decisiones en las primeras fases del proceso de desarrollo.

El proceso de gestión del tiempo en proyectos describe cómo supervisar y controlar el tiempo dedicado a un proyecto. Para ello son necesarios los siguientes elementos:

- Estimación de costes y esfuerzo.
- Planificación temporal de proyectos software. La planificación establece las fases e iteraciones del proyecto, la duración de las tareas, la organización temporal de las tareas, las relaciones de dependencia entre ellas, etc.
- Organización de recursos. Los recursos se asignan a las tareas teniendo en cuenta sus disponibilidades dentro del proyecto y su carga de trabajo.
- Seguimiento y control de proyectos software. La información relativa a la planificación debe estar actualizada en todo momento. El seguimiento se realiza

comparando los valores reales del proyecto con los planificados. Normalmente, se utilizan métodos visuales que ayudan a identificar los posibles problemas en la planificación.

El modelo CMMI y el estándar ISO/IEC 90003, así como otros puntos de referencia para gestión de proyectos como el PMBOK (Project Management Book of Knowledge) y PSP (Personal Software Process) recomiendan prácticas muy generales para la gestión del tiempo. A continuación se detallan las recomendaciones que éstos ofrecen para dar soporte a la gestión del tiempo en proyectos software.

2.1. Capability Maturity Model Integration (CMMI)

Ahora más que nunca, las compañías quieren entregar mejores productos, más rápidos y más baratos. Las tecnologías del siglo XXI hacen que la construcción de los productos sea cada vez más compleja; normalmente los componentes de los productos provienen de diferentes colaboradores software que se tienen que integrar y mantener en el producto final. Las organizaciones deben de ser capaces de gestionar y controlar este desarrollo y mantenimiento complejo de los productos.

Los modelos de madurez de las capacidades (Capability maturity models, CMMs) se centran en mejorar los procesos de una organización. Contienen los elementos esenciales de los procesos efectivos para una o más disciplinas. Paulk et al. (1995) [35] del Software Engineering Institute (SEI) crearon el primer modelo de madurez de las capacidades diseñado para las empresas de software, “The Capability Maturity Model: Guidelines for Improving the Software Process”. CMM Integration se desarrolló para resolver el problema de la utilización de múltiples modelos CMM: The Capability Maturity Model for Software (SW-CMM), The Systems Engineering Capability Model (SECM), y The Integrated Product Development Capability Maturity Model (IPD-CMM). CMMI fue desarrollado por el SEI de la Universidad Carnegie Mellon, y publicado en su primera versión en Enero de 2002.

Actualmente, existen 4 disciplinas disponibles para la planificación de la mejora de procesos usando CMMI [9]:

- Ingeniería de Sistema: Cubre la construcción de un sistema con o sin software

- Ingeniería de Software: Cubre la construcción de soluciones software
- Integración de productos y procesos de desarrollo: Cubre la relación a largo plazo con el cliente
- Relación con proveedores: Cubre los procesos relacionados con la subcontratación de partes del sistema

Estas disciplinas se abordan por las áreas de procesos asociados a ellas, y por los componentes del modelo llamados amplificaciones de disciplina (pieza de información relevante para una disciplina en particular). Un área de proceso es un conjunto de buenas prácticas relacionadas en un área que, cuando se aplican en conjunto, cumplen una serie de metas consideradas importantes para lograr una mejora significativa en esa área. Las prácticas de cada una de las áreas están clasificadas y detalladas en componentes del modelo, y se pueden resumir para ilustrar sus relaciones tal y como se visualiza en la Figura 1 obtenida de [35]. A continuación se detalla el objetivo de cada uno de los elementos de la Figura 1, para que en secciones posteriores, podamos conocer los conceptos requeridos en las prácticas que tratemos.

La **declaración del propósito** (Purpose Statement) describe el objetivo del área de proceso. La sección de notas introductorias (Introductory Notes) del área de proceso describe los principales conceptos del área. El elemento de **áreas del proceso relacionadas** (Related Process Areas) lista las referencias de las áreas de procesos relacionadas y refleja las relaciones de alto nivel entre ellas. Los **objetivos específicos** (Specific Goals, SG) describen las características únicas que deben estar presentes para satisfacer el área de proceso. Los **objetivos genéricos** (Generic Goal, GG) se llaman "genéricos" porque la misma declaración de objetivo aparece en múltiples áreas de procesos. Estos objetivos deben estar presentes para institucionalizar los procesos que implementan un área de proceso. Una **práctica específica** (Specific Practice, SP) es una descripción de una actividad que se considera importante para alcanzar el objetivo específico asociado. Los **productos de trabajo típicos** (Typical Work Products) lista las salidas de una práctica específica. Una **subpráctica** (subpractices) es una descripción detallada que proporciona una guía para interpretar e implementar una práctica específica. De la misma forma que los objetivos genéricos, las **prácticas genéricas** (Generic Practices, GP) se llaman "genéricas" porque la misma práctica aparece en múltiples áreas de procesos. Una práctica genérica es una descripción de una actividad que se considera importante para alcanzar el objetivo genérico asociado. Una

elaboración de una práctica genérica (Generic Practice Elaborations) proporciona una guía de cómo debe aplicarse una práctica genérica en un área de proceso.

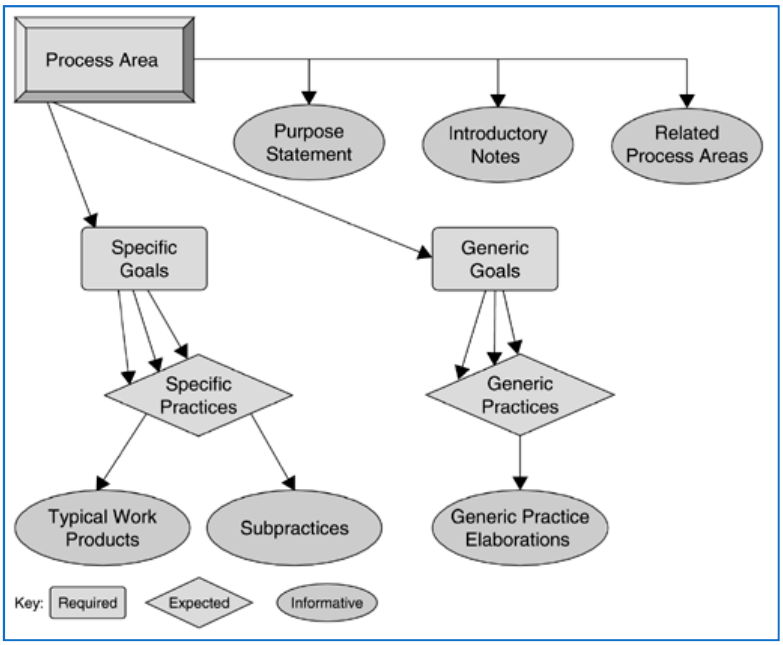


Figura 1 Componentes del modelo CMMI

Niveles, Capacidades y Resultados de CMMI

Los niveles son usados en CMMI para describir el camino evolutivo recomendado para una organización que quiere mejorar los procesos que utiliza para desarrollar y mantener sus productos y servicios. CMMI [8] propone dos caminos de mejora, la representación continua y por etapas. La representación continua permite a las organizaciones mejorar gradualmente los procesos correspondientes a un área de proceso individual (o áreas de proceso) seleccionada por la organización. Mientras que la representación por etapas (madurez) permite a las organizaciones mejorar un conjunto de procesos relacionados, abordando de forma incremental conjuntos sucesivos de áreas de procesos. La Figura 2 permite observar que la representación continua se centra en la capacidad de una área de proceso (medida por los niveles de capacidad de la Tabla 1), y la representación por etapas se centra en la madurez de la organización (medida por los niveles de madurez de la Tabla 2).

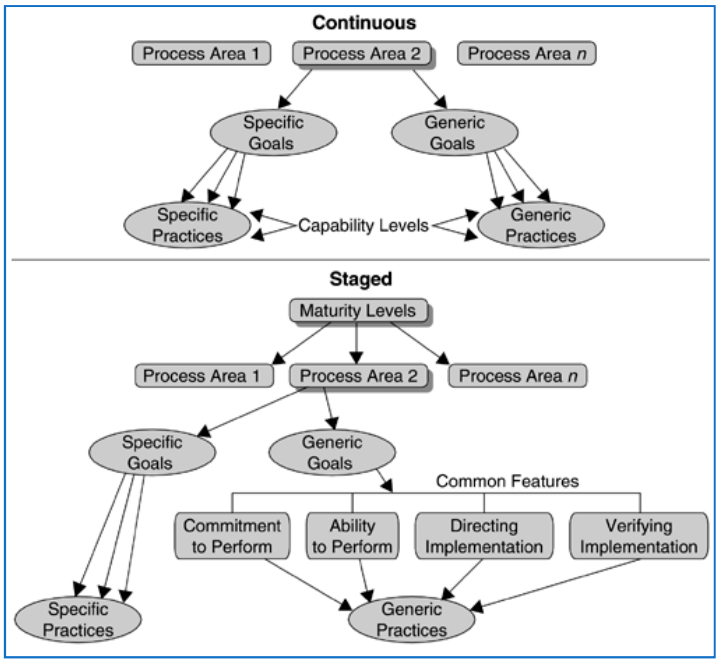


Figura 2 Estructura de la representación continua y por etapas

Niveles	Niveles de Capacidad	Propósito
Nivel 0	Incompleto	El proceso no se realiza, o no se consiguen sus objetivos.
Nivel 1	Ejecutado	El proceso se ejecuta y se logra su objetivo.
Nivel 2	Gestionado	El proceso se planifica, revisa y evalúa para comprobar que cumplen los requisitos.
Nivel 3	Definido	El proceso se ajusta a la política de procesos que existe en la organización, alineada con las directivas de la empresa.
Nivel 4	Cuantitativamente gestionado	El proceso se controla utilizando técnicas cuantitativas.
Nivel 5	En Optimización	El proceso se revisa y modifica sistemáticamente para adaptarlo a los objetivos del negocio. Mejora continua.

Tabla 1 Niveles de capacidad de la Representación Continua

Niveles	Nivel de Madurez	Propósito
Nivel 1	Inicial	El proceso de software es impredecible, sin control y reactivo. El éxito de los proyectos depende del talento de las personas involucradas.
Nivel 2	Gestionado	Existen procesos básicos de gestión en los proyectos (costo, calendario, funcionalidad, etc.). Los procesos existentes hacen que se puedan repetir éxitos en proyectos de similares características.

Nivel 3	Definido	Existe un proceso de software documentado y estandarizado dentro de la organización.
Nivel 4	Cuantitativamente gestionado	La organización recolecta métricas del proceso software y de los productos desarrollados. Tanto el proceso como los productos se entienden y controlan cuantitativamente.
Nivel 5	En Optimización	Existe una mejora continua del proceso software, basada en la realimentación cuantitativa del proceso y en la puesta en práctica de ideas y tecnologías innovadoras.

Tabla 2 Niveles de madurez de la Representación por etapas

Áreas de proceso de Gestión de Proyectos

Las áreas de proceso se pueden agrupar en cuatro categorías: Gestión de Procesos, Gestión de Proyectos, Ingeniería y Apoyo. En este trabajo vamos a centrarnos en la categoría de Gestión de Proyectos puesto que es el objeto de la tesina.

Las áreas de proceso de Gestión de Proyectos cubren las actividades de gestión de proyectos relacionadas con la planificación, seguimiento y control del proyecto. Las áreas de proceso de Gestión de Proyectos son las siguientes:

- **Planificación del Proyecto**: la planificación comienza con los requisitos que definen el producto y el proyecto. Esta planificación incluye la gestión de proyectos y actividades de ingeniería que se llevarán a cabo a través del proyecto. El proyecto examinará otras planificaciones que afecten al proyecto y establecerá los compromisos de acuerdo con los stakeholders relevantes por sus contribuciones en otros proyectos.
- **Seguimiento y Control del proyecto**: incluye actividades de seguimiento y de toma de medidas correctivas. La planificación del proyecto especifica el nivel apropiado de seguimiento del proyecto, la frecuencia de las revisiones del progreso, y las medidas usadas para seguir el progreso del proyecto. El progreso se determina comparando el estado del proyecto con la planificación. Cuando el estado actual se desvía significativamente de los valores esperados, se toman acciones correctivas.
- **Gestión de acuerdos con proveedores**: responde a la necesidad del proyecto para adquirir las porciones de trabajo que son producidos por los proveedores.

- **Gestión de proyectos integrada**: establece y mantiene el proceso definido del proyecto que es adaptado desde el conjunto de procesos estándares de la organización. El proyecto es gestionado mediante este proceso.
- **Gestión del riesgo**: tiene un continuo enfoque prospectivo para la gestión de riesgos con actividades que incluyen la identificación de los parámetros de riesgo, evaluaciones de riesgos y mitigación de riesgos.
- **Gestión cuantitativa del proyecto**: aplica técnicas cuantitativas y estadísticas para la gestión del rendimiento del proceso y calidad del producto. Los objetivos de calidad y de rendimiento del proceso se basan en los objetivos establecidos por la organización.
- **Control integrado de equipos**: proporcionar la formación y la logística a cada equipo integrado.
- **Gestión integrada de proveedores**: identifica las fuentes de los productos que pueden ser utilizados para satisfacer los requisitos del proyecto, y realiza un seguimiento de determinados proveedores de productos y procesos, manteniendo un proyecto cooperativo de relaciones con los proveedores.

Planificación, seguimiento y control de proyectos desde CMMI

La siguiente sección tiene como objetivo detallar las prácticas, técnicas y métodos propuestos por las diferentes áreas de Gestión de Proyectos, relacionadas con la planificación, seguimiento y control de los proyectos software. Las áreas involucradas con este fin son: la Planificación del Proyecto, y Seguimiento y Control del Proyecto. Además, existe un área de proceso incluida en la categoría de Apoyo llamada Medición y Análisis.

El objetivo de la Medición y el Análisis es desarrollar y sostener una capacidad de medición que sea usada para ayudar a las necesidades de información de la gerencia. Los datos tomados para la medición deben estar alineados con los objetivos de la empresa para proporcionar información útil a la misma. Se ha de implantar un mecanismo de recogida de datos, almacenamiento y análisis de los mismos de forma que las decisiones que se tomen puedan estar basadas en estos datos. Este sistema tiene que permitir además:

- Una planificación y estimación objetiva
- Comparar el rendimiento actual contra el rendimiento esperado en la planificación
- Identificar y resolver problemas relacionados con los procesos
- Proporcionar una base para añadir métricas en procesos futuros

Las áreas de procesos de gestión de proyectos fundamentales tratan las actividades relacionadas con el establecimiento y mantenimiento de la planificación del proyecto, el establecimiento y mantenimiento de los compromisos, seguimiento de los progresos contra la planificación, y adopción de medidas correctivas. A continuación, vamos a detallar estas actividades organizadas según el área de proceso a la que pertenecen, el objetivo específico que deben satisfacer, y las prácticas específicas relacionadas con la temática de la tesina.

En el área Planificación del Proyecto son fundamentales los siguientes objetivos específicos y prácticas específicas:

SG 1 Establecer Estimaciones

SP 1.1 Estimar el ámbito del proyecto: su objetivo es establecer una estructura de alto nivel de trabajo desglosado (Work Breakdown Structure, WBS) para estimar el alcance del proyecto. Un WBS divide el proyecto en un conjunto interconectado de componentes manejables, por ejemplo las unidades de trabajo o tareas. Para ello propone seguir el siguiente proceso:

1. Desarrollar un WBS basado en la arquitectura del producto. El WBS permite identificar los siguientes elementos:
    - Identificar riesgos y sus tareas de mitigación
    - Las tareas para ser entregadas y las actividades de apoyo
    - Las tareas para la adquisición de habilidades y conocimientos
    - Tareas para el desarrollo de planes de soporte necesarios, como la gestión de la configuración, aseguramiento de la calidad, etc.
    - Tareas para la integración y gestión de elementos que no son de desarrollo
2. Identificar los paquetes de trabajo en suficiente detalle para especificar las estimaciones de las tareas del proyecto, responsabilidades y programación.
3. Identificar las componentes del producto que serán externamente adquiridos.

4. Identificar los productos de trabajo que serán reutilizados.

SP 1.2 Establecer las estimaciones de los componentes del producto y los atributos de las tareas: su objetivo es establecer y mantener las estimaciones de los atributos de los componentes del producto y tareas. El tamaño es la entrada principal a muchos modelos utilizados para estimar el esfuerzo, costo y programación. Los modelos también pueden basarse en entradas como la conectividad, la complejidad y estructura. Para ello propone seguir el siguiente proceso:

1. Determinar el enfoque técnico para el proyecto, es decir, arquitectura, tecnologías, etc.
2. Usar métodos para determinar los atributos de las componentes de productos y tareas que serán usadas para estimar las necesidades de recursos. CMMI únicamente propone ejemplos de medidas de tamaño como el número de funciones, Puntos de Función, Líneas de Código, etc.
3. Estimar los atributos de los componentes de productos y tareas.
4. Estimar, según proceda, la mano de obra, maquinaria, materiales y métodos que serán requeridos por el proyecto.

SP 1.4 Determinar las estimaciones de esfuerzo y costes: su objetivo es estimar el esfuerzo y el coste del proyecto para los componentes del producto y las tareas basadas en estimaciones lógicas. Para ello propone seguir el siguiente proceso:

1. Coleccionar los modelos o datos históricos que serán usados para transformar los atributos de los componentes de producto y tareas, en estimaciones de horas de trabajo y coste.
2. Incluir el apoyo de necesidades de infraestructura al estimar el esfuerzo y coste. CMMI añade ejemplos de recursos críticos, como el espacio en disco o la capacidad de la red.
3. Estimar el esfuerzo y coste usando modelos y datos históricos. CMMI indica algunas entradas para realizar las estimaciones (juicio del experto, costes externos, etc.), pero no cómo hacerlas.

SG 2 Desarrollar un Plan de Proyecto

SP 2.1 Establecer el presupuesto y el calendario: el presupuesto del proyecto y el calendario se basan en las estimaciones elaboradas, y tienen como objetivo garantizar

que la asignación de presupuesto, la complejidad de las tareas y las dependencias de las tareas se abordan adecuadamente. Para ello propone seguir el siguiente proceso:

1. Identificar los principales hitos. Pueden basarse en eventos o en el calendario.
2. Identificar supuestos de la programación. Cuando se desarrolla una programación, normalmente se hacen supuestos sobre la duración de ciertas actividades.
3. Identificar restricciones mediante la evaluación de los atributos de los componentes de trabajo o tareas.
4. Identificar las dependencias entre las tareas para minimizar la duración de los proyectos. CMMI nombra técnicas utilizadas para determinar el orden entre las tareas: Critical Path Method (CPM), Program Evaluation and Review Technique (PERT) y Programación de Recursos Limitados.
5. Definir el presupuesto y la programación.

El establecimiento y mantenimiento del presupuesto y programación del proyecto, normalmente incluye lo siguiente:

- Definir la disponibilidad de los recursos comprometidos.
- Definir actividades con una duración adecuada.
- Identificar los hitos para la entrega del producto al cliente.
- Definir los hitos con una separación de tiempo adecuada.
- Definir la dependencia ente las actividades.
- Definir el calendario de actividades e hitos para apoyar la exactitud en la medición del progreso.
- Determinar el tiempo introduciendo las etapas de las actividades.
- Determinar rupturas en las programaciones subordinadas.
- Usar datos históricos para verificar la programación.

SP 2.4 Planificar los recursos del proyecto: su objetivo es definir los recursos del proyecto (mano de obra, maquinaria y equipos, materiales, y métodos) y las cantidades necesarias para realizar las actividades del proyecto. Para ello propone seguir el siguiente proceso:

1. Determinar los requisitos del proceso. Los procesos usados para gestionar el proyecto deben estar identificados, definidos y coordinados con los stakeholders.

2. Determinar las necesidades de personal. Estas necesidades dependen de la descomposición del proyecto en tareas, roles y responsabilidades.
3. Determinar las facilidades, equipamiento y componentes necesarios.

En el área Seguimiento y Control del Proyecto son fundamentales los siguientes objetivos específicos y prácticas específicas:

SG 1 Seguimiento del proyecto contra la planificación

SP 1.1 Seguimiento de los parámetros de planificación del proyecto: el seguimiento consiste en medir los valores reales de los parámetros de la planificación del proyecto, comparar los valores reales con los estimados en el plan, e identificar las desviaciones significativas. Para ello propone seguir el siguiente proceso:

1. Seguimiento del progreso comparando el plan actual y el programado.
2. Seguimiento del coste y esfuerzo gastado en el proyecto para identificar desviaciones sobre los presupuestos del plan.
3. Seguimiento de los atributos de los componentes de productos y tareas.
4. Seguimiento de los recursos proporcionados y usados.

SP 1.6 Realizar evaluaciones del progreso: Las evaluaciones del progreso son revisiones regulares sobre el proyecto con el objetivo de mantener a los stakeholders informados. Estas revisiones pueden ser informales o pueden no estar explícitas en el plan.

SG2 Gestión de acciones correctivas

SP 2.1 Analizar cuestiones: su objetivo es recolectar y analizar los problemas, y determinar las acciones correctivas necesarias para resolverlos.

SP 2.2 Tomar acciones correctivas: primero propone determinar y documentar las medidas adecuadas y necesarias para abordar los problemas identificados, por ejemplo, modificar los requisitos, revisar las estimaciones y planes, renegociar compromisos, añadir recursos, etc. Después se debe llegar a un acuerdo con los stakeholders sobre la adopción de estas medidas, y finalmente, negociar estos cambios con los compromisos internos y externos.

SP 2.3 Gestionar las acciones correctivas: su objetivo es realizar un seguimiento de la completitud de las acciones correctivas y analizar los resultados para demostrar que son efectivas.

En el área Medición y Análisis son fundamentales los siguientes objetivos específicos y prácticas específicas:

SG 1 Alinear las actividades de medición y análisis

SP 1.1 Establecer los objetivos de medición: su objetivo es determinar los objetivos que se quieren medir y las acciones a adoptar según los resultados de los análisis. El resultado de aplicación de esta práctica es un documento con esta información.

SP 1.2 Especificar las medidas: su objetivo es refinar los objetivos anteriores en medidas cuantificables y precisas. Estas medidas son base si se obtienen directamente de la medición, o derivadas si combina medidas de otros datos. CMMI propone ejemplos de medidas base (estimación y medición actual del esfuerzo y coste, medidas de calidad, etc.) y medidas derivadas (valor acumulado, densidad de defectos, etc.).

SP 1.4 Especificar los procedimientos de análisis: su objetivo es especificar cómo serán analizados y presentados los datos medidos. CMMI orienta la elección de las técnicas de análisis, por ejemplo que estas técnicas sean visuales, basadas en datos estadísticos, utilizar herramientas adecuadas, etc. sin especificar cómo.

Comentarios Finales acerca CMMI

El modelo CMMI se desarrolló para proveer a una organización una guía para la mejora del proceso organizacional y de las capacidades de gestionar el desarrollo, adquisición y mantenimiento de productos y servicios. CMMI no es un proceso, es un modelo que describe las características de procesos efectivos [10]. Ofrece una colección de buenas prácticas para compararlas con las buenas prácticas de las organizaciones, y así mejorar el proceso. Estas buenas prácticas son procesos que las organizaciones deben añadir a su forma de trabajo para poder mejorar en ciertas áreas de la organización. La organización es la encarga de decidir cómo incluir estos procesos, puesto que CMMI no indica métodos, técnicas o herramientas útiles para llevarlo a cabo, es decir, CMMI ayuda a saber QUÉ hacer sin saber CÓMO.

Con lo que respecta a la planificación y seguimiento de proyectos de desarrollo y mantenimiento software, CMMI ofrece buenas prácticas que indican que se debe estimar el alcance del proyecto, las tareas, esfuerzos y costes, y que se debe desarrollar un plan de proyecto estableciendo un calendario con las estimaciones anteriores. Entre las actividades que CMMI aconseja para establecer el plan de proyecto, está la referente a la organización temporal de las tareas (recomienda el uso de técnicas como PERT o CPM), la planificación de recursos, y el seguimiento del proyecto basándose en la comparación de los valores actuales con los establecidos en la planificación.

2.2. Proceso Software Personal (PSP)

El modelo CMM evalúa y estima la madurez de los procesos software de una organización. Sin embargo, un factor importante que determina la productividad y calidad de los productos es la madurez y capacidad del desarrollar individual. Motivado por el éxito de CMM, el SEI comenzó a trabajar en el Proceso Software Personal [12].

El proceso personal de software es un conjunto de prácticas disciplinadas para la gestión del tiempo y mejora de la productividad personal de los programadores o ingenieros de software, en tareas de desarrollo y mantenimiento de software. Está alineado y diseñado para emplearse en organizaciones con modelos de procesos CMMI o ISO 15504. Además, fue propuesto por Watts Humphrey en 1995 y estaba dirigido a estudiantes. A partir de 1997 con el lanzamiento del libro "An introduction to the Personal Software Process" se dirige a ingenieros principiantes. Se puede considerar como la guía de trabajo personal para ingenieros de software en organizaciones que emplean un modelo CMMI con nivel de madurez o de capacidad de procesos que implica la medición cualitativa y mejora de procesos. PSP define 4 niveles de madurez e identifica los pasos necesarios para alcanzar el siguiente nivel de madurez; Medición Personal (PSP0), Planificación Personal (PSP1), Calidad Personal (PSP2) y Proceso Cíclico (PSP3).

PSP proporciona métodos detallados de planificación y estimación, muestra a los ingenieros cómo controlar su rendimiento frente a estos planes y explica cómo los procesos definidos guían su trabajo.

El diseño de PSP se basa en los siguientes principios de planificación y de calidad [45]:

- Cada ingeniero es diferente; para ser más eficiente, debe planificar su trabajo basándose en datos tomados de su propia trayectoria profesional.
- Para mejorar auténticamente su trabajo, los ingenieros deben usar procesos personales bien definidos y cuantificados.
- Para obtener productos de calidad, el ingeniero debe asumir la responsabilidad personal de la calidad de sus productos. Los buenos productos no se obtienen por azar, sino como consecuencia de un esfuerzo positivo para hacer un trabajo de calidad.
- Cuanto antes se detecten y corrijan los defectos menos esfuerzo será necesario.
- Es más efectivo evitar los defectos que detectarlos y corregirlos.
- Trabajar bien es siempre la forma más rápida y económica de trabajar.

El Proceso Software Personal, desde el punto de vista de procesos de desarrollo, es un marco de trabajo que ayuda a los ingenieros de software a medir y mejorar su forma de trabajar. Las fases de PSP son la Planificación (se describen las tareas y se realizan las estimaciones temporales y de tamaño), Diseño, Codificación, Compilación, Pruebas, y la fase Postmortem (se calculan los tiempos reales invertidos).

El plan de proyecto, según PSP, define el trabajo y cómo se hará, y corresponde con la primera fase del proceso PSP (Planificación). Las actividades principales que proporciona PSP para el plan de proyecto son: definir las tareas, estimar el tiempo y los recursos necesarios, y ofrecer un marco de trabajo para gestionar la revisión y el control. Un plan de proyecto bien documentado, es un punto de referencia para comparar con el rendimiento real. Esta comparación permite a los planificadores ver sus errores de estimación y mejorar su exactitud en la planificación [20].

Puesto que la fase de Planificación es dónde el proceso PSP realiza la planificación y seguimiento de los proyectos de desarrollo software, vamos a ver cómo aborda cada una de las actividades anteriores.

Después de obtener los requisitos y definir las tareas, PSP propone definir el Plan de Proyecto que incluye, inicialmente, una estimación del tamaño del producto, del tiempo estimado en hacer el trabajo y la fecha de finalización del trabajo. Si por ejemplo las estimaciones son muy bajas o muy altas, PSP recomienda realizar ajustes. Estas estimaciones se introducen en el Plan de Proyecto formado por las Figura 3 y Figura 4,

a mano alzada. También PSP recomienda utilizar datos históricos, de forma que comprobando los valores de tiempo medio, máximo y mínimo para los trabajos más recientes, se puede seleccionar el valor más adecuado para el nuevo proyecto. La sección Resumen de la Figura 3 contiene los datos de velocidad utilizados para hacer el plan, tanto los estimados (columna Plan) como los reales (columna Real). Los campos de la sección Resumen son:

- Minutos/LOC: minutos por líneas de código.
- LOC/Hora: líneas de código por hora. Normalmente se utiliza para analizar la productividad por hora.
- Defectos/KLOC
- Rendimiento
- Valoración/Fallo

La sección Tamaño del Programa (LOC) de la Figura 3, contiene los datos estimados y reales de los tamaños de los programas y de los rangos de dichos tamaños. La medida utilizada para el tamaño del programa son las líneas de código, sin tener en cuenta comentarios ni líneas en blanco. Según PSP, para estimar el tamaño de un programa, primero se debe examinar los requisitos de los programas a desarrollar, después se clasifican los tamaños relativos de los nuevos programas en los que ya están escritos, y finalmente, se ubican dentro del histórico de rangos de tamaños. Por ejemplo, los tamaños máximo y mínimo de la Figura 3, se han obtenido del histórico de rangos de programas de características similares al nuevo (606 y 506 respectivamente). El agente propietario del programa debe determinar, mediante un juicio, los programas que usará como valores extremos. El tamaño total (valor 556 de la Figura 3) corresponde al tamaño medio de los valores extremos.

Ingeniero: X			
Programa: AFG			Fecha: 11/05/08
Descripción: Aplicación que permite el registro y control del Almacén AFG Orientado a la venta de Fideos y Galletas			Lenguaje: Delphi 5
Resumen	**Plan**	**Real**	**Hasta la fecha**
Minutos/LOC	2	3.5	3.5
LOC/Hora	30	17	17
Defectos/KLOC		61.6	61.6
Rendimiento		55.6 %	55.6 %
Valoración/Fallo		0.52	0.52
Tamaño programa (LOC)	**Plan**	**Real**	**Hasta la fecha**
Total nuevo & cambiado	556	730	730
Tamaño máximo	606		
Tamaño mínimo	506		

Figura 3 Ejemplo de resumen del plan del proyecto

La sección Tiempo por Fase del Plan de Proyecto de la Figura 4, se utiliza para introducir los datos de las fases del proceso de desarrollo. Cada fila representa los tiempos planificados y reales para cada fase del proceso. Durante la fase de planificación los datos se escriben en la columna Plan. Para estimar el tiempo que se va a dedicar en cada fase, se asigna a cada fase un porcentaje de tiempo de desarrollo total (columna % Hasta la Fecha de la Figura 4), basándose en la utilización del tiempo en proyectos anteriores.

Para calcular el tiempo total planificado para el desarrollo de un nuevo programa, se estima el tamaño del programa en LOC y se multiplica por los Minutos/LOC planificados de la Figura 3. Los tiempos máximos y mínimos son calculados de la misma forma, pero con sus respectivas LOC de la Figura 3.

Tiempo por Fase (min.)	Plan	Real	Hasta la fecha	% Hasta la fecha
Planificación		50	50	2 %
Diseño		353	353	14 %
Codificación		1008	1008	40 %
Revisión del código		374	374	15 %
Compilación		80	80	3 %
Pruebas		636	636	25 %
Postmorten		39	39	1 %
Total	**1112**	**2540**	**2540**	**100 %**
Tiempo máximo	1212			
Tiempo mínimo	1012			

Figura 4 Ejemplo de resumen del plan del proyecto

Durante la fase de Postmorten y una vez finalizado el programa, se escribe el tiempo de desarrollo real en la columna Real (en minutos), tanto el tiempo total como el tiempo de cada fase del proceso de desarrollo. La columna Hasta la Fecha contiene el total de todos los tiempos dedicados en cada fase para todos los programas acabados. La columna de % Hasta la Fecha tiene el porcentaje de los tiempos de la columna Hasta la Fecha. Para cada registro correspondiente a la columna Hasta la Fecha del Plan de Proyecto de las Figura 3 y Figura 4, se escribe la suma del tiempo real y el tiempo Hasta la Fecha de los programas más recientes. En la columna % Hasta la Fecha, se multiplica 100 por el tiempo Hasta la Fecha y se divide por el total de tiempo Hasta la Fecha. Estas dos columnas proporcionan una forma sencilla de calcular la distribución de los porcentajes de tiempo de desarrollo para las fases del proyecto.

El tiempo real se va registrando en el Cuaderno de Registro de Tiempos que se muestra en la Figura 5. Cada periodo de tiempo se introduce en una línea con la siguiente información:

- Fecha: fecha de realización de alguna actividad.
- Comienzo y fin de la actividad.
- Interrupción: pérdidas de tiempo debido a interrupciones.
- Δ Tiempo: tiempo dedicado a la actividad en minutos, entre los tiempos de comienzo y fin menos el tiempo de interrupción.
- Actividad.
- Comentarios: descripción completa de lo que se está haciendo.
- C: se rellena cuando se completa la tarea (Completado).
- U: número de unidades de una tarea acabada (Unidades).

Fecha	Comienzo	Fin	Tiempo de Interrupción	A Tiempo	Actividad	Comentarios	C	U
15/04/08	15:10	15:45	0	35	Diseñar F. Insertar Galleta		X	1
	15:50	16:40	0	50	Diseñar F. Modificar Galleta		X	1
	16:30	16:58	0	28	Diseñar F. Eliminar Galleta		X	1
	17:10	17:40	0	30	Diseñar F. Insertar Fideo		X	1
16/04/08	14:50	15:30	0	45	Diseñar F. Modificar Fideo		X	1
	15:45	16:05	0	20	Diseñar F. Eliminar Fideo		X	1
	16:15	17:06	10	41	Diseñar F. Búsqueda	Descanso	X	1
17/04/08	15:05	16:28	5, 10	68	Diseñar F. que realice consultas	Descanso, Descanso	X	1
	16:35	17:21	10	36	Diseñar F. que dibuje estadísticas	Descanso	X	1

Figura 5 Cuaderno de Registro de Tiempos

PSP indica que cuando se olvide registrar una hora de inicio, de fin o la duración de una interrupción, se puede hacer una estimación ya que es lo más parecido al tiempo real. También propone utilizar un cronómetro para controlar las interrupciones, y recomienda, que se debe resumir el tiempo puntualmente. Según Humphrey (2001) [45], otra aproximación para registrar tiempos es utilizar un computador. Humphrey destaca: “He intentado esto y he observado que lleva más tiempo y era menos adecuado que tomar notas en papel”. En secciones posteriores y como propuesta nuestra, veremos que en el ámbito de los procesos de desarrollo, registrar los tiempos en un computador es una muy buena práctica.

A continuación, vamos a explicar cómo PSP controla el progreso del proyecto mediante las programaciones, y cómo realiza el seguimiento del plan de proyecto. Cuando se tienen varios compromisos de trabajo es necesaria una programación y una gestión del tiempo más sofisticada. Según Humphrey (2001) [45], una programación es una lista ordenada por tiempos de eventos planificados, por ejemplo, un Diagrama de Gantt. Esta técnica permite coordinar el trabajo de múltiples actividades, y seguir el progreso del proyecto frente al plan y la programación. Además, para controlar el progreso del proyecto, el trabajo debe dividirse en varias partes que puedan ser estimadas y planificadas. Estos puntos de la programación son medibles para determinar el progreso, y reciben el nombre de puntos de control o hitos.

El seguimiento de un plan de proyecto permite determinar si el proyecto va adelantado o retrasado según lo programado, y actuar a tiempo frente a los problemas detectados.

PSP también utiliza los Diagramas de Gantt para informar del progreso frente a lo programado. En el diagrama, la fecha actual se indica por medio de una línea vertical doble. Las actividades parcialmente terminadas o completas se indican mediante una línea horizontal sobre el bloque de la tarea, tal y como se muestra en la actividad "Call Joe" de la Figura 6.

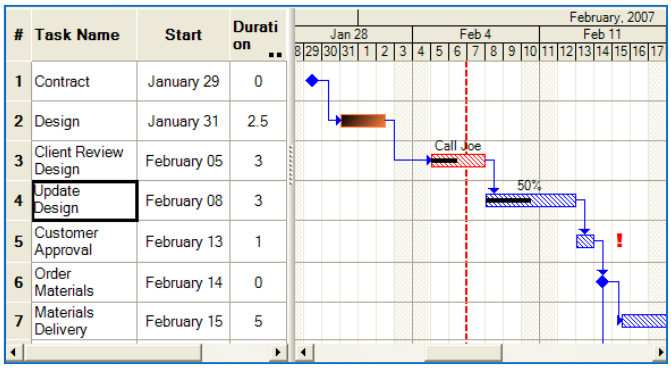


Figura 6 Diagrama de Gantt

Comentarios Finales respecto PSP

PSP ha sido desarrollado para mejorar el rendimiento individual de los desarrolladores. En esta sección nos hemos centrado en las técnicas propuestas por PSP para planificar el trabajo, gestionar el tiempo, controlar el progreso del proyecto y realizar el seguimiento del trabajo. Para planificar el trabajo individual, los desarrolladores deben estimar los tiempos de desarrollo y productividad, y el tamaño de lo que van a desarrollar. Estas estimaciones se hacen a partir de los datos históricos recopilados y un juicio de valor. Al iniciar los proyectos, las estimaciones no son precisas ya que no se dispone de datos históricos, pero con el tiempo, las estimaciones pasan a ser más precisas ya que se dispone de más datos y los desarrolladores están más entrenados. La gestión del tiempo se realiza a través de cuadernos o cualquier otro soporte informático para la recolección de los datos temporales invertidos. Kamatar y Hayes (2000) [27] realizaron un informe sobre su experiencia con PSP y consideraron que la recolección de datos fue difícil a veces. Tuvieron que almacenar los datos en varios soportes (notas en papel, hojas de cálculo, cuadernos, etc.), y luego llevarlos a hojas de cálculo (Plan de Proyecto) durante la fase Postmortem. Aunque con el tiempo los desarrolladores aprenden esta disciplina de registro de tiempo, requiere ser ordenado, riguroso y constante en los registros para que sus planificaciones sean precisas.

La técnica utilizada por PSP para controlar y realizar un seguimiento del progreso y estado del proyecto es el diagrama de Gantt. Estos diagramas permiten a los desarrolladores tener una visión clara de la organización temporal de sus tareas, del estado en el que se encuentra cada una de ellas, y compararlas para ver si cumple o no con sus compromisos. Las técnicas propuestas por PSP, soportan un proceso de desarrollo iterativo a nivel de granularidad individual o de un equipo pequeño de desarrolladores, pero cuando requiere de un equipo o proyecto mayor, es necesario realizar una asignación de los recursos en base a sus disponibilidades y balancear sus cargas de trabajo, a lo cual no da soporte. Por tanto, las prácticas PSP están limitadas a equipos de desarrollo pequeños, pero ofrecen muy buenos consejos en cuanto a gestión del tiempo.

2.3. Project Management Body of Knowledge (PMBOK)

La Guía del PMBOK® es un estándar para la gestión de proyectos de desarrollado propuesto por el Project Management Institute (PMI). En 1987, el PMI publicó la primera edición del PMBOK en un intento por documentar y estandarizar información y prácticas generalmente aceptadas en la gestión de proyectos. Actualmente, el PMBOK ofrece la cuarta edición realizada en Diciembre de 2008. La Guía PMBOK proporciona y promueve un vocabulario común para la discusión, la escritura y la aplicación de la gestión de proyectos.

La gestión de proyectos es la aplicación de los conocimientos, habilidades, herramientas y técnicas a las actividades del proyecto para satisfacer los requisitos del proyecto. La gestión de proyectos se realiza a través de procesos, utilizando los conocimientos de gestión de proyectos, habilidades, herramientas y técnicas que reciben entradas y generan salidas [38].

El PMBOK es una colección de procesos y áreas de conocimiento generalmente aceptadas como las mejores prácticas dentro de la gestión de proyectos. El PMBOK es un estándar reconocido internacionalmente (IEEE Std. 1490-2003) [24] que provee los fundamentos de la gestión de proyectos que son aplicables a un amplio rango de proyectos, incluyendo construcción, software, ingeniería, etc. Los procesos de gestión de proyectos describen qué hacer para administrar un proyecto, mientras que el ciclo de

vida de un proyecto describe el trabajo involucrado en un proyecto. Este estándar documenta información necesaria para iniciar, planificar, ejecutar, supervisar y controlar, y cerrar un proyecto. El estándar describe la naturaleza de los procesos de gestión de proyectos en términos de integración entre los procesos, las interacciones entre ellos, y los fines que persiguen.

El PMBOK reconoce 5 grupos de procesos básicos y 9 áreas de conocimiento comunes a casi todos los proyectos. Los procesos se cubren e interactúan a través de un proyecto o fase. Los procesos son descritos en términos de: Entradas (documentos, planes, diseños, etc.), Herramientas y Técnicas (mecanismos aplicados a las entradas), y Salidas (documentos, productos, etc.). Los procesos básicos reconocidos por PMBOK son los Procesos de Inicio, Procesos de Planificación, Procesos de ejecución, Procesos de seguimiento y control, y Procesos de cierre que se muestran en la Figura 7.

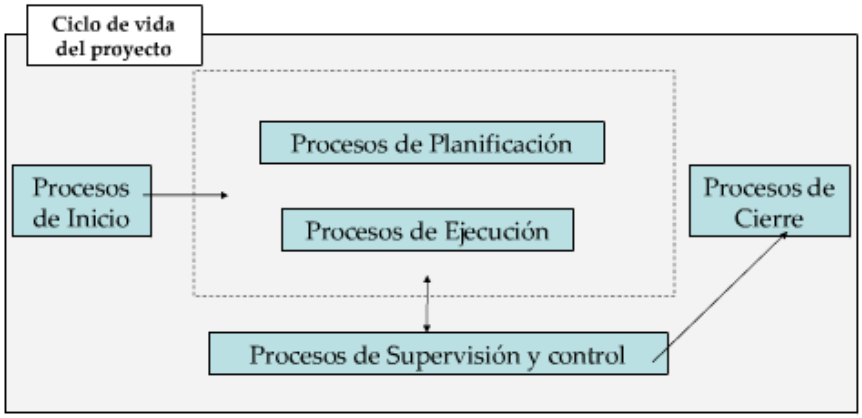


Figura 7 Ciclo de vida del proyecto y los 5 procesos básicos.

Las 10 áreas del conocimiento mencionadas en el PMBOK son: Gestión de la Integración, Gestión del Alcance, Gestión del Tiempo, Gestión de la Calidad, Gestión de Costos, Gestión del Riesgo, Gestión de Recursos Humanos, Gestión de la Comunicación, y Gestión de las Compras y Adquisiciones. Cada una de estas áreas se ven afectadas en uno de los procesos básicos de PMBOK. A continuación se muestran todos los procesos clasificados en el proceso básico en el que está incluido, y el área de conocimiento al que pertenece.

Procesos de Iniciación	
Procesos	**Área de conocimiento**
Desarrollar el Acta de Constitución del Proyecto	Integración
Desarrollar el Enunciado del Alcance Preliminar del Proyecto	Integración

Procesos de Planificación	
Procesos	**Área de conocimiento**
Planificación del Alcance	Alcance
Definición del alcance	Alcance
Crear EDT (WBS)	Alcance
Definición de actividades	Tiempo
Establecimiento de la secuencia de actividades	Tiempo
Estimación de los recursos de las actividades	Tiempo
Estimación de la duración de las actividades	Tiempo
Desarrollo del cronograma	Tiempo
Estimación de costos	Costo
Presupuesto del proyecto	Costo
Planificación de la calidad	Calidad
Planificación de los recursos humanos	RRHH
Planificación de las comunicaciones	Comunicaciones
Planificación de la gestión de riesgos	Riesgos
Identificación de riesgos	Riesgos
Análisis cualitativo de riesgos	Riesgos
Análisis cuantitativo de riegos	Riesgos
Planificación de la respuesta a riesgos	Riesgos
Planificar las compras y adquisiciones	Adquisiciones
Planificar la contratación	Adquisiciones
Procesos de Ejecución	
Procesos	**Área de conocimiento**
Realizar el aseguramiento de calidad	Calidad
Adquirir el equipo de proyecto	RRHH
Desarrollar el equipo de proyecto	RRHH
Distribución de la información	Comunicaciones
Solicitar respuestas de vendedores	Adquisiciones
Selección de vendedores	Adquisiciones
Procesos de Seguimiento y Control	
Procesos	**Área de conocimiento**
Verificación del alcance	Alcance
Control del alcance	Alcance
Control del cronograma	Tiempo
Control de costos	Costo
Realizar control de calidad	Calidad
Gestionar el equipo del proyecto	RRHH

Informar el desempeño	Comunicaciones
Gestionar a los interesados	Comunicaciones
Seguimiento y control de riesgos	Riesgos
Administración del contrato	Adquisiciones
Procesos de Cierre	
Procesos	**Área de conocimiento**
Cerrar el proyecto	Integración
Cierre del contrato	Adquisiciones

Tabla 3 Procesos y Áreas de Conocimiento

Como objetivo de este capítulo vamos a centrarnos en los procesos de Planificación, y Seguimiento y Control, concretamente, en las áreas de conocimiento de Alcance, Tiempo, y RRHH puesto que están relacionas con este trabajo. Para cada uno de los procesos, el PMBOK especifica las entradas, técnicas o herramientas, y salidas.

Gestión del Alcance en el Proceso de Planificación

La gestión del alcance del proyecto incluye los procesos necesarios para asegurar que el proyecto incluya todo el trabajo requerido para completar exitosamente el proyecto. Éste vela principalmente por definir y controlar lo que se incluye y lo que no se incluye en el proyecto. A continuación se detallan los procesos de la Gestión del Alcance relacionados con nuestro trabajo, la planificación del alcance y crear la Estructura de División de Trabajo.

La **planificación del alcance** es el proceso de elaborar y documentar progresivamente el trabajo del proyecto (alcance del proyecto) que da lugar al producto del proyecto. La planificación del alcance comienza con las entradas iniciales de la descripción del producto, el permiso legal del proyecto y la definición inicial de las restricciones y supuestos (ver Tabla 4). Cabe indicar que la descripción del proyecto incorpora los requerimientos del producto que refleja las necesidades del cliente que se han acordado, y el diseño del producto que cumple con los requerimientos del producto. Los resultados o consecuencias de la planificación del alcance son la declaración del alcance y el plan de gestión del alcance, junto el detalle de respaldo. La declaración del alcance forma la base de un acuerdo entre el proyecto y el cliente del proyecto, identificando tanto los objetivos del proyecto como las prestaciones del mismo. Los equipos de proyecto desarrollan múltiples declaraciones de alcance que son las adecuadas para el nivel de descomposición del trabajo del proyecto. Se utilizan las técnicas del juicio

experto, plantillas, formularios o estándares para elaborar y documentar el trabajo del proyecto (requisitos en la cuarta edición del PMBOK).

Según el PMBOK, es necesario el *juicio experto* con el fin de evaluar las entradas para un proceso determinado. Dicha práctica puede ser proporcionada por cualquier grupo o individuo con un conocimiento o entrenamiento especializado, y se puede obtener de muchas fuentes, como unidades dentro de la organización o consultores por ejemplo.

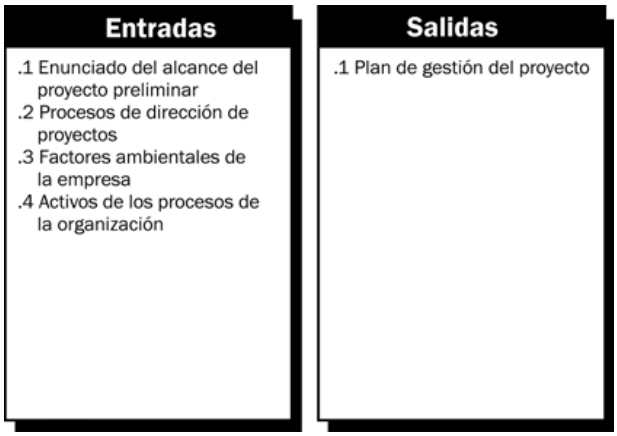


Tabla 4 Desarrollar el Plan de Gestión del Proyecto: Entradas y Salidas

El **crear la Estructura de División de Trabajo** (EDT o WBS en inglés) implica la subdivisión de las principales prestaciones del proyecto en componentes más pequeños y más manejables, con el objeto de:

- Mejorar la precisión en las estimaciones de costo, duración y recursos.
- Definir una base para la medición y control del rendimiento/desempeño.
- Facilitar las claras asignaciones de responsabilidades.

La adecuada definición del alcance es crítica para el éxito del proyecto. Las entradas y salidas de este proceso son las que se muestran en la Tabla 5. Las herramientas o técnicas utilizadas para realizar este desglose son las plantillas de Estructuras de División del Trabajo y la Descomposición. Veamos a continuación cada una de estas técnicas.

Entradas	Salidas
.1 Activos de los procesos de la organización .2 Enunciado del alcance del proyecto .3 Plan de gestión del alcance del proyecto .4 Solicitudes de cambio aprobadas	.1 Enunciado del alcance del proyecto (actualizaciones) .2 Estructura de desglose del trabajo .3 Diccionario de la EDT .4 Línea base del alcance .5 Plan de gestión del alcance del proyecto (actualizaciones) .6 Cambios solicitados

Tabla 5 Definición del Alcance: Entradas y Salidas

Una *EDT* es una agrupación orientada a la definición de los componentes del proyecto, la cual organiza y define el alcance total del proyecto; todo trabajo que no esté contemplado en la EDT está fuera del alcance del proyecto. Al igual que la declaración del alcance, la EDT se utiliza, con frecuencia, para desarrollar o confirmar una comprensión común del alcance del proyecto. Cada nivel descendente representa una descripción significativamente detalla de los entregables del proyecto. La EDT se presenta normalmente en forma de diagrama, como se ilustra en la Figura 8. Los ítems del nivel más bajo de la EDT pueden ser referidos como paquetes de trabajo.

Las descripciones de los componentes del trabajo son, a menudo, recopilados en un diccionario de EDTs. Este diccionario incluirá, comúnmente, las descripciones de los paquetes de trabajo, como también otra información de planificación como son las fechas de programas, los presupuestos de costos y las designaciones de personal.

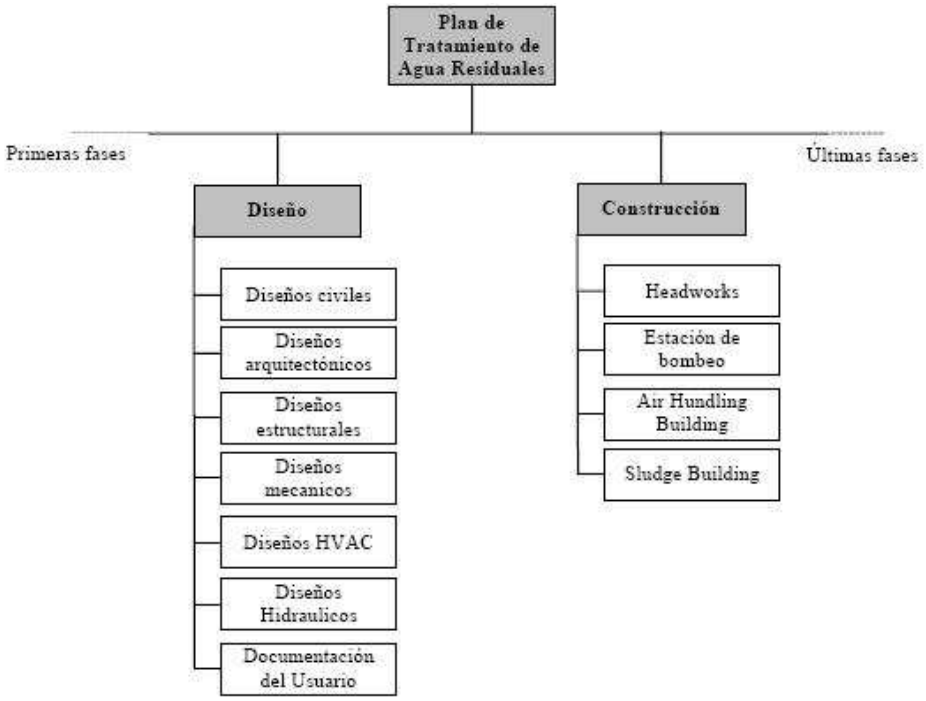


Figura 8 Ejemplo de EDT

La EDT de un proyecto anterior, puede ser utilizada como plantilla para un nuevo proyecto. Aun cuando cada proyecto es único, a menudo es posible "reutilizar" las EDTs dado que la mayoría de los proyectos serán, de cierto modo, similares a otros. Por ejemplo, gran parte de los proyectos de una determinada organización tendrán ciclos de vida de proyectos iguales o similares y, por lo tanto, tendrán los mismos entregables u otros similares a los requeridos de cada fase.

La técnica de *descomposición* implica subdividir los principales entregables del proyecto en componentes más pequeños y más manejables, hasta que se definan con suficiente nivel de detalle como para respaldar el desarrollo de las actividades del proyecto (planificación, ejecución, control y término). La descomposición implica los principales pasos que a continuación se indican:

- Identificar los principales entregables del proyecto, incluida la gestión del proyecto. Deben definirse siempre en términos de cómo se organizará realmente el proyecto, por ejemplo, las fases del ciclo de vida.
- Decidir si se pueden desarrollar estimaciones de costos y duración adecuadas con cierto nivel de detalle para cada una de los entregables.
- Identificar los componentes que formen parte de la prestación. Estos componentes (servicios o productos por ejemplo) se deben describir en términos

de resultados tangibles y verificables, con el objeto de facilitar la medición del rendimiento/desempeño.

- Verificar que la descomposición se haya realizado correctamente, es decir, es necesaria y suficiente.

Gestión del Tiempo en el Proceso de Planificación

La gestión del tiempo del proyecto incluye los procesos requeridos para asegurar que el proyecto se complete a tiempo o puntualmente.

Los principales procesos involucrados en el desarrollo del programa o calendario del proyecto, son:

- La Definición de las Actividades es el proceso necesario para identificar las actividades específicas que deben realizarse para producir los diversos productos entregables del proyecto.
- El Establecimiento de la Secuencia de las Actividades es el proceso necesario para identificar y documentar las dependencias entre las actividades del cronograma.
- La Estimación de Recursos de las Actividades es el proceso necesario para estimar los tipos y las cantidades de recursos necesarios para realizar cada actividad del cronograma.
- La Estimación de la Duración de las Actividades es el proceso necesario para estimar la cantidad de períodos laborables que se requerirán para completar cada actividad del cronograma.
- El Desarrollo del Cronograma es el proceso necesario para analizar las secuencias de las actividades, la duración de las actividades, los requisitos de los recursos y las restricciones del cronograma para crear el cronograma del proyecto.

A continuación se detallan los procesos de la Gestión del tiempo relacionados con nuestro trabajo, que corresponde con todos los procesos resumidos en los puntos anteriores de esta sección.

La **definición de las actividades** implica la identificación y documentación de las actividades específicas que se deberán ejecutan con el objeto de producir los entregables identificados en la Estructura de División del Trabajo (EDT). Las entradas y salidas de este proceso son las que se muestran en la Tabla 6. Las herramientas o técnicas utilizadas para realizar la definición de las actividades son la descomposición, el uso de plantillas, la técnica de planeación continua con incremento de detalle, juicio del experto y el componente de planificación. Veamos a continuación algunas de estas técnicas.

Entradas	Salidas
.1 Factores ambientales de la empresa .2 Activos de los procesos de la organización .3 Enunciado del alcance del proyecto .4 Estructura de desglose del trabajo .5 Diccionario de la EDT .6 Plan de gestión del proyecto	.1 Lista de actividades .2 Atributos de la actividad .3 Lista de hitos .4 Cambios solicitados

Tabla 6 Definición de las Actividades: Entradas y Salidas

Dentro del contexto del proceso de Definición de las Actividades, la *descomposición* implica subdividir los paquetes de trabajo del proyecto en componentes más pequeños y más manejables, a fin de permitir un mejor control de gestión. La principal diferencia entre la modalidad de descomposición que tratamos en este apartado y la descrita en la Definición del Alcance, está en que los resultados finales en este caso se describen como actividades en vez de cómo prestaciones. La EDT y la lista de actividades se desarrollan, por lo general, en forma secuencial, siendo la EDT la base de desarrollo de la lista final de actividades.

La lista de actividades a la que hace referencia PMBOK, debe incluir todas las actividades que se llevarán a cabo en el proyecto. Ésta debe organizarse como una extensión de la EDT, a fin de ayudar a garantizar que se encuentre completa, y que no incluye ninguna actividad que no sea requerida como parte del alcance del proyecto. Como con la EDT, la lista de actividades debe incluir descripciones de cada actividad, a fin de asegurarse de que los miembros del equipo del proyecto entiendan cómo se va a realizar el trabajo.

A menudo, se utiliza como *plantilla* para un nuevo proyecto una lista de actividades o una parte de la lista de actividades de un proyecto anterior. Las actividades de las plantillas pueden contener, también, una lista de las habilidades (recursos) y sus horas de trabajo requeridas, una identificación de los riesgos, las prestaciones esperadas y cualquier otra información descriptiva.

Rolling Wave Planning o la técnica de planeación continua con incremento de detalle, hace referencia a una técnica de planificación que consiste en planificar en detalle el trabajo que se tiene que realizar a corto plazo, mientras que el trabajo a largo plazo se planifica a un nivel más alto de la EDT. A medida que avanza el desarrollo del proyecto, y se conocen más detalles de la siguiente fase, se va planificando en detalle lo que anteriormente sólo estaba planificado a alto nivel. Esta técnica es muy útil en proyectos en los que hay un grado de incertidumbre elevado y a medida que se van conociendo detalles de las siguientes fases podemos ir planificando los detalles.

La **secuencia de las actividades** implica identificar y documentar las relaciones lógicas entre las actividades. Las actividades se deben secuenciar de manera exacta, a fin de respaldar el posterior desarrollo de un programa realista y alcanzable. Las entradas y salidas de este proceso son las que se muestran en la Tabla 7. Las herramientas o técnicas utilizadas para realizar la definición de las actividades son el Método de Diagramas de Precedencia (MDP o PDM en inglés), Método del Diagrama de Flechas (MDF o ADM en inglés), plantillas de red, métodos de determinación de dependencia y la aplicación de adelantos y retrasos. Veamos a continuación algunas de estas técnicas.

Entradas	Salidas
.1 Enunciado del alcance del proyecto .2 Lista de actividades .3 Atributos de la actividad .4 Lista de hitos .5 Solicitudes de cambio aprobadas	.1 Diagrama de red del cronograma del proyecto .2 Lista de actividades (actualizaciones) .3 Atributos de la actividad (actualizaciones) .4 Cambios solicitados

Tabla 7 Establecimiento de la Secuencia de las Actividades: Entradas y Salidas

El *Método de diagramas de precedencias* (MDP) permite construir un diagrama de red de proyecto que utiliza cuadros o rectángulos (nodos) para representar las actividades y

las conecta con flechas que muestran las relaciones de dependencia. Este método incluye cuatro tipos de relaciones de dependencia o precedencia:

- Terminar-para-comenzar – el inicio del trabajo del sucesor depende del término del trabajo del predecesor.
- Terminar-para-terminar – el término del trabajo del sucesor depende del término del trabajo del predecesor.
- Comenzar-para-comenzar – el inicio del trabajo del sucesor depende del inicio del trabajo del predecesor.
- Comenzar-para terminar – el término del sucesor es dependiente del inicio del predecesor.

El *Método de diagramas con flechas* (MDF) permite construir un diagrama de red de proyecto que utiliza flechas para representar las actividades y las conecta en los nodos para demostrar sus relaciones de dependencia. El MDF utiliza únicamente relaciones de dependencia terminar-para-empezar y puede requerir el uso de actividades reiterativas con el objeto de definir correctamente todas las relaciones lógicas.

Las *Plantillas de red* tienen como objetivo acelerar la preparación de los diagramas de red del proyecto, es posible recurrir a redes estandarizadas. Éstas pueden incluir un proyecto completo o bien sólo una parte de éste. Las partes de una red se conocen a menudo como subredes. Las subredes son especialmente útiles cuando un proyecto incluye varias características idénticas o casi idénticas, tales como los pisos de un edificio de oficinas, los ensayos clínicos de un proyecto de investigación, etc.

La *determinación de la dependencia* consiste en que los miembros del equipo deben seleccionar el tipo de dependencia entre las actividades. Existen tres tipos de dependencias:

- Dependencia obligatoria - son aquellas que son inherentes a la naturaleza de la labor que realizan. Estas dependencias implican limitaciones físicas, como por ejemplo en un proyecto de electrónica, donde el modelo debe construirse antes de ser probado.
- Dependencia discrecional - se establecen sobre la base de conocimientos sobre las mejores prácticas dentro de un área particular o algún aspecto inusual del

proyecto donde se desea una secuencia específica, aunque hay otras secuencias aceptables.

- Dependencia externa - son las que implican una relación entre las actividades del proyecto y las actividades de fuera del proyecto.

En la *Aplicación de Adelantos y Retrasos*, el equipo de dirección del proyecto determina las dependencias que pueden requerir un adelanto o un retraso para definir con exactitud la relación lógica.

- Un adelanto permite la aceleración de la actividad sucesora. Por ejemplo, el equipo de redacción técnica puede comenzar a escribir el segundo borrador de un documento grande (la actividad sucesora) quince días antes de terminar de escribir el primer borrador completo (la actividad predecesora). Esto puede lograrse mediante una relación final a inicio con un período de adelanto de quince días.
- Un retraso causa una demora en la actividad sucesora. Por ejemplo, para dar cuenta del período de diez días que el hormigón tarda en curarse, se puede utilizar un retraso de diez días en una relación final a inicio, lo que significa que la actividad sucesora no puede comenzar hasta diez días después de finalizada la predecesora.

La **estimación de recursos de las actividades** consiste en determinar qué recursos (personas, equipos o materiales) y qué cantidades de cada recurso se utilizará, y cuándo cada uno de los recursos estará disponible para realizar actividades del proyecto. Las entradas y salidas de este proceso son las que se muestran en la Tabla 8. Las herramientas o técnicas utilizadas para realizar la estimación de recursos de las actividades son el juicio del experto, análisis alternativos, datos de estimación publicados, software de gestión de proyectos y la estimación ascendente. Veamos a continuación algunas de estas técnicas.

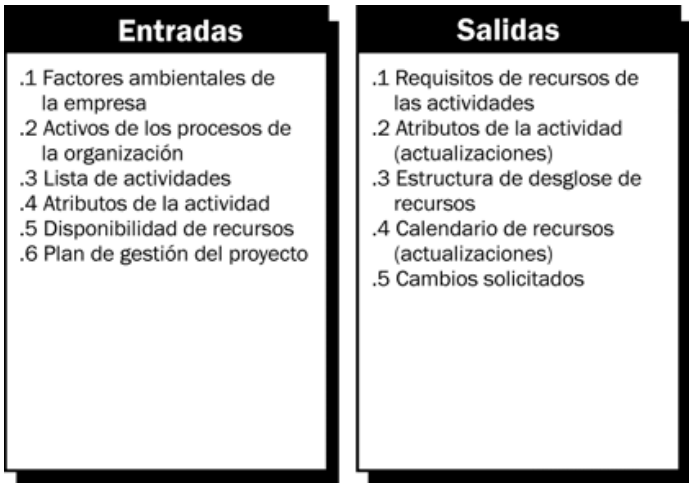


Tabla 8 Estimación de Recursos de las Actividades: Entradas y Salidas

Muchas actividades del cronograma o programación cuentan con *métodos alternativos* de realización. Éstos incluyen el uso de distintos niveles de capacidad o habilidades de los recursos, diferente tamaño o tipo de máquinas, diferentes herramientas (manuales frente a automatizadas) y la decisión de fabricación propia o compra a terceros con respecto al recurso.

En la técnica de *Datos de Estimación Publicados*, varias empresas publican periódicamente los índices de producción actualizados y los costes unitarios de los recursos para una extensa variedad de industrias, materiales y equipos, en diferentes países y en diferentes ubicaciones geográficas dentro de esos países.

El *software de gestión de proyectos* tiene la capacidad de ayudar a planificar, organizar y gestionar los conjuntos de recursos, y de desarrollar estimaciones de recursos. Dependiendo de la complejidad del software, podrán definirse las estructuras de desglose de recursos, las disponibilidades de recursos y las tarifas de recursos, así como también diversos calendarios de recursos.

En la *estimación de abajo-hacia-arriba* (Bottom-up Estimating), cuando no se puede estimar una actividad del cronograma con un grado razonable de confianza, el trabajo que aparece dentro de la actividad del cronograma se descompone con más detalle. Se estiman las necesidades de recursos de cada una de las partes inferiores y más detalladas del trabajo, y estas estimaciones se suman luego en una cantidad total para cada uno de los recursos de la actividad del cronograma. Las actividades del cronograma pueden o no tener dependencias entre sí que pueden afectar a la aplicación y al uso de los recursos. Si existen dependencias, este patrón de uso de

recursos se refleja en los requisitos estimados de la actividad del cronograma y se documenta.

La **estimación de la duración de las actividades** es el proceso de tomar información sobre el alcance del proyecto y de los recursos y luego, desarrollar las duraciones para ingresarlas en los programas. Las entradas para las estimaciones de duración se originan, comúnmente, a partir de la persona o grupo del equipo del proyecto que está más familiarizado con la naturaleza de una actividad específica. La estimación se elabora, a menudo, de forma progresiva, así es posible suponer que la estimación será progresivamente más exacta. La persona o grupo del equipo del proyecto que esté más familiarizado con la naturaleza de una actividad específica debe llevar a cabo, o al menos aprobar, la estimación. La estimación de la cantidad de periodos de trabajo requeridos para completar una actividad requerirá, a menudo, también de la consideración del tiempo transcurrido. Las entradas y salidas de este proceso son las que se muestran en la Tabla 9. Las herramientas o técnicas utilizadas para realizar la estimación de la duración de las actividades son el juicio del experto, estimación análoga, estimación paramétrica, estimaciones por Tres Valores (Tiempo PERT), y Análisis de Reserva. Veamos a continuación algunas de estas técnicas.

Entradas	Salidas
.1 Factores ambientales de la empresa .2 Activos de los procesos de la organización .3 Enunciado del alcance del proyecto .4 Lista de actividades .5 Atributos de la actividad .6 Requisitos de recursos de las actividades .7 Calendario de recursos .8 Plan de gestión del proyecto • Registro de riesgos • Estimación de costes de la actividad	.1 Estimación de la duración de las actividades .2 Atributos de la actividad (actualizaciones)

Tabla 9 Estimación de la Duración de las Actividades: Entradas y Salidas

A menudo, es difícil estimar las duraciones de las actividades debido a la cantidad de factores que pueden influir en ellas, como los niveles de recursos o la productividad de recursos. El *juicio del experto*, guiado por información histórica, puede usarse siempre que sea posible. Los miembros individuales del equipo del proyecto también pueden aportar información acerca de la estimación de la duración o las duraciones máximas recomendadas de las actividades, teniendo en cuenta proyectos anteriores similares. Si

no se cuenta con ese conocimiento, las estimaciones de la duración son más inciertas y arriesgadas.

La *estimación análoga*, también conocida como estimación de arriba-hacia-abajo, significa utilizar la duración real de una actividad previa similar como base para estimar la duración de una actividad futura. Se utiliza frecuentemente para estimar la duración del proyecto cuando existe una cantidad limitada de información detallada acerca del proyecto (por ejemplo, en las primeras fases). La estimación análoga es una forma de juicio del experto.

En la *estimación paramétrica*, la estimación de la base para las duraciones de las actividades pude determinarse cuantitativamente multiplicando la cantidad de trabajo a realizar por el ratio de productividad. Por ejemplo, los ratios de productividad en un proyecto de diseño pueden estimarse por la cantidad de dibujos multiplicado por las horas de trabajo por dibujo; o de una instalación de cable, en metros de cable por horas de trabajo por metro. Para determinar la duración de la actividad en períodos laborables, las cantidades totales de recursos se multiplican por las horas de trabajo por período laborable o la capacidad de producción por período laborable, y se dividen por la cantidad de recursos que se aplican.

> Ejemplo: Duración de la actividad.
> Pintado de una habitación de 40 m2
> Ratio productividad de un pintor: 10m2/h
> Duración: 40/10= 4 horas de trabajo (1/2 jornada de trabajo)

En la estimación *PERT*, la precisión de la estimación de la duración de la actividad puede mejorarse teniendo en cuenta la cantidad de riesgo de la estimación original. Las estimaciones por tres valores se basan en determinar tres tipos de estimaciones:

- Más probable (m) - La duración de la actividad del cronograma, teniendo en cuenta: los recursos que probablemente serán asignados, su productividad, las expectativas realistas de disponibilidad para la actividad del cronograma, las dependencias de otros participantes y las interrupciones.
- Optimista (o) - La duración de la actividad se basa en el mejor escenario posible de lo que se describe en la estimación más probable.

- Pesimista (p) - La duración de la actividad se basa en el peor escenario posible de lo que se describe en la estimación más probable.

Se puede elaborar una estimación de la duración de la actividad utilizando un promedio de las tres duraciones estimadas. Este promedio con frecuencia suministra una estimación de la duración de la actividad más precisa que la estimación de valor único, más probable.

El tiempo PERT se define como: (1o + 4m + 1p) / 6

Los equipos del proyecto pueden decidir agregar tiempo adicional, denominado reservas para contingencias, *reservas de tiempo* o colchón, al cronograma del proyecto, en reconocimiento al riesgo del cronograma. La reserva de tiempo puede ser un porcentaje de la duración estimada de la actividad, una cantidad fija de períodos laborables, o puede desarrollarse mediante el análisis cuantitativo de riesgos del cronograma. La reserva de tiempo puede utilizarse de forma total o parcial, o reducirse o eliminarse con posterioridad, a medida que se dispone de información más precisa sobre el proyecto. Dicha reserva se documenta junto con otros datos y asunciones relacionados.

El **desarrollo del cronograma** del proyecto, un proceso iterativo, determina las fechas de inicio y finalización planificadas para las actividades del proyecto. El desarrollo del cronograma exige que se revisen y se corrijan las estimaciones de duración y las estimaciones de los recursos para crear un cronograma del proyecto aprobado que pueda servir como línea base (baseline) con respecto a la cual poder medir el avance. El desarrollo del cronograma continúa a lo largo del proyecto, a medida que el trabajo avanza, el plan de gestión del proyecto cambia, y los eventos de riesgo anticipados ocurren o desaparecen al tiempo que se identifican nuevos riesgos. Las entradas y salidas de este proceso son las que se muestran en la Tabla 10. Las herramientas o técnicas utilizadas para desarrollar el cronograma son Análisis de la Red del Cronograma (Diagrama Gantt), Método del Camino Crítico, Compresión del Cronograma, Análisis “¿Qué pasa si...?”, Nivelación de Recursos, Método de Cadena Crítica, Software de Gestión de Proyectos, Calendarios Aplicables y Ajuste de Adelantos y Retrasos. Veamos a continuación algunas de estas técnicas.

Entradas	Salidas
.1 Activos de los procesos de la organización .2 Enunciado del alcance del proyecto .3 Lista de actividades .4 Atributos de la actividad .5 Diagramas de red del cronograma del proyecto .6 Requisitos de recursos de las actividades .7 Calendario de recursos .8 Estimación de la duración de las actividades .9 Plan de gestión del proyecto ▪ Registro de riesgos	.1 Cronograma del proyecto .2 Datos del modelo de cronograma .3 Línea base del cronograma .4 Requisitos de recursos (actualizaciones) .5 Atributos de la actividad (actualizaciones) .6 Calendario del proyecto (actualizaciones) .7 Cambios solicitados .8 Plan de gestión del proyecto (actualizaciones) ▪ Plan de gestión del cronograma (actualizaciones)

Tabla 10 Desarrollo del Cronograma: Entradas y Salidas

El *análisis de la red del cronograma* es una técnica que genera el cronograma del proyecto. Emplea un modelo de cronograma y diversas técnicas analíticas, como por ejemplo el método del camino crítico, el método de cadena crítica, el análisis "¿Qué pasa si...?" y la nivelación de recursos, para calcular las fechas de inicio y finalización tempranas y tardías, y las fechas de inicio y finalización planificadas para las partes no completadas de las actividades del cronograma del proyecto. Si el diagrama de red del cronograma utilizado en el modelo tiene algún bucle de red o extremo abierto de la red, esos bucles y extremos abiertos se ajustarán antes de aplicar una de las técnicas analíticas. Algunos caminos de red pueden contener puntos de convergencia o divergencia de caminos que pueden identificarse y utilizarse en el análisis de compresión del cronograma o en otros análisis.

El *método del camino crítico* es una técnica de análisis de la red del cronograma que se realiza utilizando el modelo de cronograma. El método del camino crítico calcula las fechas de inicio y finalización tempranas y tardías teóricas para todas las actividades del cronograma, sin considerar las limitaciones de recursos, realizando un análisis de recorrido hacia adelante y un análisis de recorrido hacia atrás a través de los caminos de red del cronograma del proyecto. Las fechas de inicio y finalización tempranas y tardías resultantes no son necesariamente el cronograma del proyecto; en cambio, indican los períodos dentro de los cuales debería programarse la actividad del cronograma, dadas las duraciones de las actividades, las relaciones lógicas, los adelantos, los retrasos y otras restricciones conocidas.

Las fechas de inicio y finalización tempranas y tardías calculadas pueden o no ser las mismas en cualquier camino de red, dado que la holgura total, que muestra la

flexibilidad del cronograma, puede ser positiva, negativa o cero. En cualquier camino de red, la flexibilidad del cronograma se mide por la diferencia positiva entre las fechas tempranas y tardías, y se denomina "holgura total". Los caminos críticos tienen una holgura total igual a cero o negativa, y las actividades del cronograma en un camino crítico se denominan "actividades críticas". Pueden ser necesarios ajustes en las duraciones de las actividades, las relaciones lógicas, los adelantos y los retrasos, u otras restricciones del cronograma para producir caminos de red con una holgura total igual a cero o positiva. Una vez que la holgura total para un camino de red es igual a cero o positiva, también puede determinarse la holgura libre, que es la cantidad de tiempo que una actividad del cronograma puede ser demorada sin demorar la fecha de inicio temprana de cualquier actividad sucesora inmediata dentro del camino de red.

La *compresión del cronograma* acorta el cronograma del proyecto sin modificar el alcance del proyecto, para cumplir con las restricciones del cronograma, las fechas impuestas u otros objetivos del cronograma. Las técnicas de compresión del cronograma incluyen:

- Intensificación. La técnica de compresión del cronograma en la cual se analizan las concesiones de coste y cronograma para determinar cómo obtener la mayor compresión con el mínimo incremento de coste. La intensificación no siempre produce una alternativa viable y puede ocasionar un incremento de costes.
- Ejecución rápida. Una técnica de compresión del cronograma en la cual las fases o actividades que normalmente se realizarían de forma secuencial, se realizan en paralelo. La ejecución rápida puede dar como resultado un reproceso y aumento del riesgo. Esto da como resultado sacrificar coste por tiempo, y aumenta el riesgo de lograr el cronograma acortado del proyecto.

El *análisis "¿Qué pasa si...?"* es un análisis de la pregunta "¿Qué pasa si se produce la situación representada por el escenario 'X'?" Un análisis de la red del cronograma se realiza usando el modelo de cronograma para calcular diferentes escenarios, tales como la demora en la entrega de uno de los principales componentes, la ampliación de la duración de un diseño específico o la aparición de factores externos, como una huelga o un cambio en el proceso de permisos. Los resultados del análisis "Qué pasa si" pueden usarse para evaluar la viabilidad del cronograma del proyecto en condiciones adversas, y preparar los planes de contingencia y respuesta para superar o mitigar el impacto de

situaciones inesperadas. La simulación supone el cálculo de múltiples duraciones del proyecto con diferentes conjuntos de asunciones de actividades. La técnica más común es la del Análisis Monte Carlo [14], en el cual se define una distribución de posibles duraciones de las actividades para cada actividad del cronograma, y esa distribución se usa para calcular una distribución de posibles resultados para todo el proyecto.

La *nivelación de recursos* es una técnica de análisis de la red del cronograma aplicada a un modelo de cronograma que ya ha sido analizado por medio del método del camino crítico. La nivelación de recursos se usa para abordar las actividades del cronograma que deben realizarse para cumplir con fechas de entrega determinadas, para abordar situaciones en las que se dispone de recursos compartidos o críticos necesarios sólo en ciertos momentos o en cantidades limitadas, o para mantener el uso de recursos seleccionados a un nivel constante durante períodos específicos del trabajo del proyecto. Este enfoque de nivelación del uso de recursos puede hacer que cambie el camino crítico original.

La *cadena crítica* es otra técnica de análisis de la red del cronograma que modifica el cronograma del proyecto para contemplar los recursos limitados. La cadena crítica combina los enfoques determinístico y probabilístico. Inicialmente, el diagrama de red del cronograma del proyecto se construye usando estimaciones no conservadoras para las duraciones de las actividades dentro del modelo de cronograma, con las dependencias necesarias y restricciones definidas como entradas. Luego se calcula el camino crítico. Después de identificar el camino crítico, se introduce la disponibilidad de recursos y se determina el cronograma limitado por los recursos resultante. Éste, en general, tiene un camino crítico alterado. El método de cadena crítica agrega colchones de duración o reservas de tiempo que son actividades del cronograma no laborables, para mantener el enfoque en las duraciones de las actividades planificadas. Una vez que se determinan las actividades colchón del cronograma, las actividades planificadas se programan para las fechas de inicio y finalización planificadas más tardías posibles. En consecuencia, en lugar de gestionar la holgura total de los caminos de red, el método de cadena crítica se centra en gestionar las duraciones de las actividades colchón y los recursos aplicados a actividades del cronograma planificadas.

El *software de gestión de proyectos* para la elaboración de cronogramas se utiliza ampliamente para ayudar en el desarrollo del cronograma. Otras herramientas pueden

ser capaces de interactuar de forma directa o indirecta con el software de gestión de proyectos para llevar a cabo los requisitos de otras Áreas de Conocimiento. Estos productos automatizan el cálculo del análisis matemático del camino crítico de recorrido hacia adelante y hacia atrás, y la nivelación de recursos, y de esa manera, permiten la consideración rápida de muchas alternativas del cronograma. También se usan ampliamente para imprimir o mostrar en pantalla las salidas de los cronogramas desarrollados.

Los *calendarios del proyecto* y los *calendarios de recursos* identifican los períodos en que se autoriza el trabajo. Los calendarios del proyecto afectan a todas las actividades. Los calendarios de recursos afectan a un recurso específico o una categoría de recursos. Los calendarios de recursos reflejan cómo algunos recursos trabajan sólo durante las horas de trabajo normales, mientras que otros trabajan tres turnos completos, o un miembro del equipo del proyecto puede no estar disponible, por estar de vacaciones o en un programa de formación, o un contrato de trabajo puede limitar a ciertos trabajadores a trabajar durante determinados días de la semana.

Gestión del tiempo en el Proceso de Seguimiento y Control

El **Control del Cronograma** es el proceso necesario para controlar los cambios en el cronograma del proyecto. El Control del Cronograma se refiere a:

- Determinar el estado actual de la programación del proyecto
- La influencia de los factores que crean cambios de programación
- Determinar que la programación del proyecto ha cambiado
- Gestión de los cambios reales que se producen.

Las entradas y salidas de este proceso son las que se muestran en la Tabla 11. Las herramientas o técnicas utilizadas para desarrollar el cronograma son informes del progreso, el sistema de control de cambios en el cronograma, medición del rendimiento, software de gestión de proyectos, análisis de la varianza y gráficos de barras comparativas. Veamos a continuación algunas de estas técnicas.

Entradas	Salidas
.1 Plan de gestión del cronograma .2 Línea base del cronograma .3 Informes de rendimiento .4 Solicitudes de cambio aprobadas	.1 Datos del modelo de cronograma (actualizaciones) .2 Línea base del cronograma (actualizaciones) .3 Mediciones del rendimiento .4 Cambios solicitados .5 Acciones correctivas recomendadas .6 Activos de los procesos de la organización (actualizaciones) .7 Lista de actividades (actualizaciones) .8 Atributos de la actividad (actualizaciones) .9 Plan de gestión del proyecto (actualizaciones)

Tabla 11 Control del Cronograma: Entradas y Salidas

El *informe de progreso* y el estado actual de la programación incluye las fechas de inicio y fin actuales, y la duración restante de las actividades del cronograma que no están terminadas. Si la medición del progreso se hace a través del tiempo invertido, también se puede incluir el porcentaje completado de las actividades en progreso.

El *sistema de control de cambios en el cronograma* define los procedimientos por los que el cronograma del proyecto puede ser cambiado. Incluye la documentación, sistemas de seguimiento, y los niveles de aprobación necesarios para autorizar los cambios.

Las *técnicas de medición del rendimiento* producen la Varianza del Cronograma (diferencia del progreso logrado con respecto al cronograma del proyecto) y el Índice de Rendimiento del Cronograma (representa la eficiencia del avance con respecto al cronograma, y sirve para estimar la fecha de conclusión del proyecto), que se utilizan para evaluar la magnitud de cualquier variación que se produce sobre el cronograma del proyecto.

El *software de gestión de proyectos* para la programación o cronograma, permite realizar un seguimiento que compara las fechas previstas con las fechas reales, y de prever los efectos de los cambios de cronograma del proyecto.

Realizar el *análisis de variación* del cronograma durante el proceso de seguimiento del cronograma es una función clave de control de la programación. Comparar las fechas del cronograma con las fechas actuales de comienzo y fin proporciona información útil

para la detección de desviaciones, y para la aplicación de medidas correctoras en caso de retrasos.

Para facilitar el análisis del progreso del cronograma, es conveniente utilizar un *gráfico de barras de comparación*, que muestra dos barras para cada actividad del cronograma. Una barra muestra el estado real actual y el otro muestra el estado de la línea base (baseline) del proyecto. Esto muestra gráficamente dónde ha avanzado el cronograma según lo previsto, o cuándo se ha producido una disminución.

Comentarios Finales acerca del PMBOK

PMBOK es el estándar más ampliamente reconocido para manejar y administrar proyectos. Uno de los párrafos introductorios del PMBOK es: «"Buenas prácticas" no quiere decir que los conocimientos descritos deban aplicarse siempre de manera uniforme en todos los proyectos: el equipo de dirección del proyecto es el responsable de determinar lo que es apropiado para cada proyecto.» Cualquier industria o sector debe construir sus mejores prácticas específicas para su área de aplicación. Para la ingeniería del software, PMBOK define todos los procesos necesarios para una buena gestión de proyectos software, pero procesos como el control del cronograma o la planificación del alcance, requieren de más componentes para llevarlos a cabo. Por ejemplo, ¿qué consideraciones deben tenerse cuando el proceso de desarrollo es iterativo e incremental, o las actividades del proyecto siguen un workflow?

PMBOK destaca que el conocimiento para dirigir un proyecto reside en los practicantes y académicos que los aplican y los desarrollan; en otras palabras, dicho conocimiento es producto tanto de la experiencia como del estudio y del desarrollo sistemático.

Para cada uno de los procesos de las áreas de conocimiento, el PMBOK plantea o sugiere una serie de entradas, técnicas y salidas. Para los procesos relacionados con la planificación del proyecto, PMBOK propone procesos de estimación de duración de actividades y recursos, teniendo en cuenta la carga de trabajo y la disponibilidad de los recursos. También identifica mecanismos para organizar temporalmente las actividades en el cronograma, identificando la precedencia entre actividades, hitos, fases e iteraciones. Ofrece un conjunto de procesos y técnicas para realizar el seguimiento y control sobre el cronograma del proyecto y conocer las posibles desviaciones que puede haber entre los valores planificados y actuales. Las técnicas propuestas por PMBOK

para el control y seguimiento se basan en comparar estos valores tanto para los recursos como para el tiempo.

Las limitaciones del PMBOK es que esta guía resulta demasiado extensa para los proyectos pequeños, ya que existen muchos procesos, documentación o entradas/salidas que no se requieren en organizaciones pequeñas. Además, tiene que ser adaptado a la industria del área de aplicación, tamaño y alcance del proyecto, el tiempo y el presupuesto y las exigencias de calidad.

2.4. Estándar ISO/IEC 90003

La familia ISO 9000:2000 es un conjunto de normas y pautas internacionales diseñadas para apoyar la gestión de la calidad en las organizaciones. Desde su primera publicación en 1987, han conseguido convertirse como una de la normas base para el establecimiento de los sistemas de gestión de la calidad. ISO 9000:2000 agrupa las normas ISO 9001, 9002, 9003 y 9004.

ISO 9001, 9002 y 9003 son modelos de sistemas de calidad para el aseguramiento de la calidad externa. Actualmente estos modelos se han convertido en subgrupos exitosos de otros, entre ellos el ISO 9001, catalogado como el más comprensible, abarcando etapas de diseño, manufactura, instalación y sistemas de servicio. Por su parte la norma ISO 9002 se especializa en producción e instalación. La ISO 9003 enfoca su documentación en la inspección y exámenes de productos finales, y la ISO 9004 provee una guía para uso interno, permitiendo el desarrollo de sistemas de calidad propios en los negocios. La aparición en el mercado de estos modelos de procesos (centrados en la mejora) y la aplicación y adaptación de la normativa ISO 9000 como ISO 90003:2004, están ofreciendo a las empresas y departamentos de desarrollo software, la posibilidad de adaptarse a una forma de trabajo caracterizada principalmente por buscar la satisfacción de los clientes, disponer de una mejor visibilidad y control de la calidad de los procesos, y de los productos software finales.

La norma ISO/IEC 90003 [26] fue preparada por el comité técnico conjunto ISO/ IEC JTC1 (Joint Technical Committe) (tecnología de la información) subcomité SC7 (software e ingeniería de sistemas). La primera edición de ISO/IEC 90003 canceló y

sustituyó a la norma ISO/ IEC 9000-3: 1997, que se ha actualizado en conformidad con ISO 9001:2000.

Mientras que el objetivo de la norma ISO 9001 es la de establecer un sistema de calidad, la ISO/IEC 90003:2004 provee las especificaciones de cómo aplicar la ISO 9001 a los procesos de software, entre ellos los procesos de adquisición, provisión, desarrollo, operación y mantenimiento y servicios de ayuda relacionados. Además, partiendo del hecho de que es muy poco probable obtener buenos resultados en un proyecto de desarrollo de software en organizaciones de tamaño mediano, algunos temas como la Administración de la Configuración o Planificación de Proyectos que no están presentes en las normas de la familia ISO 9000, son incluidos en la ISO/IEC 90003.

ISO/IEC 90003 no impone un modelo de ciclo de vida específico. Asimismo, no provee métodos específicos para evaluar la capacidad de aseguramiento de la calidad de una organización. Por lo tanto, puede ser combinado con otros enfoques más específicos.

La Estructura del estándar ISO/IEC 90003 es:

- Ámbito
- Normas para la consulta
- Términos y definiciones
- Sistema de gestión de la calidad
- Responsabilidad de la dirección
- Gestión de los recursos
- Realización del producto
- Medición, análisis y mejora

Para cada apartado de la norma ISO 9001:2000, el ISO/IEC 90003 ofrece una descripción de la interpretación o adaptación que debe realizarse en el caso de las empresas de desarrollo de software. El punto 7 de la norma trata la Realización del Producto, que es donde se encuentran los aspectos relacionados con el desarrollo de software, y en concreto, el punto 7.1 que trata la Planificación de la realización del producto, y el 7.3 del Diseño y desarrollo. Además, el punto 8 de la norma hace referencia a la Medición, análisis y mejora de los procesos, en concreto, los relacionados con el seguimiento y control de los procesos (punto 8.2.3 del estándar). A

continuación, vamos a detallar los aspectos relacionados con la planificación y seguimiento del proyecto.

Planificación de la realización del producto (Punto 7.1 del ISO/IEC 90003)

La organización debe planificar y desarrollar los procesos necesarios para la realización del producto. La planificación de la realización del producto deberá ser coherente con los requisitos de los otros procesos del sistema de gestión de calidad.

En la planificación de la realización del producto, la organización debe determinar lo siguiente, según corresponda:

- Los objetivos de calidad y los requisitos para el producto
- La necesidad de establecer procesos, documentos y proporcionar recursos específicos para el producto
- La verificación necesaria, validación, seguimiento, inspección y prueba de actividades específicas para el producto, y los criterios para la aceptación del producto
- Los registros necesarios para demostrar que la realización de los procesos y el producto resultante cumplen los requisitos

Los procesos, actividades y tareas deben ser planificadas y realizadas utilizando modelos de ciclo de vida adecuados a la naturaleza de un proyecto de software, teniendo en cuenta el tamaño, la complejidad, seguridad, riesgo y la integridad.

La planificación del diseño y desarrollo de software debe dar lugar a una definición de qué productos se van a producir, quién los va a producir, y cuándo se van a producir. La planificación de la calidad de software a nivel de proyecto o producto, tal y como se explica a continuación, debe dar lugar a una descripción de cómo los productos específicos se van a desarrollar, evaluar o mantenerse.

Diseño y desarrollo (Punto 7.3 del ISO/IEC 90003)

La planificación del diseño y desarrollo debe estar dirigida a conseguir:

1. Las actividades de los requerimientos de análisis, diseño y desarrollo, codificación, integración, testeo, instalación y soporte para la aceptación de productos software. Para ello es necesario identificar:

- Las actividades que se llevan a cabo
- Las entradas y salidas requeridas por cada actividad
- La verificación requerida de las salidas de cada actividad
- Las actividades de gestión y soporte que se llevan a cabo
- El entrenamiento requerido del equipo

2. La planificación para el control del producto y la provisión de servicios

3. La organización de los recursos del proyecto, incluyendo la estructura del proyecto, responsabilidades, uso de proveedores, y recursos materiales que son utilizados

4. El análisis de los posibles riesgos y problemas asociados con el diseño y desarrollo

5. La programación, identificando:

    - Las etapas del proyecto
    - La estructura de división de trabajo (WBS)
    - Los recursos asociados y la medida de la duración
    - Las dependencias asociadas tanto a las actividades como a los recursos
    - Los hitos
    - Las actividades de verificación y validación

6. La identificación de:

    - Estándares, reglas, prácticas y convenciones, metodologías, modelos de ciclo de vida, etc.
    - Herramientas y técnicas para el desarrollo
    - Facilidades hardware y software para el desarrollo
    - Prácticas de gestión de la configuración
    - Métodos de control de los productos software no conformes
    - Procedimiento de recuperación y control de acceso a los productos software
    - Métodos de control para la protección antivirus
    - Controles de seguridad

7. La identificación de planificaciones relacionadas

8. El control de documentos incluyendo archivado y distribución

La ISO/IEC 90003 también indica que la planificación debe revisarse periódicamente y modificarse si es apropiado.

Seguimiento y medición de los procesos (Punto 8.2.3 del ISO/IEC 90003)

El seguimiento, medición y la mejora de los procesos deben ser identificados como parte de la planificación (incluido en punto 8 de la norma ISO/IEC 90003). Las organizaciones normalmente miden algunos aspectos de sus procesos con el fin de realizar un seguimiento, gestión y mejora sobre ellos. Las medidas más frecuentes según el estándar son:

- La duración actual y planificada de la actividad de un proceso
- El coste actual y planificado de la actividad de un proceso
- Los niveles de calidad planificados y medidas progresivas de las características seleccionadas de calidad

Comentarios Finales acerca de ISO/IEC 90003

Este estándar indica a grandes rasgos qué debe incluir la planificación de la realización de un producto, y la planificación de las etapas de diseño e implementación del ciclo de vida. En la planificación del diseño e implementación, indica los puntos que tenemos que identificar para conseguir un desarrollo de calidad. Algunos de estos puntos están relacionados con la gestión de proyectos software, como por ejemplo la definición de actividades, fases del proyecto, la estructura de división de trabajo, los recursos asociados a las actividades, dependencias entre actividades, etc. Sin embargo, no especifica cómo realizar dicha planificación, ni cómo realizar el seguimiento y control de los proyectos, únicamente se llevan a cabo. Es necesario que este estándar se aplique junto a otros estándares, guías o metodologías que complementen los aspectos básicos necesarios para la gestión de proyectos de desarrollo y mantenimiento software.

Capítulo 3. Gestión del Tiempo soportada por herramientas

A continuación vamos a comentar algunas de las herramientas estudiadas en este trabajo clasificadas en:

- Herramientas genérica para la gestión de proyectos
- Herramientas CASE con gestión de proyectos y metodología tradicional
- Herramientas para gestión de proyectos basadas en metodologías ágiles

Para cada una de las herramientas estudiadas vamos a especificar cómo abordan la gestión del tiempo considerando los siguientes aspectos:

- Cómo realizan la estimación del trabajo
- Cómo visualizan la planificación temporal del trabajo
- Si consideran la precedencia entre las actividades
- Si la ejecución del proyecto se basa en iteraciones
- Cómo se asigna y balancea el trabajo de los recursos que participan en el proyecto
- Cómo se actualizan los datos registrados del trabajo en la planificación real del proyecto
- Mediante qué técnicas se realiza el seguimiento del estado del proyecto

3.1. Herramientas genéricas para la Gestión de Proyectos (GP)

Según Letelier y Penadés (2006) [30], este enfoque centra su atención en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Se caracteriza por estar dirigido por la documentación que se genera en cada una de las actividades desarrolladas. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno volátil especialmente en cuanto a requisitos.

En este contexto metodológico, gestión de proyectos se basa en un conjunto de técnicas y procedimientos que ayudan a los jefes de proyectos definir y dirigir el trabajo del proyecto. Estas técnicas se centran en la definición de la estructura de trabajo del

proyecto, la programación y presupuestos de las actividades del proyecto, seguimiento y control de la ejecución del proyecto mientras el trabajo está siendo realizado, y evaluación y presentación de informes sobre el estado del proyecto a lo largo del ciclo de vida del proyecto. Estas técnicas han ido surgiendo a lo largo de los años:

En 1917 Henry L. Gantt desarrolla un sistema de representación gráfica donde se plasman las tareas o actividades a realizar frente a una escala de tiempo (en MS-Project, hoy se denomina escala temporal). Hoy esta forma de representación la utiliza MS-Project en su vista denominada Diagrama de Gantt.

En 1950 las empresas buscaban expertos capaces de coordinar las tareas y relacionarlas.

En 1957 Surge el CPM (Critical Path Method) o método de ruta crítica que permite calcular la duración de un proyecto en virtud de las tareas que lo componen y su encadenamiento y/o simultaneidad. Ese mismo año, el gobierno americano crea el diagrama de PERT (Program Evaluation and Review Technique) que relaciona las tareas.

En 1958, se demuestra la relación entre el diagrama de PERT y del CPM, y bajo esta relación se establece que cualquier modificación en la duración en una tarea situada la ruta crítica repercute en la modificación de la fecha de terminación del proyecto. Más tarde detallaremos que es eso de la ruta crítica.

La Tabla 12 describe las herramientas y técnicas más importantes usadas en un enfoque tradicional [41].

Técnicas/Herramientas	Propósito
Work Breakdown Structure (WBS)	Definición básica del trabajo del proyecto. Precede la planificación del proyecto y las estimaciones de costes.
Matrices de responsabilidad	Integración de la organización del proyecto con el WBS – asignación de responsabilidades.
Gráficos de barras y Diagramas de Gantt	Representación simple de la planificación del proyecto. No muestran las relaciones de precedencia entre las actividades.
Técnicas de red de proyectos	Técnicas de red para la planificación del trabajo. Proporciona el análisis del impacto que las actividades tienen entre sí y la determinación de

	las actividades críticas y camino crítico.
Programación de costes	Identificación de los requisitos de capital de los recursos. Estimación de los presupuestos realistas que proveen estándares contra los que se miden la ejecución del proyecto.
Control del proyecto: Análisis de la varianza, PERT/cost, Earned Value, y otros.	Ofrece la detección de sobrecostos del proyecto y la necesidad de medidas correctoras. El WBS, Diagramas de Gannt y otras técnicas de planificación son incorporados en el proceso de control del proyecto.

Tabla 12 Resumen de las técnicas y herramientas de gestión de proyectos

Una metodología establece las pautas a seguir dentro del desarrollo de software, indica quién (roles) debe hacer qué (artefactos), cuándo (flujos de trabajo) y cómo (actividades).

A continuación se detalla el Proceso Unificado Racional (Rational Unified Process, RUP), uno de los métodos más usados dentro de las metodologías tradicionales.

RUP es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Los orígenes de RUP se remontan al modelo espiral original de Barry Boehm. Ken Hartman, uno de los contribuidores claves de RUP colaboró con Boehm en la investigación. En 1995 Rational Software compró una compañía sueca llamada Objectory AB, fundada por Ivar Jacobson, famoso por haber incorporado los casos de uso a los métodos de desarrollo orientados a objetos. El Rational Unified Process fue el resultado de una convergencia de Rational Approach y Objectory. El primer resultado de esta fusión fue el Rational Objectory Process, la primera versión de RUP, fue puesta en el mercado en 1998, siendo el arquitecto en jefe Philippe Kruchten. Actualmente, RUP es un producto de Rational (IBM).

Este proceso de desarrollo software se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. El proceso RUP continua a lo largo de dos ejes:

- El eje horizontal representa el tiempo y se expresa en términos de fases, iteraciones e hitos.

- El eje vertical es estático y representa el proceso en términos de actividades, artefactos, roles y flujos de trabajo.

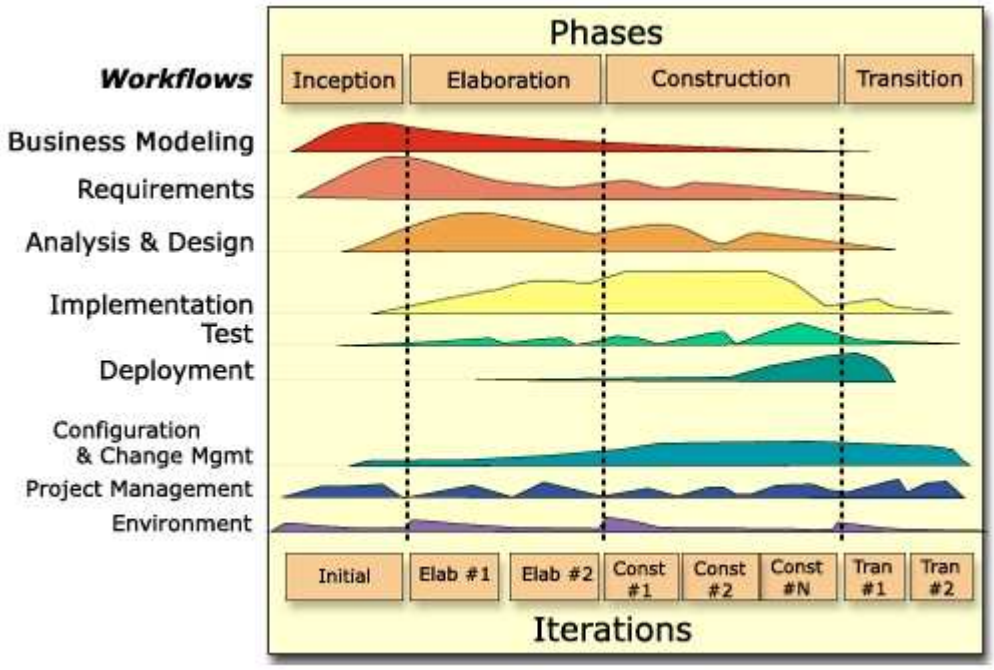


Figura 9 Representación del proceso de desarrollo RUP

La gestión del proyectos (Project Management) es una disciplina que se controla a lo largo de las 4 fases de RUP (inicial, elaboración, construcción y transición) tal y como indica la Figura 9. El objetivo de esta disciplina es gestionar, planificar y distribuir las actividades del equipo, centrándose en la calidad, programación, alcance y riesgos. La función básica de la planificación del proyecto en RUP es determinar información sobre:

- Roles
- Presupuesto
- Qué entregar y cuándo
- Asignación de recursos
- Proporcionar una base para medir el progreso y los gastos
- Línea base para realizar el seguimiento del proyecto
- Ayudar a ver lo que salió mal cuando un proyecto falla

3.1.1. MICROSOFT OFFICE PROJECT

Microsoft Office Project (MS-Project) es un programa de la suite Microsoft Office usado para la gestión de proyectos que siguen un enfoque tradicional. MS-Project es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para apoyar a los jefes de proyecto en el desarrollo de planes, asignación de recursos a tareas, realizar un seguimiento del progreso del proyecto, administrar presupuesto y analizar cargas de trabajo. Además, aplica algunos de los procedimientos descritos en la sección 2.3 PMBOK de este trabajo.

Los pasos a seguir en la planificación para que el proyecto se desarrolle sin problemas, pueden ser los siguientes:

- Definir la lista de tareas e hitos.
- Estimar la duración de cada tarea.
- Determinar las relaciones entre las tareas.
- Construir la estructura de descomposición de trabajo (WBS).
- Evaluación de los recursos del proyecto.
- Asignación de recursos a las tareas.
- Optimizar la estructura.

Una vez el trabajo del proyecto está desglosado en tareas, MS-Project permite calcular una programación realista basándose en la duración de las tareas, las dependencias entre ellas y las delimitaciones. Primero propone comenzar recopilando información básica que ayude a estimar cuánto tardarán las tareas en completarse, mediante:

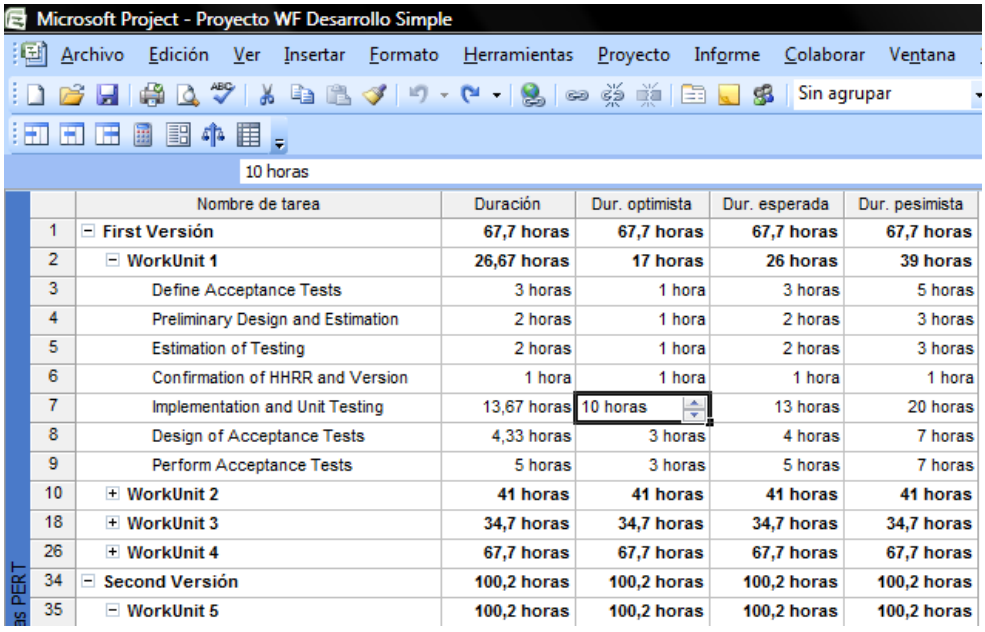
- Propia experiencia
- Experiencia de los miembros del equipo en tareas similares
- Proyectos anteriores
- Estándares y referencias del sector

Dado que no es posible predecir los problemas que puedan retrasar las tareas, MS-Project propone añadir un poco de "margen" (búfer de tiempo) a las mismas, como medida de la administración de riesgos del proyecto. Microsoft Office [32] indica algunas formas de generar un búfer de tiempo para el proyecto:

- Agregar un porcentaje a cada duración. Por ejemplo, si la realización de una tarea lleva 40 horas, la adición de un búfer del 5% la cambia a 42 horas. Un búfer del 10% la cambia a 44 horas.
- Agregar una tarea de búfer que contenga un margen de tiempo extra en la planificación. Por ejemplo, si se necesita un día adicional para una tarea que se está retrasando, se puede eliminar la tarea de búfer y agregar su tiempo a la tarea que se está retrasando. De este modo, la fecha de finalización de la fase o del proyecto permanecerá inalterada.

En segundo lugar, es necesario crear un hito. Actúa como punto de referencia que marca un evento importante de un proyecto y se utiliza para supervisar el estado del proyecto.

En tercer lugar propone estimar las duraciones de la programación y simular las futuras cargas de recursos y su efecto sobre las escalas de tiempo del proyecto. El análisis PERT permite analizar el mejor y el peor escenario a fin de estimar de forma precisa. Utiliza las duraciones optimista, pesimista y esperada de las tareas para calcular un promedio ponderado de las tres duraciones o bien, las considera por separado. La Figura 10 muestra un ejemplo en el que se han estimado las actividades/tareas necesarias para desarrollar una unidad de trabajo que está asignada a una versión. El análisis PERT que realiza la herramienta utiliza la información introducida por el usuario en los campos Dur. optimista, Dur. esperada y Dur. pesimista. Con toda esta información, Project calcula la fecha de inicio y finalización de cada una de las tareas.



	Nombre de tarea	Duración	Dur. optimista	Dur. esperada	Dur. pesimista
1	**First Versión**	**67,7 horas**	**67,7 horas**	**67,7 horas**	**67,7 horas**
2	**WorkUnit 1**	**26,67 horas**	**17 horas**	**26 horas**	**39 horas**
3	Define Acceptance Tests	3 horas	1 hora	3 horas	5 horas
4	Preliminary Design and Estimation	2 horas	1 hora	2 horas	3 horas
5	Estimation of Testing	2 horas	1 hora	2 horas	3 horas
6	Confirmation of HHRR and Version	1 hora	1 hora	1 hora	1 hora
7	Implementation and Unit Testing	13,67 horas	10 horas	13 horas	20 horas
8	Design of Acceptance Tests	4,33 horas	3 horas	4 horas	7 horas
9	Perform Acceptance Tests	5 horas	3 horas	5 horas	7 horas
10	**WorkUnit 2**	**41 horas**	**41 horas**	**41 horas**	**41 horas**
18	**WorkUnit 3**	**34,7 horas**	**34,7 horas**	**34,7 horas**	**34,7 horas**
26	**WorkUnit 4**	**67,7 horas**	**67,7 horas**	**67,7 horas**	**67,7 horas**
34	**Second Versión**	**100,2 horas**	**100,2 horas**	**100,2 horas**	**100,2 horas**
35	**WorkUnit 5**	**100,2 horas**	**100,2 horas**	**100,2 horas**	**100,2 horas**

Figura 10 Estimación de las tareas en Microsoft Office Project

Para determinar las relaciones entre las tareas, MS-Project ofrece un mecanismo visual (ver el Diagrama de Gantt de la Figura 11) para facilitar la comprensión de la secuenciación de las tareas definidas y estimadas anteriormente.

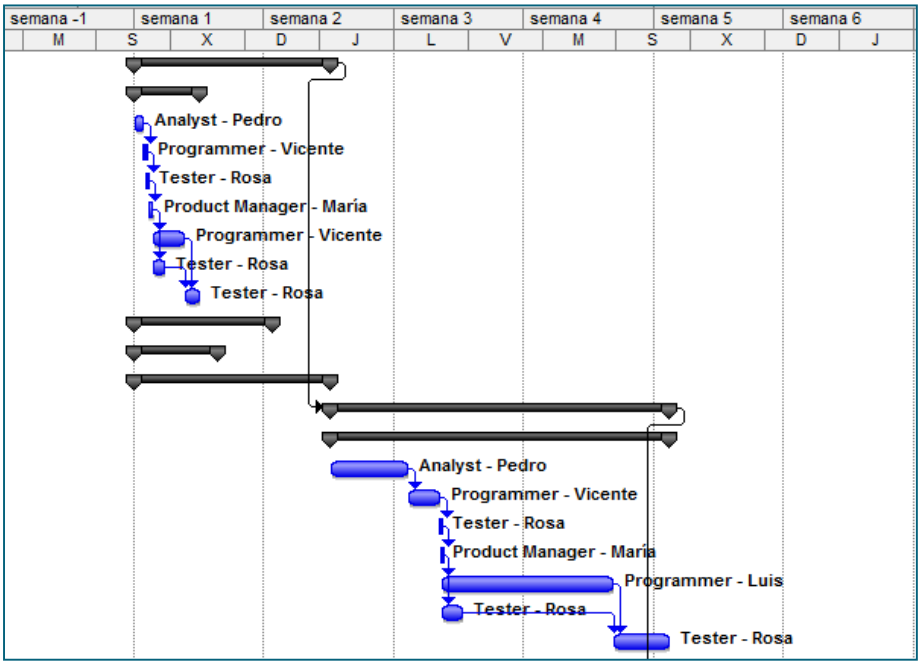
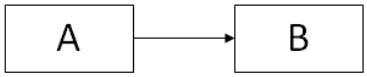
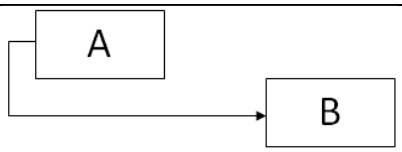
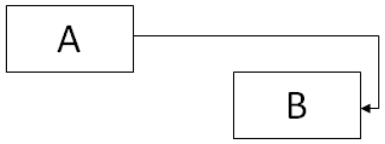


Figura 11 Diagrama de Gantt de las tareas

Las tareas se tienen que ejecutar en un determinado orden. Las tareas de un proyecto tienen que estar vinculadas entre ellas, y sus vínculos establecen sus relaciones de orden. MS-Project establece cuatro posibles tipos de vínculos ya vistos en la sección 2.3 PMBOK de este trabajo (Fin A Comienzo, Comienzo a Comienzo, Fin a Fin, Comienzo a Fin)

La Tabla 13 muestra la representación de cada uno de estos vínculos:

Tipo de vínculo	Representación
Fin A Comienzo	
Comienzo a Comienzo	
Fin a Fin	

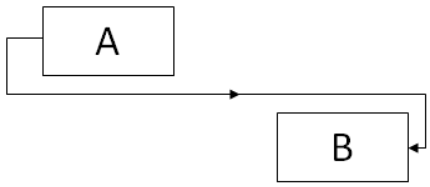
Comienzo a Fin	

Tabla 13 Representación de los vínculos entre tareas

MS-Project permite crear recursos, incluirlos en proyectos, y asignarlos a las tareas para indicar qué recurso es responsable de completar cada asignación, lo cual ayuda a calcular el número de recursos que se utilizarán en el proyecto.

Un recurso se suele definir como cualquiera de las personas, del equipo y de los materiales utilizados para completar las tareas de las que se compone un proyecto. La Figura 12 contiene toda la información de los recursos asociados a un proyecto. El campo Tipo especifica si se trata de un recurso de trabajo, material o costo. El campo Capacidad máxima corresponde con el número de unidades de disponibilidad máxima del recurso en el proyecto. Por ejemplo, si tenemos un recurso disponible dos días por semana para el proyecto, podemos especificar como capacidad máxima el 40%. O bien, supongamos que contamos con un recurso denominado Ingenieros, un solo recurso que representa a tres ingenieros individuales del equipo y cuya capacidad la podemos especificar con el 300%.

Nombre del recurso	Tipo	Iniciales	Capacidad máxima	Tasa estándar	Tasa horas extra	Costo/Uso	Acumular	Calendario base
Product Manager -	Trabajo	P	100%	20,00 €/hora	50,00 €/hora	0,00 €	Prorrateo	Estándar
Analyst - Pedro	Trabajo	A	100%	10,00 €/hora	30,00 €/hora	0,00 €	Prorrateo	Estándar
Programmer - Vice	Trabajo	P	100%	10,00 €/hora	30,00 €/hora	0,00 €	Prorrateo	Estándar
Tester - Rosa	Trabajo	T	100%	10,00 €/hora	30,00 €/hora	0,00 €	Prorrateo	Estándar
Programmer - Luis	Trabajo	P	100%	10,00 €/hora	30,00 €/hora	0,00 €	Prorrateo	Estándar

Figura 12 Hoja de Recursos de un proyecto

Para asignar los recursos a las tareas, sólo debemos de considerar aquellos recursos que disponen del tiempo suficiente para completar la tarea. La Figura 13 corresponde con la ventana de asignación de recursos a las tareas. En este ejemplo, filtra los recursos que disponen, como máximo, de 1 hora para trabajar. El recurso Luis tiene asignado 50% de dedicación, y el recurso Rosa tiene una dedicación del 100%.

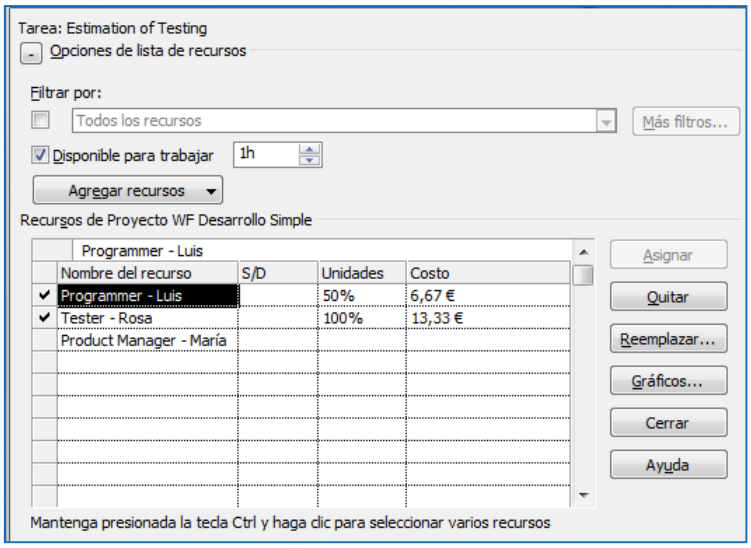


Figura 13 Asignación de recursos a las tareas

Para cada recurso asignado a una tarea se incluye información del trabajo (cantidad de trabajo que se ha asignado al recurso), de las unidades o coste según el tipo de recurso (cantidad de un recurso que se asigna a la tarea, 100% equivale a tiempo completo del recurso) y periodos de trabajo (horas designadas en un calendario de recursos durante las que se puede realizar trabajo).

Para el jefe de proyecto, uno de los aspectos más importantes de su función es la supervisión de las asignaciones de cada uno de los recursos, para que pueda equilibrar de forma eficaz sus cargas de trabajo. Es posible que algunos recursos se encuentren sobreasignados mientras que otros pueden estar infraasignados. MS-Project permite revisar la eficacia con la que se están usando los recursos en el proyecto y si se necesita realizar algún ajuste viendo sus cargas de trabajo y disponibilidad.

La característica de redistribución de recursos permite resolver conflictos entre recursos y sobreasignaciones. Esta redistribución funciona dividiendo las tareas o agregando un retraso a las tareas hasta que los recursos (de trabajo, materiales y costo) asignados a dichas tareas ya no estén sobreasignados. A causa de estos cambios en las tareas, se puede retrasar la fecha de fin de algunas tareas y, en consecuencia, la fecha de fin del proyecto.

Cuando un recurso está sobreasignado (se excede su capacidad máxima), MS-Project resalta su nombre en rojo desde cualquier vista de recursos, como por ejemplo la vista mostrada en la Figura 14.

Nombre del recurso	% mpleta	Trabajo	Horas extra	Trabajo previsto	Real	Restante	Detalles	08 jun '09			
								L	M	X	J
Product Manager - María	14%	3,6 horas	0 horas	3 horas	0,5 horas	3,1 horas	Trabajo	0,3h			
Analyst - Pedro	24%	152 horas	8 horas	156 horas	36 horas	116 horas	Trabajo				
Programmer - Vicente	8%	522,97 horas	2 horas	535 horas	44 horas	478,97 horas	Trabajo	23h	16h	15,6h	8h
Tester - Rosa	11%	178,83 horas	1 hora	180 horas	20 horas	158,83 horas	Trabajo	16,7h	7,3h	0,4h	3,6h
Programmer - Luis	10%	110 horas	0 horas	110 horas	11 horas	99 horas	Trabajo	7,9h	8h	8h	8h
Implementation and Unit	*5%*	*60 horas*	*0 horas*	*60 horas*	*3 horas*	*57 horas*	Trabajo				
Implementation and Unit	*20%*	*15 horas*	*0 horas*	*15 horas*	*3 horas*	*12 horas*	Trabajo				
Implementation and Unit	*14%*	*35 horas*	*0 horas*	*35 horas*	*5 horas*	*30 horas*	Trabajo	7,9h	8h	8h	8h

Figura 14 Vista de Uso de Recursos

La vista Uso de recursos anterior muestra los recursos del proyecto y, agrupadas debajo de ellos, las tareas que tienen asignadas. Esta vista permite conocer cuántas horas de trabajo tiene programadas cada recurso en tareas específicas (Trabajo = Trabajo restante + Trabajo real), el porcentaje completado (Porcentaje de trabajo completado = Trabajo real / Trabajo), las horas extra, el tiempo real dedicado a la tarea, el tiempo restante en finalizar la tarea, etc. También permite determinar el tiempo disponible de cada recurso para asignarle trabajo adicional.

Antes de redistribuir los recursos, MS-Project recomienda establecer las prioridades de las tareas. Se trata de una indicación de la importancia de una tarea y su disponibilidad para la redistribución. El valor de prioridad que se asigne a la tarea es un valor subjetivo comprendido entre 1 y 1000, lo que permite especificar el nivel de control que tiene sobre el proceso de redistribución. Por ejemplo, si no se desea que MS-Project redistribuya una tarea en particular, se debe establecer su nivel de prioridad en 1000. De manera predeterminada, los valores de prioridad se establecen en 500, es decir, un nivel medio de control. Las tareas que tienen una prioridad inferior se retrasan o dividen antes que las que tienen un nivel más alto.

MS-Project, para controlar el estado del proyecto, compara la información del mismo a lo largo del tiempo mediante las estimaciones originales del proyecto y los datos reales. Para ello propone:

- Crear un plan provisional con la programación actualizada y compararlo con el plan de previsión.
- Conservar versiones diferentes de un proyecto.
- Comparar dos versiones de un mismo proyecto.

Los datos reales de la duración de las tareas se recopilan manualmente en la información general de cada tarea (Figura 15). La herramienta permite introducir el

porcentaje completado de la tarea, introducir las fechas de comienzo y fin, o bien, modificar el valor de su duración.

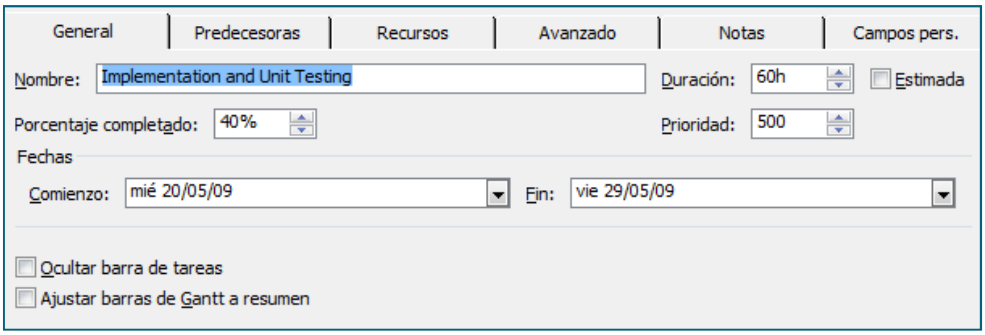


Figura 15 Información de la tarea

Para recopilar estos datos se recomienda:

- Decidir sobre qué información del proyecto se va a realizar el seguimiento. Por ejemplo, las fechas de inicio y fin de las tareas, los porcentajes de completado de las tareas y costos, etc.
- Recopilar datos reales para actualizar la información.
- Decidir el método de recopilación de datos. La herramienta propone recopilar los datos por teléfono, entrevistando a las fuentes o rellenando formularios.
- Elegir una frecuencia de recopilación de datos.

3.2. Herramientas CASE con GP y metodología tradicional

En las últimas décadas se ha trabajado en el área de desarrollo de sistemas para encontrar técnicas que permitan incrementar la productividad y el control de calidad en cualquier proceso de elaboración se software, y hoy en día la tecnología CASE, (Computer Aided Software Engineering) transforma la actividad de desarrollar software en un proceso automatizado, es decir, tecnología que permita utilizar o apoyar una o más fases del ciclo de vida del desarrollo del software.

Los objetivos principales de las herramientas CASE son [23]:

- Permitir la aplicación práctica de metodologías, lo que resulta muy difícil sin emplear herramientas.
- Facilitar la realización de prototipos y el desarrollo conjunto de aplicaciones.
- Simplificar el mantenimiento del software.

- Mejorar y estandarizar la documentación.
- Aumentar la portabilidad de las aplicaciones.
- Facilitar la reutilización de componentes de software.
- Permitir un desarrollo y un refinamiento -visual- de las aplicaciones, mediante la utilización de controles gráficos (piezas de código reutilizables).

Las herramientas CASE orientadas a la gestión de proyectos buscan facilitar la coordinación y cooperación entre los miembros del equipo de trabajo, los cuales necesitan compartir información, realizar tareas independientes, etc. Vessey y Sravanapudi (1995) [44] sugieren que estas herramientas deben ofrecer facilidades para compartir la información (método sencillo de comunicación), proporcionar un seguimiento de las capacidades, control, gestión del tiempo, etc. Desde el punto de vista del seguimiento, muchos autores destacan la necesidad de controlar tanto los aspectos del producto como de los usuarios, como por ejemplo las autenticaciones de los usuarios (Charette, 1986) [7] o mantener los cambios hechos en el diseño (Henderson y Cooprider, 1990) [15]. Desde el punto de vista de la gestión del tiempo, otros autores [44] piensan que las herramientas CASE deberían incluir un calendario con la planificación personal del tiempo.

No es frecuente encontrar herramientas CASE asociadas a gestión de proyectos, pero hemos incluido el estudio de Enterprise Architect como un ejemplo para ilustrar este ámbito.

3.2.1. ENTERPRISE ARCHITECT (EA)

Enterprise Architect es una herramienta CASE, que abarca todos los aspectos del ciclo de desarrollo de software. Se trata de una herramienta colaborativa, visual, y con un gran conjunto de características que ayudan a los analistas, testers, jefes de proyectos, etc., a construir y documentar de forma robusta, y facilitar el mantenimiento de los sistemas y procesos. Proporciona un soporte básico en la gestión de proyectos, permitiendo estimar el tamaño del proyecto, medir el riesgo y el esfuerzo, asignar recursos, y aplicar el control de cambios y procedimientos de mantenimiento.

La estimación del proyecto se elabora en base al tiempo y esfuerzo que se requiere para construir y desplegar la solución, mediante la técnica Puntos de Casos de Uso. El modelo Puntos de Casos de Uso (Use Case Points) (Karner, 1993) [28] está inspirado en

el modelo de estimación Puntos de Función. Primero se mide la funcionalidad del sistema basándose en el modelo de casos de uso mediante le recuento de los Puntos de Casos de Uso Sin Ajustar (Unadjusted Use Case Point, UUCP). Después se evalúa el Factor de Complejidad Técnica (Technical Complexity Factor, TCF) involucrado en el desarrollo de esta funcionalidad, y por último, se calcula el Factor del entorno (Environmental Factor, EF) propuesto por Karner. Los Puntos de Casos de Uso (UCP) son el producto de estos tres factores, y como resultado, obtiene una estimación del esfuerzo en desarrollar el sistema medido en horas-hombre.

Puntos de Casos de Uso Sin Ajustar (UUCP)

Para calcular los UUCP se suman los pesos de los actores y casos de uso que aparecen en el modelo de Casos de Uso, de acuerdo a los pesos que Karner propone en la Tabla 14 y Tabla 15. La siguiente fórmula permite calcular los UUCP, donde n_i es el número de ítems de variedad i (número de casos de uso o actores) y W_i es el peso asociado a esa variedad.

$$UUCP = \sum_{i=1}^{6} n_i * W_i$$

Complejidad	Definición	Peso
SIMPLE	El actor representa otro sistema con una interfaz de programación de aplicaciones definidas.	1
AVERAGE	El actor es una interacción con otro sistema a través de un protocolo, o una interacción humana con un terminal de línea.	2
COMPLEX	El actor interactúa a través de una interfaz gráfica de usuario.	3

Tabla 14 Pesos de los Actores

Complejidad	Definición	Peso
SIMPLE	El caso de uso tiene 3 o menos transacciones[1].	5
AVERAGE	El caso de uso tiene de 3 a 7 transacciones.	10
COMPLEX	El caso de uso tiene más de 7 transacciones.	15

Tabla 15 Pesos de Casos de Uso

[1] A los casos de uso se les asigna una complejidad basada en transacciones, que son pares de pasos acción-usuario o respuesta-sistema de los escenarios de los casos de uso.

Factor de Complejidad Técnica (TCF)

La siguiente fórmula permite calcular el TCF, donde F_i es un factor que oscila entre una escala de 0, 1, 2, 3, 4 y 5 (0 significa que es irrelevante y 5 que es esencial) (ver Tabla 16). Estos factores hacen referencia a los elementos que tienen un impacto en la complejidad del sistema en términos de su desarrollo.

$$TCF = C_1 + C_2 \sum_{i=1}^{13} F_i * W_i \qquad C_1 = 0.6 \qquad C_2 = 0.01$$

F_i	**Factores que contribuyen a la complejidad**	W_i
F_1	Sistemas Distribuidos	2
F_2	Los objetivos de rendimiento de las aplicaciones, ya sea en respuesta o el rendimiento	1
F_3	Eficiencia del Usuario Final (online)	1
F_4	Complejidad del procesamiento interno	1
F_5	Reusabilidad, el código debería ser capaz de reutilizarse en otras aplicaciones	1
F_6	Fácil de instalar	0.5
F_7	Facilidad de uso	0.5
F_8	Portabilidad	2
F_9	Variabilidad	1
F_{10}	Concurrencia	1
F_{11}	Características especiales de seguridad	1
F_{12}	Facilitar el acceso directo a terceros	1
F_{13}	Medios especiales de formación de usuarios	1

Tabla 16 Factores que contribuyen a la complejidad

Factor del entorno (EF)

La siguiente formula permite calcular el EF, donde F_i es un factor que oscila entre una escala de 0, 1, 2, 3, 4 y 5 (0 significa que es irrelevante y 5 que es esencial) (ver Tabla 17). El EF ayuda a estimar o determinar cómo de eficiente es el proyecto.

$$EF = C_1 + C_2 \sum_{i=1}^{8} F_i * W_i \qquad C_1 = 1.4 \qquad C_2 = -0.03$$

F_i	Factores que contribuyen a la eficiencia	W_i
F_1	Familiarizado con Objectory	1.5
F_2	Trabajadores a tiempo parcial	-1
F_3	Capacidad de los analistas	0.5
F_4	Experiencia en la aplicación	0.5
F_5	Experiencia en orientación a objetos	1
F_6	Motivación	1
F_7	Dificultad en el lenguaje de programación	-1
F_8	Requisitos estables	2

Tabla 17 Factores que contribuyen a la eficiencia

Puntos de casos de uso (UCP)

Finalmente los UCP se calculan mediante el producto de los resultados obtenidos anteriormente tal y como indica la siguiente fórmula. Este cálculo permite ver los recursos necesarios por UCP mediante las horas-hombre que se necesitan para finalizar el proyecto.

$$UCP = UUCP * TCF * EF$$

Como se ha comentado anteriormente, la herramienta de gestión de proyectos software Enterprise Architect, utiliza la técnica de estimación Puntos de Casos de Uso basada en el número de casos de uso para la construcción del sistema, en el nivel de dificultad de los casos de uso, en los factores del entorno de los proyectos y en algunos parámetros relativos a la construcción del sistema.

La Figura 16 muestra la ventana llamada Use Case Metrics que permite conocer la estimación total del proyecto a través de los cálculos basados en los Puntos de Casos de Uso. Además de estos cálculos, incluye el promedio de horas por caso de uso (Ave Hours per Use Case), la estimación del esfuerzo de trabajo (Estimated Work Effort, EWE) (horas de trabajo de un UCP * el número de UCP), el coste estimado (Estimated Cost) (EWE * coste por hora).

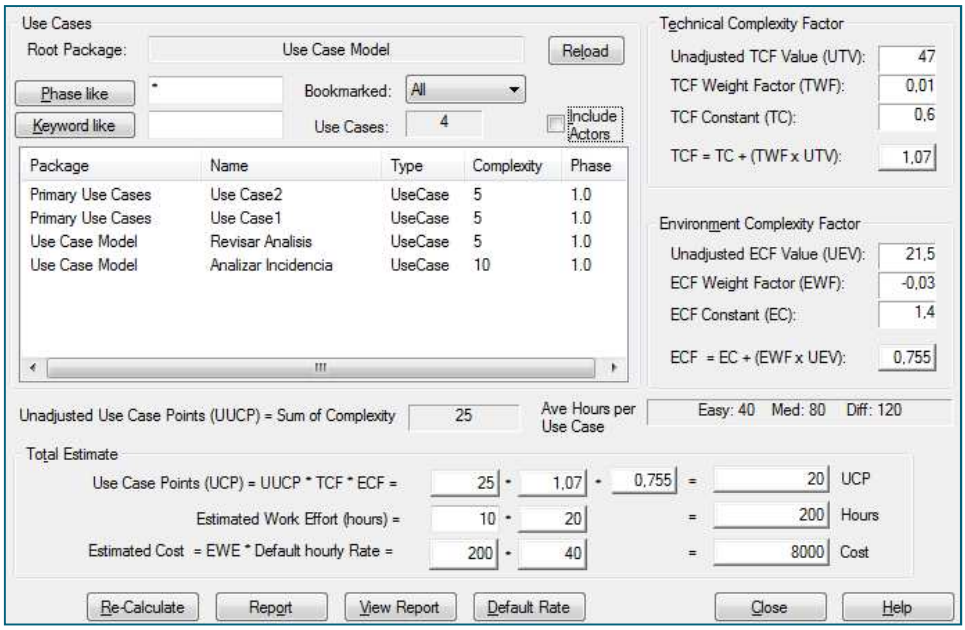


Figura 16 Ventana Use Case Metrics

Para la herramienta EA un recurso hace referencia a las personas que trabajan en el proyecto. Los recursos pueden ser asignados a roles o tareas para llevar un seguimiento del esfuerzo y la estimación de tiempo a completar. Este seguimiento se realiza a través del módulo de gestión de recursos del proyecto que ofrece la herramienta, el cual está formado por las pestañas Resource Allocation, Effort, Risks y Metrics que contiene la Figura 17. Este módulo permite añadir recursos, esfuerzos, riesgos y métricas que pueden ser añadidas a los elementos contenidos en el modelo. La Figura 17 resume para cada recurso el rol o tarea que tiene asignada, el tiempo asignado al recurso, el porcentaje completado, y las fechas de inicio y fin.

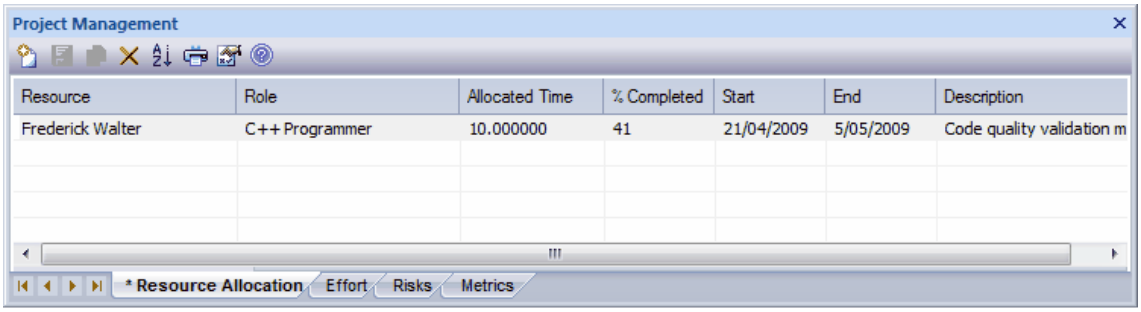


Figura 17 Ventana de gestión de proyectos de EA

A continuación vamos a detallar la funcionalidad de cada una de las pestañas de la venta de Gestión de Proyectos:

La pestaña de Asignación de Recursos (Resource Allocation) de la Figura 18, permite asignar un recurso a un rol o tarea mediante la siguiente información:

- Nombre del recurso
- Rol del recurso
- La fecha de comienzo y fin de la disponibilidad del recurso
- Tiempo asignado al recurso
- Porcentaje de la tarea que el recurso ha completado
- Tiempo esperado asignado al recurso
- Porcentaje de la tarea que el recurso ha completado
- Tiempo real dedicado por el recurso (se actualiza manualmente)
- Descripción del trabajo que está siendo hecho por el recurso

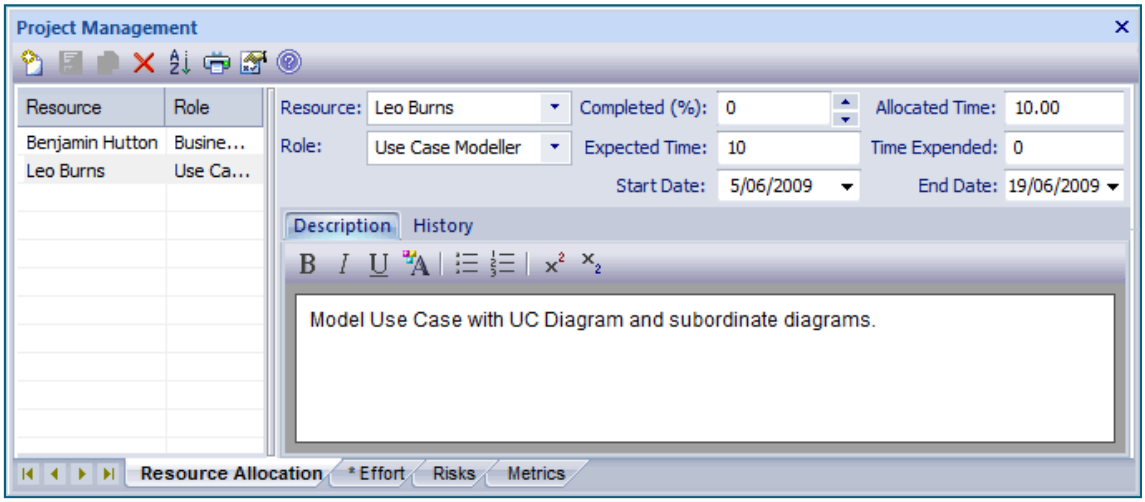


Figura 18 Gestión de proyectos – Asignación de recursos

La pestaña Esfuerzo (Effort) de la Figura 19 permite crear esfuerzos de tipos de esfuerzos predefinidos (análisis, codificación, construcción, diseño, elaboración, transición, etc.), y asignarlos a elementos del modelo. Además, los esfuerzos tienen asociado un tiempo que indica, el tiempo adicional que gastará el elemento al que se asigna el esfuerzo.

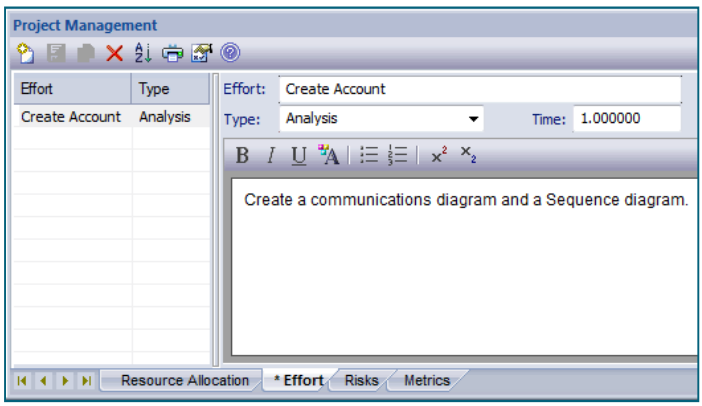


Figura 19 Gestión de proyectos – Esfuerzo

La pestaña Riesgos (Risks) de la Figura 20 permite crear riesgos de tipos de riesgos predefinidos (entregas, proveedores, etc.), y asignarlos a elementos del modelo. Además, los riesgos tienen asociado un peso que indica, el peso adicional asignado al elemento al que se asigna el riesgo.

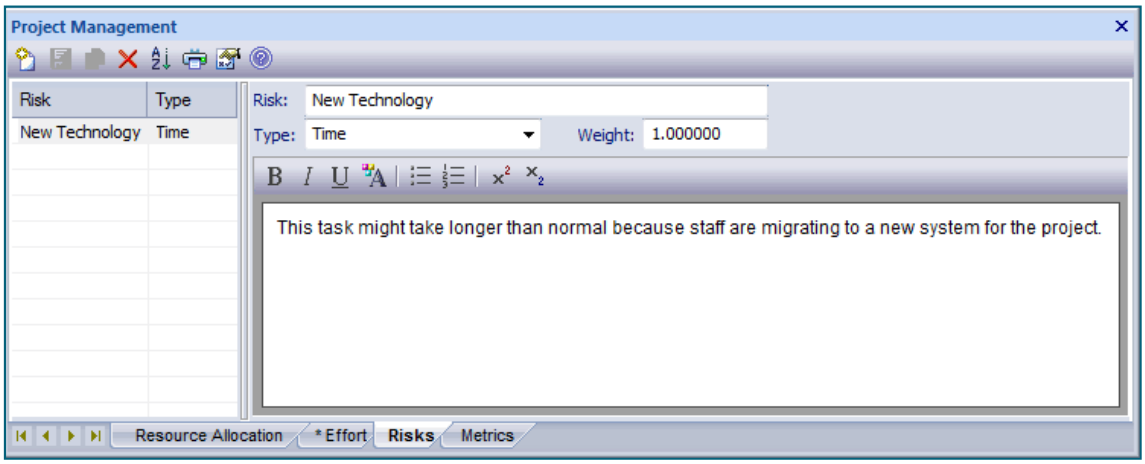


Figura 20 Gestión de proyectos – Riesgo

La pestaña Métricas (Metrics) de la Figura 21 permite crear métricas de tipos de métricas predefinidas (retrabajo, cambios, coste, progreso, equipo, etc.), y asignarlas a elementos del modelo. Además, las métricas tienen asociado un peso que indica, el peso adicional asignado al elemento al que se asigna la métrica.

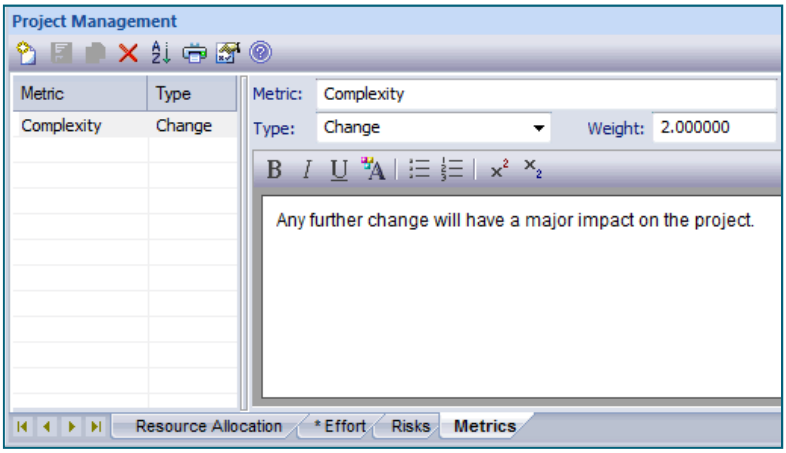


Figura 21 Gestión de proyectos – Métricas

Toda esta información está resumida en el informe de recursos de la Figura 22, el cual muestra una lista de todos los elementos que tienen recursos asignados a ellos. Además, la lista incluye el recurso asignado, la fecha de inicio y fin, el porcentaje completado y otra información relevante. La ficha superior ofrece una serie de filtros para filtrar los recursos, y además, mediante el botón Localizar Objeto (Locate Object) podemos

buscar los elementos del modelo que tienen asignados los resultados seleccionados de la lista de recursos de la figura.

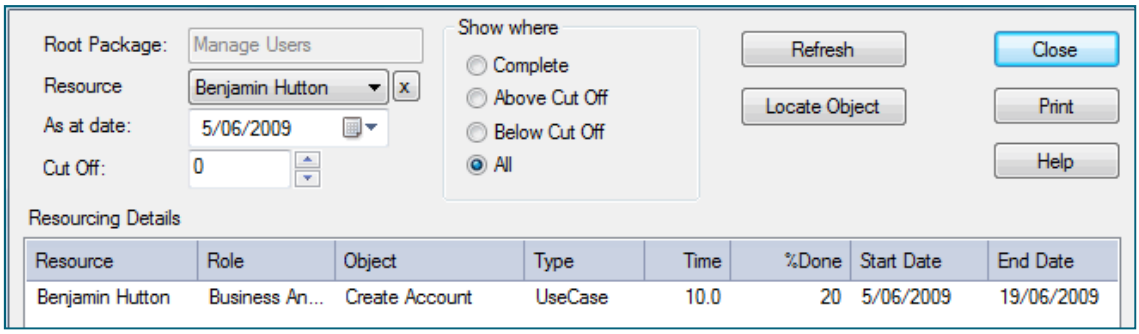


Figura 22 Informe resumen de los recursos

La Lista de Tareas del Proyecto (Project Tasks List) es una lista 'To Do' con los ítems de trabajo relevantes del proyecto y que no se guardan en ningún lugar del mismo. Por ejemplo, tareas que siempre se hacen pero no están reflejadas en ningún diagrama, como las peticiones o reuniones. Una tarea está formada por un nombre, tipo de tarea (defecto, petición, reunión, etc.), propietario de la tarea, fecha de inicio y fin prevista, estado actual, persona asignada a la tarea, prioridad (alta, media o baja), tiempo total previsto en completar la tarea y el tiempo actual invertido, porcentaje completado y fase del proyecto en la que se realiza la tarea (ver Figura 23).

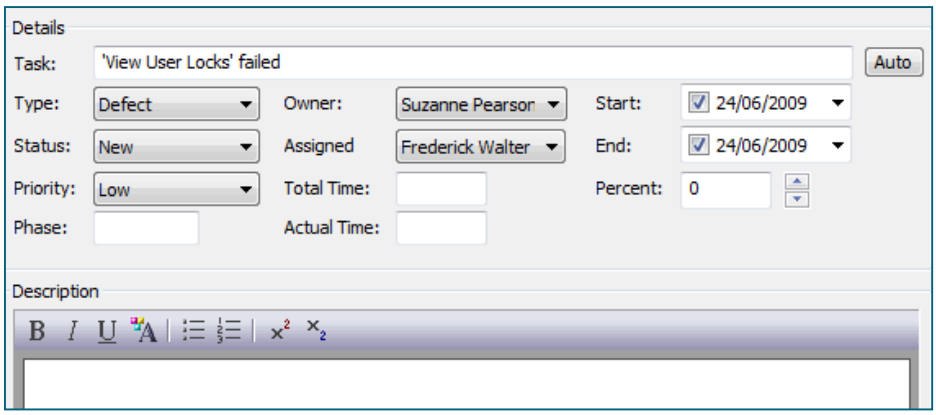


Figura 23 Ventana de creación de Tareas del Proyecto

3.3. Herramientas para GP basadas en metodologías ágiles

Tal y como se ha comentado en apartados anteriores, las metodologías tradicionales hacen énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada. Este enfoque ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo

y recursos), donde por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad (Letelier y Penadés, 2006) [30]. Ante las dificultades para utilizar metodologías tradicionales con estas restricciones de tiempo y flexibilidad, las metodologías ágiles emergen como una posible respuesta a estas restricciones.

A continuación se numeran las características que diferencian un proceso ágil de uno tradicional.

- La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
- Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
- Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
- La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
- Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
- El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
- El software que funciona es la medida principal de progreso.
- Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
- La atención continua a la calidad técnica y al buen diseño mejora la agilidad. Producir código claro y robusto es la clave para avanzar más rápidamente en el proyecto.
- La simplicidad es esencial. Tomar los caminos más simples que sean consistentes con los objetivos perseguidos. Si el código producido es simple y de alta calidad será más sencillo adaptarlo a los cambios que puedan surgir.
- Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.

- En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

Las principales diferencias de una metodología ágil con respecto de las metodologías tradicionales, tanto a nivel de proceso como del contexto del equipo y organización, están recogidas en la Tabla 18 Diferencias entre metodologías ágiles y no ágiles que ofrecen [30].

Metodología Ágil	Metodología Tradicional
Pocos artefactos. El modelado es esencial, modelos desechables	Más artefactos. El modelado es esencial, mantenimiento de modelos
Pocos roles, más genéricos y flexibles	Más roles, más específicos
No existe un contrato tradicional, debe ser bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo (además in-situ)	El cliente interactúa con el equipo de desarrollo mediante reuniones
Orientada a proyectos pequeños. Corta duración (o entregas frecuentes), equipos pequeños (< 10 integrantes) y trabajando en el mismo sitio	Aplicables a proyectos de cualquier tamaño, pero suelen ser especialmente efectivas/usadas en proyectos grandes y con equipos posiblemente dispersos
La arquitectura se define y mejora a lo largo del proyecto	Se promueve que la arquitectura se defina tempranamente en el proyecto
Énfasis en los aspectos humanos: el individuo y el trabajo en equipo	Énfasis en la definición del proceso: roles, actividades y artefactos
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Se esperan cambios durante el proyecto	Se espera que no ocurran cambios de gran impacto durante el proyecto

Tabla 18 Diferencias entre metodologías ágiles y no ágiles

De acuerdo con Boehm, las organizaciones deben evolucionar, cuidadosamente, hacia el mejor balance entre los métodos ágiles y los dirigidos por un plan que mejor se adapten a su situación. Las herramientas de gestión de proyectos que siguen un enfoque ágil deben ofrecer un balance entre la gestión del control y la colaboración y liderazgo. El papel del jefe de proyecto tradicional sobre la planificación y seguimiento de proyectos debe ser modificado por el papel del "facilitador" (Nerur et al., 2005) [33], el cual dirige y coordina la colaboración de los miembros del equipo, asegurando de que participen en cualquier decisión final. Las herramientas ágiles de gestión de proyectos no generan documentación, sino que fomentan el "lean thinking" para eliminar los procesos que no

reportan valor al producto manufacturado. Además, las metodologías ágiles se basan en la especulación, con lo cual, la planificación se hace en base a que todo es incierto dando más importancia a la evaluación que a la medición (Highsmith, 2003) [17].

Los métodos ágiles de desarrollo de software difieren en gran medida en la forma en que cubren la gestión de proyectos. La Figura 28 muestra diferentes metodologías ágiles y las actividades de desarrollo básicas. Para cada metodología se indica mediante barras las actividades del ciclo de vida que cubren (barra intermedia), las actividades cubiertas por la gestión de proyectos (barra superior) y aquellas que ofrecen una orientación concreta (barra inferior). (Abrahamssona et al., 2003) [1].

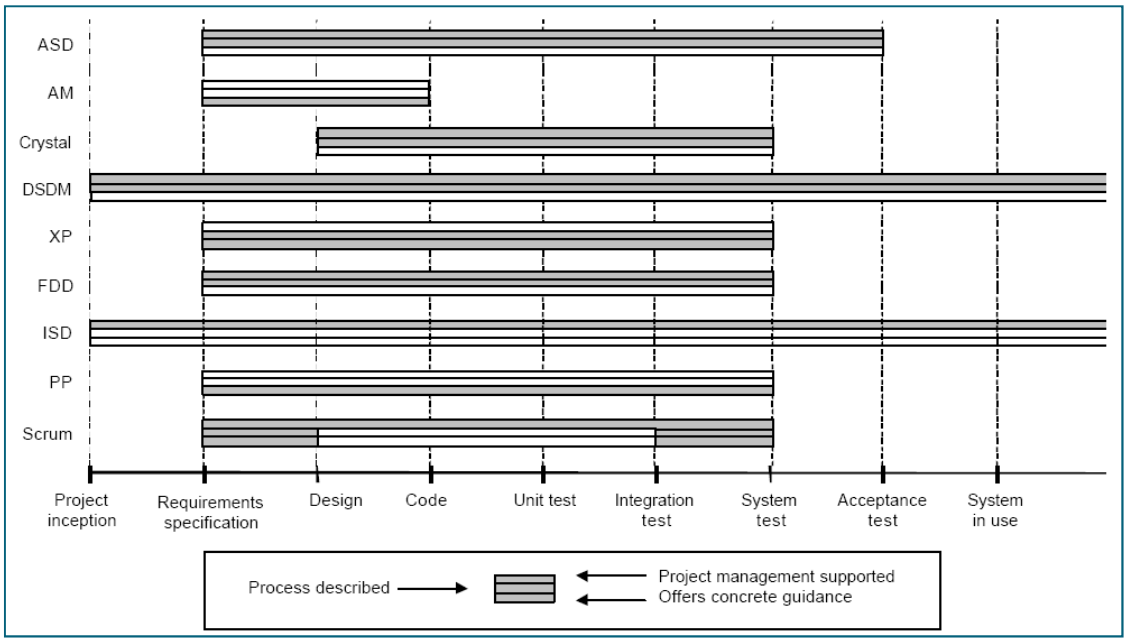


Figura 24 Comparando el ciclo de vida, gestión de proyecto y orientación concreta

Metodologías como Agile Modeling (AM) (Ambler, 2002) [2] o Pragmatic Programming (PP) (Hunt y Thomas, 2000) [21] no incluyen una perspectiva de gestión. Extreme Programming (XP) (Beck, 1999) [4] se complementa con algunas directrices sobre la gestión de proyectos, pero aún no ofrece una vista general del mismo. Scrum, en cambio, está explícitamente destinado a gestionar proyectos de desarrollo ágil de software.

El enfoque promovido por Adaptive Software Development (ASD) (Highsmith, 2000) [16] es un modelo de gestión adaptable (liderazgo-colaboración). El punto de vista en el ASD está en cambiar la cultura de desarrollo de software, diciendo que la gestión también tendrá que armonizarse en respuesta a los cambios en los proyectos. Feature-Driven Development (FDD) (Palmer y Felsing, 2002) [34] ofrece medios para la

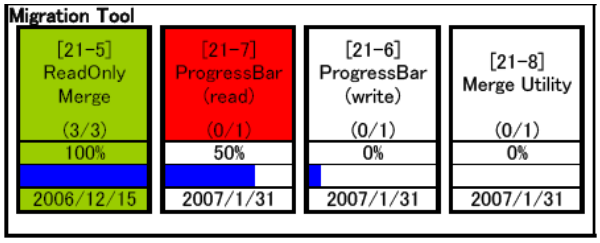
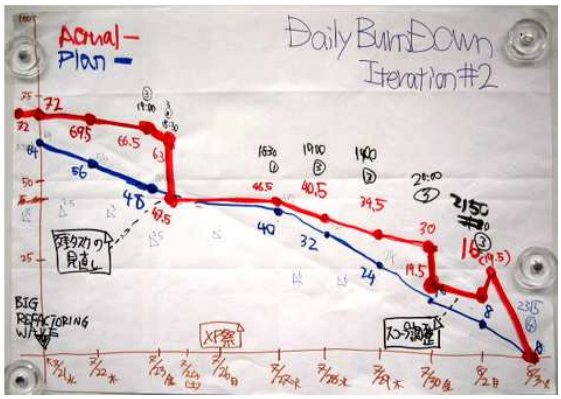
planificación de proyectos a través de las características de los productos, y el seguimiento del progreso de los proyectos. La propuesta FDD cree en la capacitación de los jefes de proyecto; tiene la última palabra sobre el alcance del proyecto, calendario, etc.

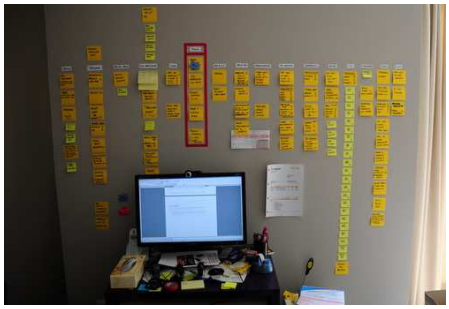
Dynamic Systems Development Method [13] sugiere un framework de controles para complementar el enfoque de desarrollo rápido de aplicaciones. Todos estos controles son diseñados para aumentar la capacidad de la organización en reaccionar a los cambios, es decir, en facilitar el trabajo de los equipos de desarrollo a través de seguimientos diarios (daily tracking) del progreso del proyecto.

La solución de la familia de metodologías Crystal (Cockburn, 2001) [11] en la gestión de proyectos se centra en la aumento de la capacidad de elegir correctamente un método para este fin.

La tabla 1 describe algunas de las herramientas y técnicas ágiles usadas en la gestión de proyectos.

Técnicas/Herramientas	**Propósito**
Kanban Board [36, 37]	Muestra el estado actual de todas las tareas por hacer dentro de una iteración. Las tareas se representan mediante tarjetas, y se clasifican en estados dentro del tablero: ToDo, Doing, y Done. 
Kanban Nano	Kanban Board Portable. 
Feature Kanban Board [18]	El eje horizontal de la tabla es una línea de tiempo, las zonas verticales representan las iteraciones o entregas, y cada tarjeta

	representa una característica que se llevará a cabo en la iteración. Ofrece una visión general de alto nivel de los productos a todos los participantes del proyecto. 
Parking Lot Chart [34]	Proporciona un resumen de alto nivel del estado del proyecto. 
Burndown Chart [43]	Muestra el estado actual de las tareas, así como la tasa de tareas pendientes en completar. 
Niko-niko Calendar (Smiley Calendar) [42]	Cada agente pone una marca en su propio calendario después de su jornada laboral. Permite ver la salud mental y motivación de los agentes en el proyecto. 

Daily Progress	Permite llevar un seguimiento del estado del proyecto o iteración de forma diaria. 

3.3.1. TARGETPROCESS

TargetProcess es una herramienta web de gestión de proyectos que impulsa los procesos de desarrollo software ágiles, centrándose en la planificación del proyecto, seguimiento del proyecto y seguimiento de errores. Soporta Extreme Programming y otras metodologías iterativas como Scrum.

Un proyecto consiste en varios tipos de entidad dependiendo del tamaño del proyecto y requisitos. La estructura global de cada proyecto se muestra en el siguiente diagrama:

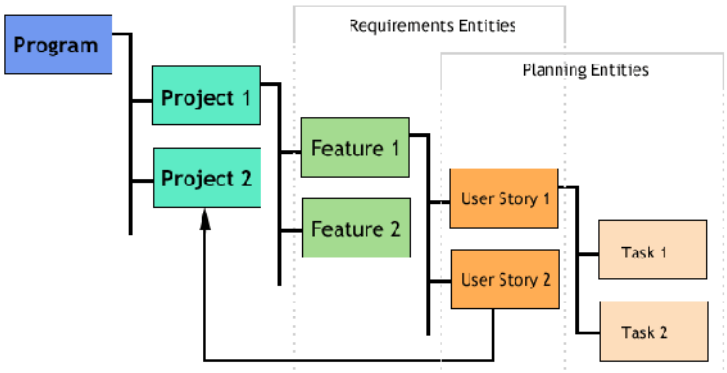


Figura 25 Estructura global de un proyecto

Un **Programa** es un proyecto grande que puede dividirse en varios proyectos más pequeños (Project 1 y Project 2 en la Figura 25). En la mayoría de los casos no se requiere de un programa, pero si los proyectos son desarrollados por diferentes equipos de trabajo, un programa ayuda a organizar el trabajo y llevar un seguimiento del progreso. Un **Proyecto** contiene el trabajo que abarca una o más características. Una **Característica** (Feature 1 y Feature 2 en la Figura 25) es un requisito de alto nivel que contiene historias de usuario (En la figura anterior el Feature 1 contiene la User Story 1 y User Story 2). Una **Historia de Usuario** es una unidad de trabajo que puede ser utilizada como requisito o como un elemento de planificación. Ejemplos típicos de

historias de usuario son “Búsqueda de contactos”, “Lista de contactos filtrable”, etc. Una **Tarea** es la unidad más pequeña de planificación que describe una pequeña parte de funcionalidad, por ejemplo, “Añadir IU de creación de usuario”, “Añadir un usuario administrador”, etc.

A continuación se muestra un ejemplo de organización de los elementos de una planificación en TargetProcess:

```
Program: NewWave CRM
Project: NewWave CRM Core
        Feature: Basic Contacts Management
                User Story: Quick Add Contact
                    Task: UI Design
                            Task: Implement Add Contact Logic
                User Story: Contacts Lis
                            Task: Filters
                            Task: List Sorting
                            Task: Contact List Page and Base Logic
                            Task: UI Design
```

La herramienta TargetProcess soporta el dearrollo iterativo. Un proyecto puede tener varias entregas (releases) y cada una de éstas puede estar formada por varias iteraciones tal y como se muestra en la Figura 26:

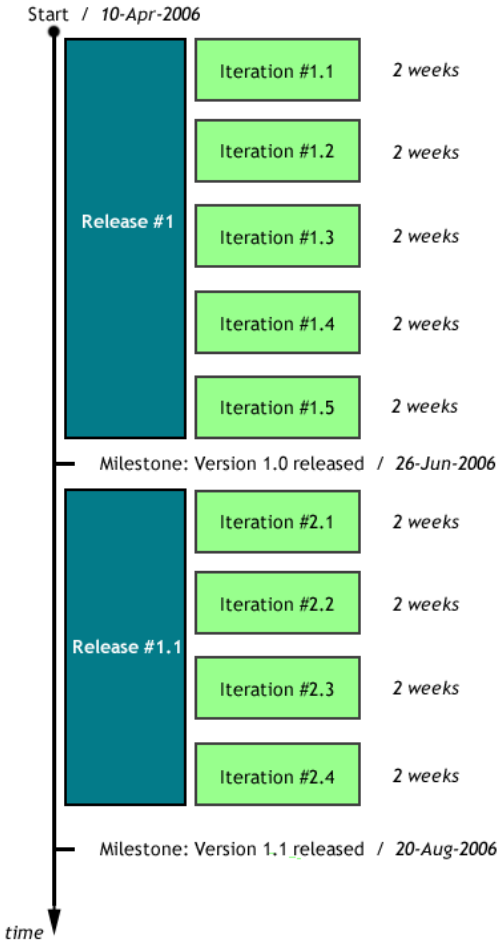


Figura 26 Desarrollo iterativo

Las características se asignan a las entregas, y las historias de usuario a las iteraciones o entregas, dependiendo de para qué se utilizan (ver Figura 27).

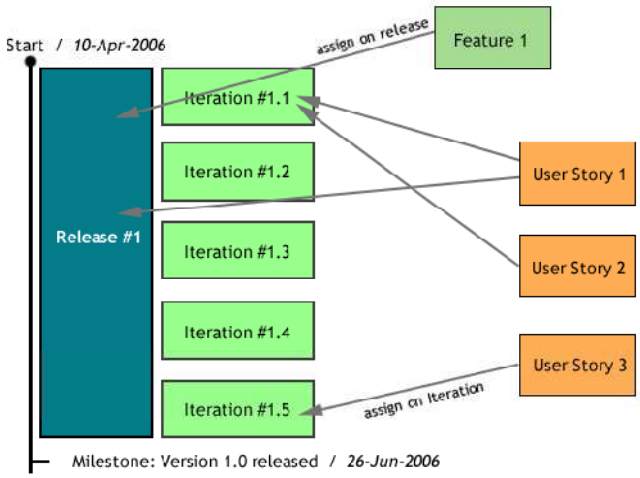


Figura 27 Asignación de características e historias de usuario

El proceso de planificación entero se resume en los siguientes pasos de acuerdo a las prácticas comentadas anteriormente:

- Añadir requisitos
- Añadir Características
- Añadir Historias de Usuario a las Características
- Crear la programación del Proyecto
- Añadir Releases e Iteraciones
- Asignar los elementos de planificación
- Asignar las Características a los Releases
- Asignar las Historias de Usuario a las Iteraciones
- Añadir detalles a la planificación
- Añadir Tareas a las Historias de Usuario
- Asignar los miembros del equipo a las Tareas

La ventana de creación de historias de usuario es la que se muestra en la Figura 28. La información asociada a la historia de usuario es: una descripción (1), la característica a la que está asignada (2), entrega o iteración asociada y el desarrollador asignado (3), tareas añadidas a la historia de usuario junto al esfuerzo estimado (4) y los archivos adjuntos (5).

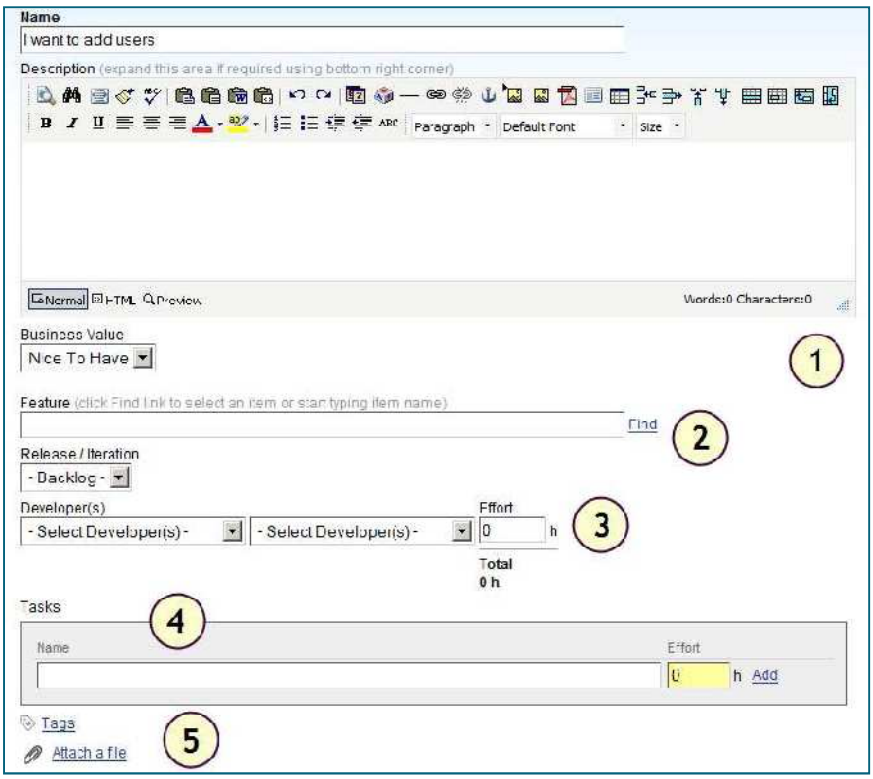


Figura 28 Creación de historias de usuario

Existen dos posibles unidades para medir el esfuerzo asociado a las historias de usuario y tareas: Horas Ideales o Puntos (Points). Las horas ideales es la unidad de esfuerzo temporal ideal; una hora ideal es el tiempo que un agente gasta en una tarea sin ninguna interrupción. Los Puntos es una unidad abstracta del tamaño de las historias de usuario que se basa en el *estímate-by-compare* para asignar la estimación. TargetProcess propone dos formas de calcular el esfuerzo asociado a las Historias de Usuario (US):

- US Total Effort = Sum(Tasks Developer Effort) + US Developer Effort
- US Total Effort = Sum(Tasks Developer Effort)

La primera propuesta calcula el esfuerzo total de la historia de usuario sumando el sumatorio de todos los esfuerzos estimados de sus tareas (Sum(Tasks Developer Effort)), y el esfuerzo estimado por el desarrollador (US Developer Effort). Mientras que la segunda propuesta, simplemente suma los esfuerzos estimados de las tareas de la historia de usuario.

La Figura 29 muestra un ejemplo en el que se calcula el esfuerzo asociado a una determinada historia de usuario utilizando la segunda forma propuesta. El esfuerzo total de la historia de usuario es 12h, el cual corresponde con el sumatorio de los esfuerzos de cada tarea (4h, 8h).

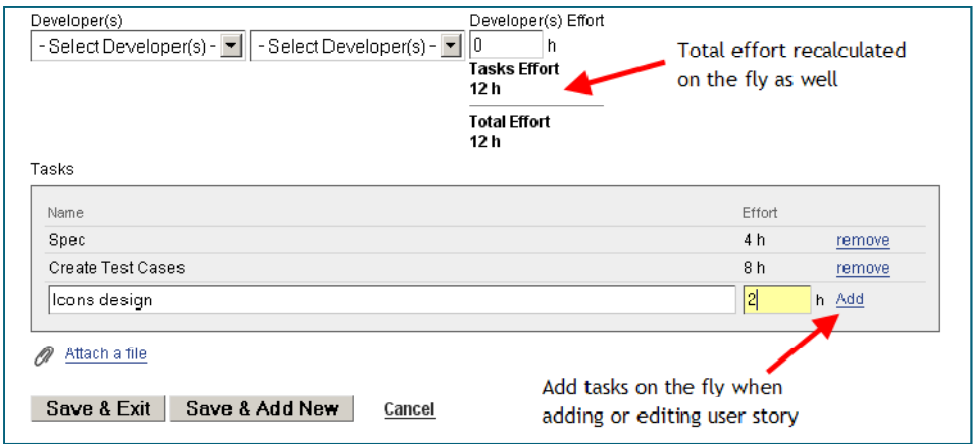


Figura 29 Cálculo del esfuerzo de una historia de usuario

Para analizar el seguimiento del progreso de una iteración, TargerProcess ofrece un módulo llamado Task Board para gestionar las tareas diarias. Contiene todas las historias de usuario de una iteración específica (lista de la izquierda de la Figura 30) y las tareas asignadas a cada una de ellas (resto de fichas que se encuentran en la parte central de la Figura 30). Para cada historia de usuario se muestra el esfuerzo estimado, el porcentaje completado, los desarrolladores asignados y el estado en el que se

encuentra (Open, In Progress, Done). Cada tarea contiene el tiempo que le queda al desarrollador para completar la tarea y el desarrollador asignado. Además, las tareas están clasificadas en diferentes columnas (ver Figura 30) según el estado en el que se encuentran (Open, In Progress, Done). El módulo Task Board de la Figura 30 permite realizar las siguientes acciones:

- Cambiar el estado de la tarea moviéndola de un área a otra.
- Asignar o reasignar los agentes a las tareas e historias de usuario.
- Actualizar el tiempo restante de las tareas. Esta actualización es realizada por los desarrolladores o por el jefe del proyecto consultando a cada uno de ellos.
- Mostrar las tareas de un agente específico aplicando un filtro.
- Ver los impedimentos de la historia de usuario. Los impedimentos hacen referencia a elementos que hacen que no se pueda continuar con el desarrollo de la historia de usuario. Cuando se asigna un impedimento a una historia de usuario, su desarrollo queda parado automáticamente. Como muestra la Figura 30, las historias de usuario con impedimentos son las que contienen el símbolo de prohibición en su ficha.

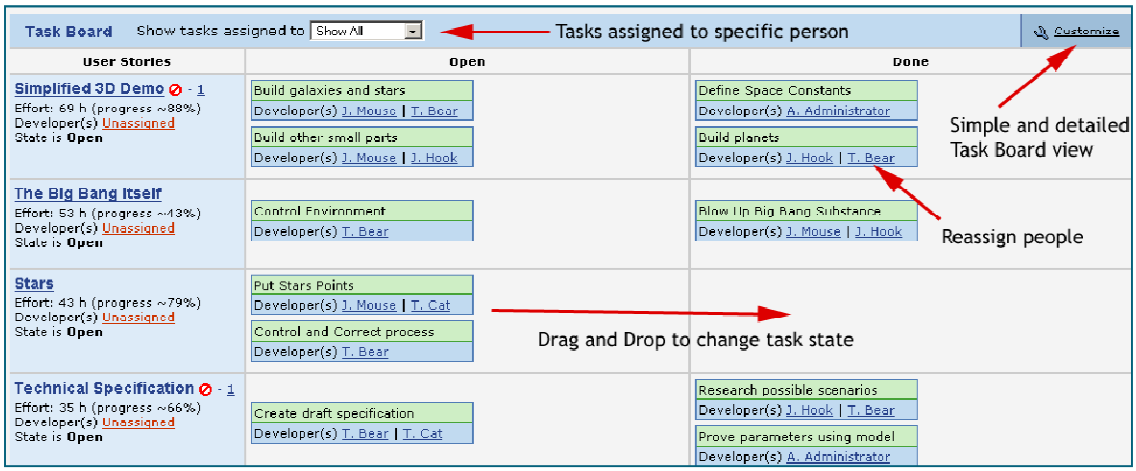


Figura 30 Task Board

La guía de usuario de TargetProcess indica que la única forma de conseguir la información actual sobre la completitud de las historias de usuario o tareas, es preguntando al desarrollador. Un agente puede introducir o actualizar el tiempo invertido y restante asociado a las historias de usuario o tareas mediante la interfaz de la Figura 31. La información necesaria para completar un registro de tiempo es: la historia de usuario o tarea correspondiente al registro de tiempo, una breve descripción de las

actividades que se van a realizar, la fecha, el tiempo invertido, el tiempo restante y el tipo de tiempo.

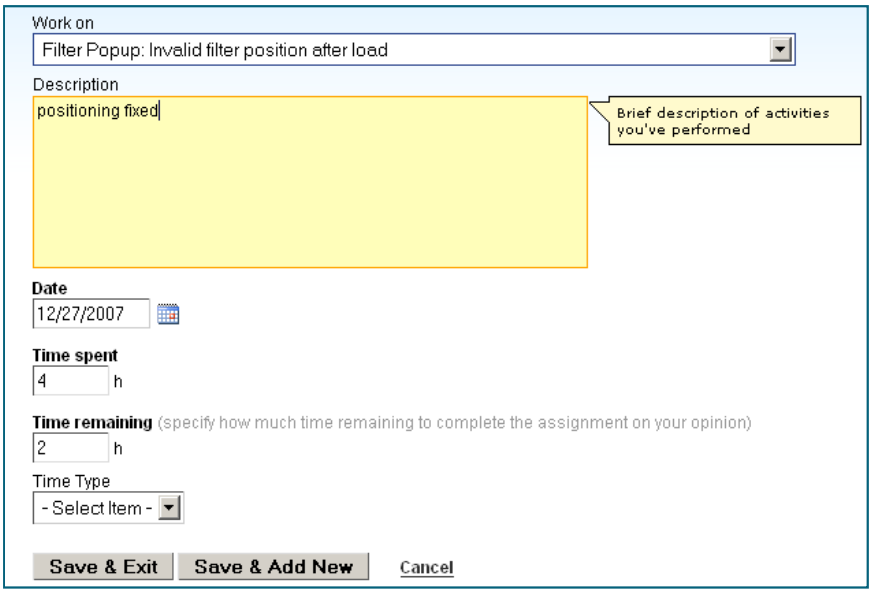


Figura 31 Tiempo invertido y tiempo restante

Existe otro módulo llamado Daily Progress (ver Figura 32) que permite llevar un seguimiento del estado de la iteración de forma diaria, mostrando las horas restantes que les queda a los desarrolladores para completar las historias de usuario o tareas. Si el tiempo restante incrementa con respecto al día anterior, se colorea en rojo la celda de las horas restantes alertando al agente de que existe un problema, y si al contrarío disminuye, permanece de color verde. Las celdas coloreadas en amarillo indican que las horas restantes disminuyen con respecto al día anterior, y han sobrepasado el esfuerzo estimado. En el caso ideal, el sumatorio de las horas restantes de una historia de usuario o tarea es igual al esfuerzo estimado.

Type	Name	Effort	23 Jul	24 Jul	25 Jul	26 Jul	27 Jul	28 Jul	29 Jul
STORY	As a User I want to see Help Panel to the left of each page with "Quick Start" use case	13pt (42h)		38	30			17	11
TASK	Create Test Cases	8h						5	2
TASK	Testing	8h							
TASK	Spec	0h							
TASK	Create object model for Help Use Cases	8h		4				2	0
TASK	Create the control of Flow Display	8h			2			0	0
TASK	Create the algorithm of disabling/hiding/resolving steps by context	8h			6			1	0
TASK	Terms replacing	2h						1	
STORY	As a User I want to Import nUnit tests results into TP as Test Plan Run (minimum)	8pt (39h)	38	33	33	32		27	24
TASK	Wire everything up	4h						4	1
TASK	Create Test Cases	6h				5			3
TASK	Testing	6h							
TASK	Changes in `Test Runs by Release/Iteration/Build` report	10h							
TASK	Spec	0h							
TASK	Empty plugin implementation	3h	2	0					
TASK	Write NUnit XML results parser with tests	5h		2	2				
TASK	Write simplest matcher to resolve TP test cases from NUnit test cases	5h						0	

Figura 32 Daily Progress

Para dar respuesta a la pregunta "¿Cuándo se completará la entrega?", TargetProcess ofrece el módulo llamado Progress Summary de la Figura 33. Como resultado de este módulo, el sistema ofrece una fecha de término del Release que se actualiza con cada actualización sobre los tiempos de las iteraciones. Para analizar el progreso compara el trabajo total asignado con respecto al completado, lo cual permite conocer el trabajo restante que queda para finalizar el Release y predecir una fecha de fin orientativa.

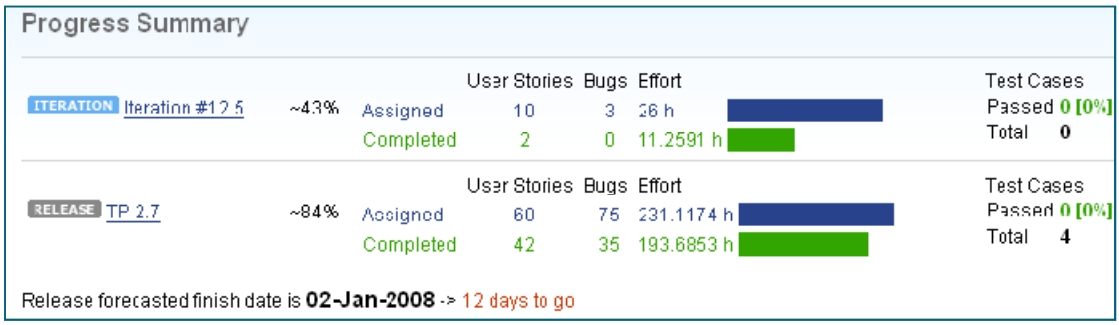


Figura 33 Progress Summary

3.3.2. VERSIONONE

La empresa VersionOne fue fundada en 2002 por Robert Holler y Rajiv Delwadia. En 2003 lanzó al mercado su herramienta de gestión y planificación de proyectos. El software de VersionOne es utilizado para coordinar los proyectos de desarrollo de software ágiles, principalmente, los planes, prioridades y el progreso del proyecto. Originalmente la herramienta se puso en marcha para eXtreme Programmning y Scrum,

y actualmente, soporta una serie de metodologías ágiles incluyendo DSDM, Agile UP y Lean/Kanban.

La suite del producto de VersionOne consiste en dos productos principales:

- V1: Agile Team. Herramienta para la planificación y seguimiento de un único proyecto, y orientada a equipos pequeños y con poca experiencia en el desarrollo ágil.
- V1: Agile Enterprise. Herramienta que apoya la gestión del ciclo de vida ágil en equipos con experiencia en el desarrollo ágil, y organizaciones que gestionan múltiples proyectos, equipos o localizaciones. Incluye la funcionalidad de V1: Agile Team más otra funcionalidad más avanzada.

En este trabajo vamos a centrarnos en la parte de planificación y seguimiento de proyectos, con lo cual, generalizaremos en la suite VersionOne y no en uno de sus productos en concreto.

La navegación que ofrece VersionOne refleja el flujo de un proceso ágil que consta en cinco etapas para la planificación y seguimiento de proyectos:

- Planificación de Productos: el equipo compone, estima y prioriza los backlog ítems (elementos o requisitos solicitados en un producto).
- Planificación de la Entrega: una vez creados los backlog ítems, el cliente tiene la opción de organizarlos en una o más entregas. Los equipos pueden también dividir el trabajo entre los grupos funcionales de los miembros del equipo.
- Planificación del Sprint: los backlog ítems se organizan en Sprints. El equipo define las pruebas de aceptación e identifica todas las tareas de cada backlog ítem del Sprint.
- Seguimiento del Sprint: los miembros del equipo actualizan el progreso y cierran los backlog ítems una vez han sido aceptados por el cliente.

A continuación, vamos a resumir estas cinco etapas del proceso para la planificación y seguimiento de proyectos.

Planificación de Productos

La Planificación del producto comienza creando la lista de características, historias de usuario o backlog ítems que integran la funcionalidad del producto. Al comienzo del proyecto el equipo identifica las características más importantes y son priorizadas por el cliente. Luego, son estimadas por el equipo de desarrollo.

En el grid de la Figura 34 se visualiza la lista de características contenidas en un proyecto o entregable determinado. La información que se muestra para cada característica es el título, ID, el usuario creador, prioridad, estimación, el proyecto al que pertenece, la fecha de la última modificación, etc. El grid ofrece mecanismos para filtrar, editar, ordenar, etc. la información desde las filas del grid.

Figura 34 Lista de características

La Figura 35 permite crear una característica. El único campo requerido por la aplicación es el título. El campo estimación (Estimate) se utiliza como una estimación de alto nivel para conducir la planificación de la entrega o iteración.

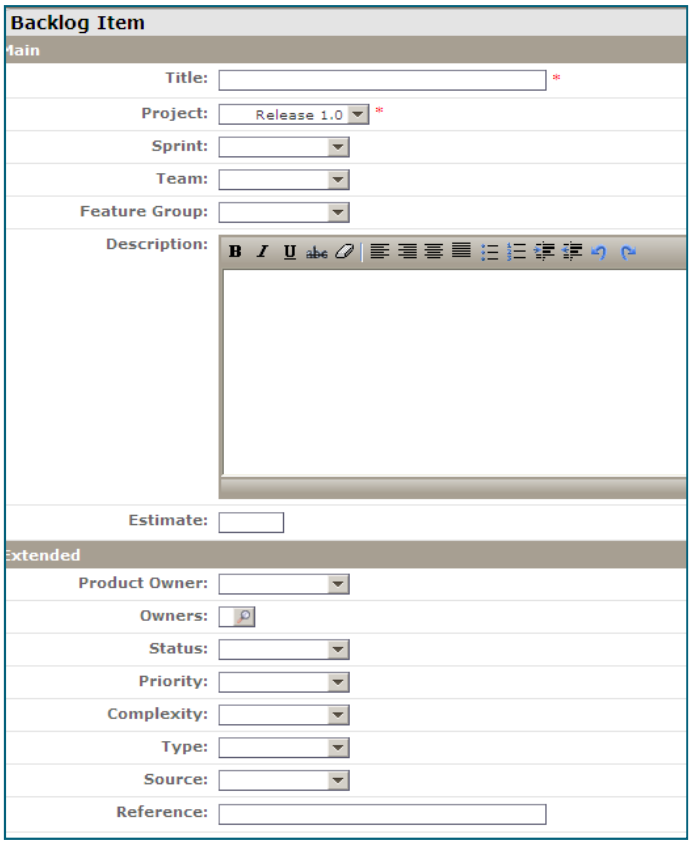


Figura 35 Creación de una característica

Planificación de la Entrega

La planificación de la entrega es la actividad en la que los equipos determinan el alcance del proyecto que será completado dentro de un plazo determinado. Las historias de usuario y/o características se asignan a un release, el cual se usa como entrada en la planificación de un conjunto de iteraciones pequeñas. La planificación de la entrega se realiza al comienzo del proyecto, pero luego se repite en intervalos a lo largo del mismo a fin de orientarlo mediante la retroalimentación con el cliente, la velocidad del equipo (trabajo que el equipo puede conseguir hacer por iteración) y cualquier otra información. El objetivo de esta etapa es comunicar las características que serán entregadas en el plazo indicado por el deadline.

Las fichas del área superior de la pestaña Planificación de la Entrega (Release Scheduling) de la Figura 36 corresponden a cada uno de las entregas definidas en el proyecto. Para poder determinar las características que serán desarrolladas en cada uno de estas entregas, se muestran las fechas de inicio y fin de la entrega, las horas estimadas totales, las horas restantes para completar la entrega, número de

características, y la estimación de cada uno de ellas. Al desplegar la ficha de la entrega, se muestran todas las características que están asignadas hasta el momento a la entrega. En el grid de la parte inferior de la Figura 36 se observa la lista de características pendientes de asignar a un proyecto o release. Al arrastrar alguna de estas características a la ficha del release, se asigna automáticamente al mismo.

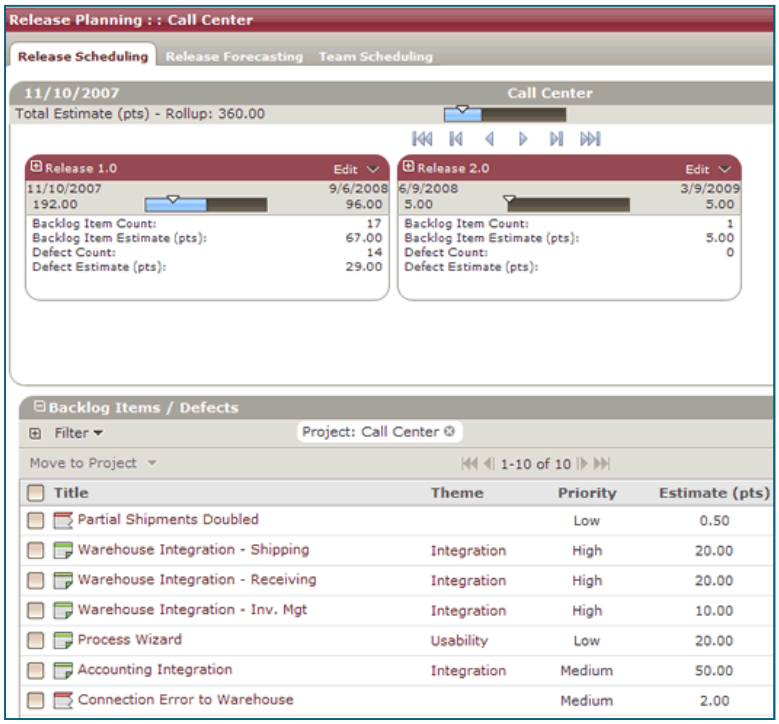


Figura 36 Planificación de la entrega

Planificación del Sprint

La planificación del Sprint es la actividad que los equipos realizan para identificar las características que serán trabajadas durante un sprint. VersionOne propone realizar una reunión al comienzo de todos los sprints en la que los equipos discuten las características con prioridad más alta, las desglosan en tareas específicas, y estiman el esfuerzo en completarlas. La velocidad y los resultados anteriores de los equipos son usados para determinar la cantidad de trabajo que el equipo programa en un sprint y se compromete a completar. Los miembros del equipo suelen seleccionar las tareas iniciales durante la reunión y luego, recogen las tareas adicionales durante el sprint.

La velocidad es un método para medir la velocidad con la que los equipos entregan el valor de negocio. El método consta en sumar las estimaciones de las características

entregadas por iteración. La velocidad es medida en las mismas unidades que las estimaciones de las características; puntos, días, días ideales u horas.

En la parte superior de la pestaña Sprint Scheduling de la Figura 37 se observan los sprints creados para un proyecto o release determinados. La información que se visualiza en cada sprint es la fecha de inicio y fin del sprint, la estimación en puntos, el número de características (backlogs y defectos) y su correspondiente estimación, etc. Al desplegar un sprint, se muestra la lista de características que tiene asignadas.

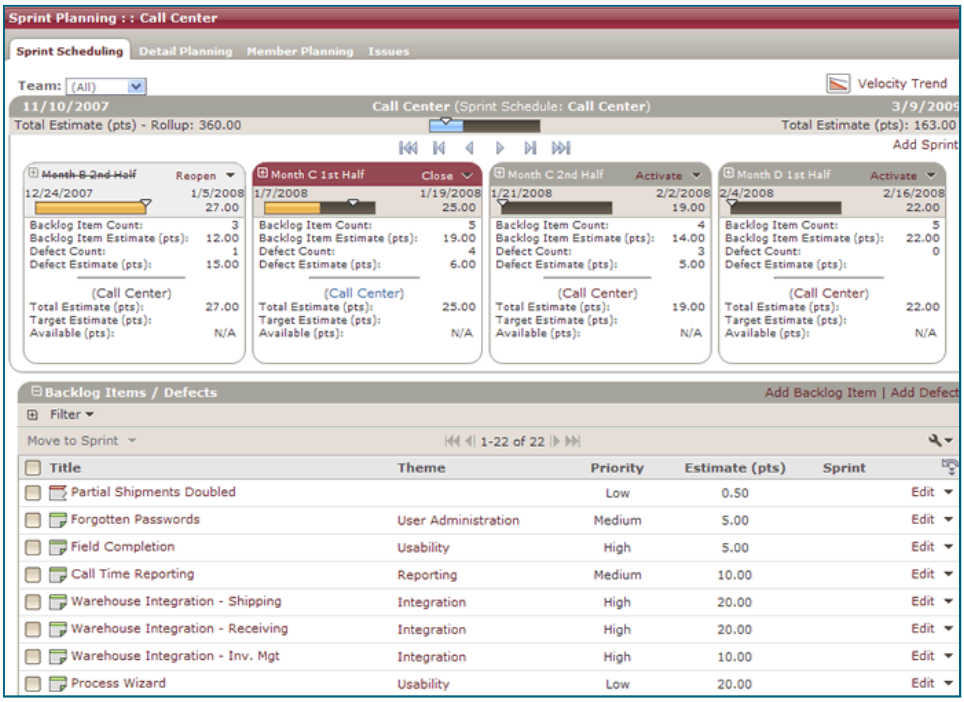


Figura 37 Planificación del Sprint

Seguimiento del Sprint

El seguimiento del Sprint es la actividad que los equipos realizan para controlar el progreso de un Sprint. Cada miembro del equipo, normalmente, actualiza el esfuerzo restante y completado de sus tareas diariamente. El esfuerzo restante representa el estado verdadero de un sprint en cualquier instante de tiempo. Conforme progresa el sprint, el esfuerzo restante requerido para completar las tareas de la iteración debería disminuir. Si en cualquier momento dado del sprint parece que el trabajo restante no se acercará a cero al final del sprint, el equipo y/o jefe del proyecto puede optar por tomar las medidas oportunas.

En la ventana Detalle del Seguimiento (Detail tracking) de la Figura 38 se puede realizar un seguimiento del estado del sprint actual o seleccionado, comparándolo con el trabajo realizado en el último sprint cerrado. La información que se muestra para el sprint es el nombre, fecha de fin, número de backlogs y defectos, el total de puntos y horas estimadas, el número de horas completadas y las horas restantes en completar el sprint. La barra de progreso facilita ver el punto en el que se encuentra el Sprint. La lista inferior está formada por el conjunto de características que forman el sprint. Cada característica representa un backlog ítem o defecto y está formada por un conjunto de tareas, por ejemplo, realizar las pruebas de aceptación, implementación, testeo, etc. necesarias para completarla. Cada tarea consta de un nombre, Id, agentes asignados a la tarea, estado (Future, In Progress, Done), puntos y horas estimadas, horas completadas, esfuerzo y horas que quedan para completar la tarea. El estado, esfuerzo y las horas restantes son campos editables que permiten a los miembros del equipo actualizar la información de la tarea, y automáticamente, actualizar el valor de las horas completadas. La información que se resume en las características corresponde al sumatorio de la información de todas las tareas que contiene, es decir, muestra por ejemplo el total de puntos estimados para todas las tareas que forman la característica.

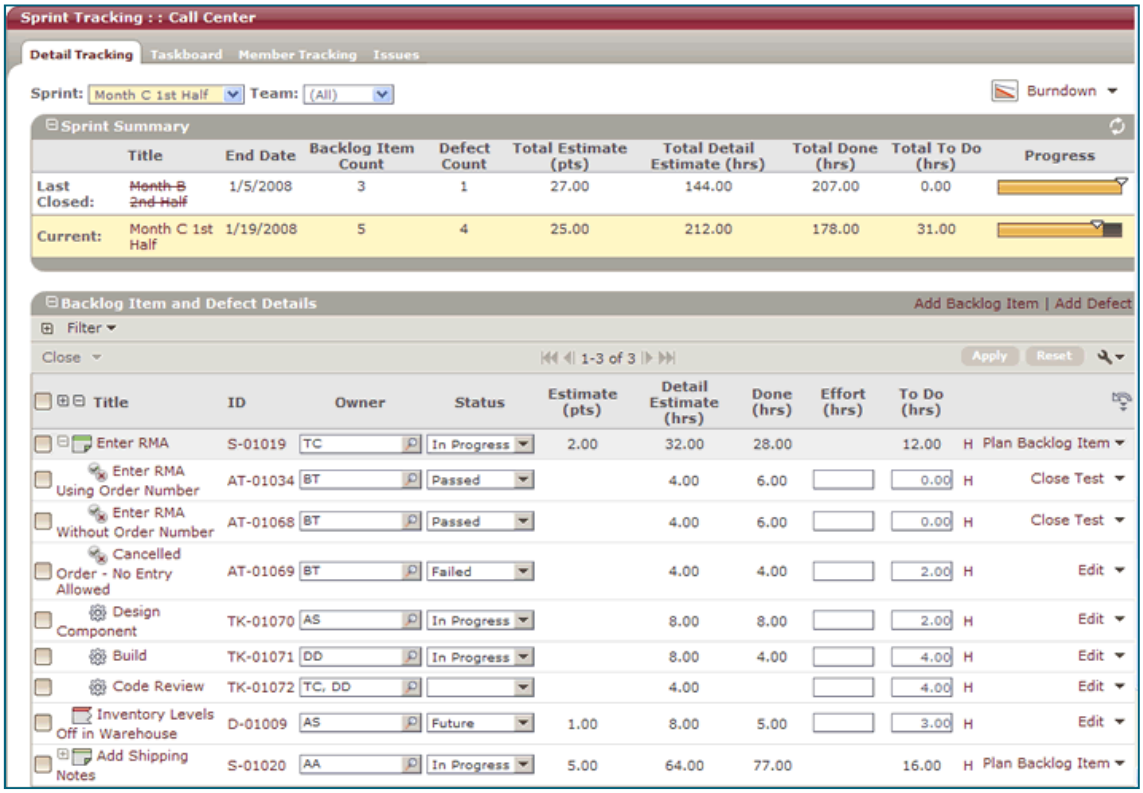


Figura 38 Pestaña Detail Planning de la Planificación del Sprint

Conclusiones respecto de herramientas

Las herramientas de gestión de proyectos software vistas en este trabajo han sido clasificadas en tres grupos: herramientas genéricas para la gestión de proyectos, herramientas CASE con gestión de proyectos y metodología tradicional, y herramientas para gestión de proyectos con metodologías ágiles.

Existen una gran variedad de herramientas genéricas para la gestión de proyectos software que cubren todas las fases del proceso, sin embargo, no ofrecen un sistema de gestión del tiempo que permita, automáticamente, conocer el estado actual de la planificación. La clave del éxito de cualquier proyecto está en comparar el estado actual del proyecto con respecto a lo planificado, y así poder detectar a tiempo los riesgos que hacen que la planificación esté desequilibrada y no se puedan cumplir los compromisos con el cliente. En la mayoría de proyectos la planificación se realiza al inicio del mismo quedando la información desactualizada con el tiempo.

Además, no son adecuadas para un proceso de desarrollo iterativo e incremental, donde se entrega funcionalidad en periodos de corta duración a lo largo del proyecto. El enfoque tradicional para abordar el desarrollo de software, según Letelier y Penadés (2006) [30], ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), donde por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. Por eso, los modelos tradicionales no soportan un proceso de desarrollo iterativo e incremental donde se realizan frecuentes entregas de corta duración.

El siguiente diagrama Gantt de la herramienta MS-Project muestra un ejemplo donde las tareas corresponden a actividades del proyecto y son realizadas de forma secuencial en el tiempo.

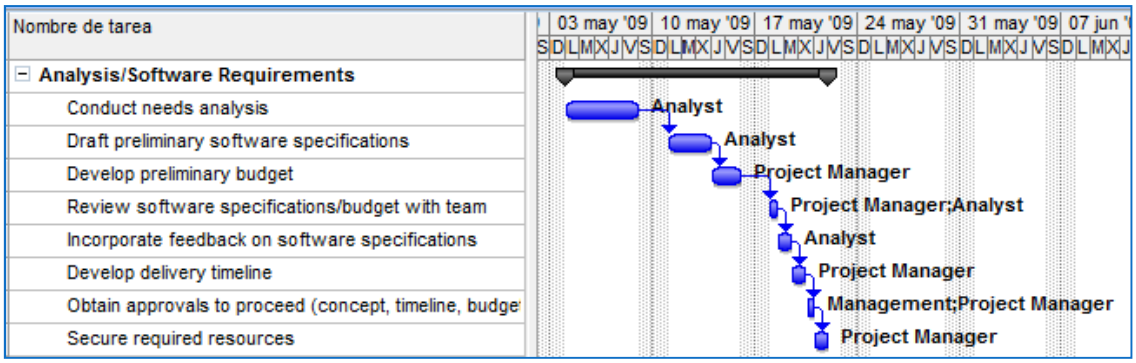


Figura 39 Proceso de desarrollo tradicional desde MS-Project

Por otra parte, MS-Project permite definir un diagrama Gantt cuyas tareas agregadas representan versiones del proyecto. Dentro de cada versión se crean unidades de trabajo cada una de las cuales se completan mediante actividades de diferentes workflows. Esta forma de organizar las tareas corresponde a un proceso iterativo e incremental, puesto que las unidades de trabajo están incluidas en versiones de funcionalidad. La siguiente imagen muestra un ejemplo de proceso de desarrollo iterativo e incremental usando MS-Project.

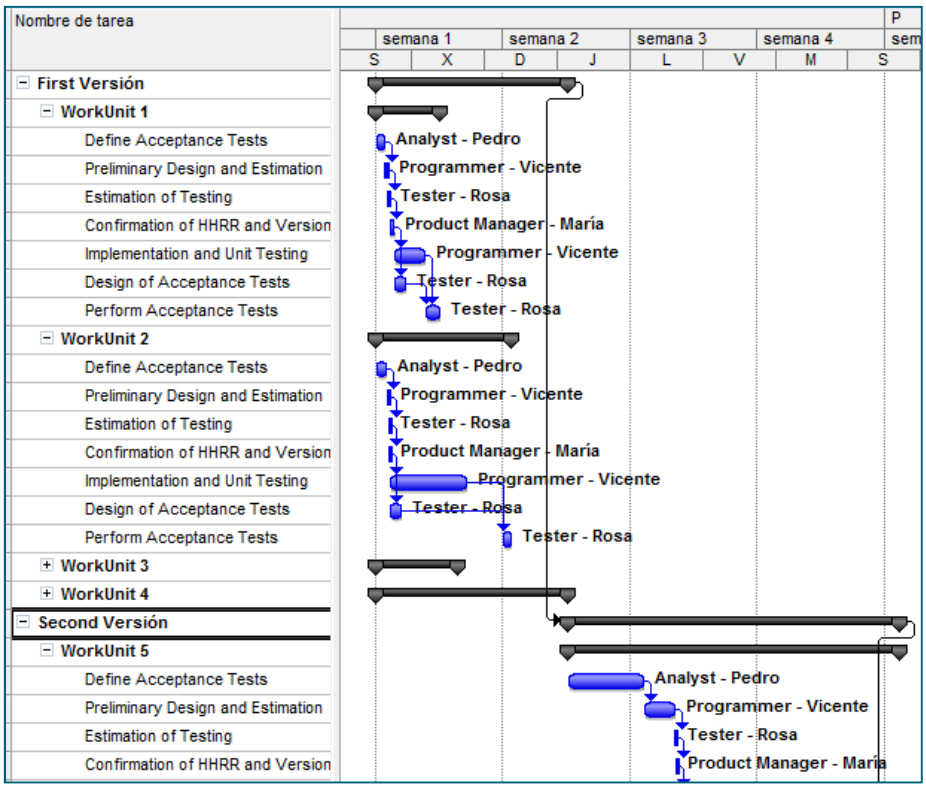


Figura 40 Proceso de desarrollo iterativo e incremental desde MS-Project

Las herramientas tradicionales de gestión de proyectos no resultan efectivas en un entorno donde predomina la colaboración entre los miembros del equipo. Estas herramientas están fuertemente orientadas al proceso y ofrecen pocos mecanismos que

permitan a los participantes del proyecto interaccionar entre ellos, tanto a nivel de comunicación como de trabajo compartido.

Cuando las unidades de trabajo siguen un workflow donde las actividades están estrechamente relacionadas entre sí, cualquier imprevisto sobre una actividad repercute en las fechas de comienzo y fin de las actividades relacionadas, por eso, no están preparadas para soportar workflows. Los agentes que utilizan estas herramientas trabajan desconectados de la herramienta, por ejemplo, los tiempos invertidos en tareas se actualizan cuando el agente los actualiza manualmente, lo agentes no son avisados o no se enteran cuando se producen cambios en los compromisos o en la planificación, etc. Todos estos inconvenientes hacen que estas herramientas genéricas no sean efectivas en la gestión de proyectos de desarrollo y mantenimiento de software.

Las herramientas CASE (en particular la estudiada, EA) con gestión de proyectos centran su atención en coordinar y facilitar la cooperación entre los miembros del equipo, dando menos interés a la planificación y seguimiento de proyectos. La información temporal de las tareas se muestra en listas, sin ninguna prioridad ni precedencia entre ellas. Los recursos se asignan a tareas, pero no ofrecen mecanismos que gestionan la disponibilidad de los agentes ni su balanceo de carga de trabajo. Por lo tanto, las herramientas CASE apoyan el desarrollo y mantenimiento de software de forma completa, pero carecen de funcionalidad suficiente respecto de la gestión de proyectos.

Las herramientas ágiles para la gestión de proyectos se acercan más a los entornos cambiantes y desarrollos iterativos e incrementales, pero son demasiado simples en cuanto a planificación y seguimiento de proyectos de desarrollo. Aunque las metodologías ágiles promueven que la clave del éxito está en los individuos, es necesaria una gestión del control sobre los proyectos. Los cambios realizados en los requisitos durante el proyecto son bienvenidos gracias a la flexibilidad de la planificación y de que no existen fechas de inicio y fin entre las tareas, dan más importancia a la evaluación que a la medición.

Las herramientas ágiles de gestión de proyectos no permiten una organización temporal detallada (a nivel de tarea), y no tienen establecida una relación de precedencia entre ellas (workflows, diagramas Gantt, etc.), simplemente existe una lista con el trabajo asignado en una iteración. Además, las tareas se asignan a los agentes sin considerar su

disponibilidad de trabajo, esto lo resuelven de manera personal (cara a cara). Para actualizar el estado de las tareas, los agentes introducen los datos temporales de forma manual e imprecisa ya que la herramienta está totalmente desconectada del trabajo del agente. El jefe de proyecto utiliza esta información, y al propio agente, para tomar decisiones con respecto al estado de su trabajo. Resultan interesantes los mecanismos que utilizan para seguir el avance del proyecto (tablas Kanban, BurnDown Charts, etc.), pero carecen de información temporal actualizada debido al rudimentario sistema de gestión del tiempo que ofrecen.

En conclusión, una adecuada planificación y seguimiento de proyectos de desarrollo y mantenimiento de software requiere combinar aspectos metodológicos ágiles y tradicionales, puesto que de forma individual, las herramientas que apoyan dichas metodologías presentan deficiencias tal y como se ha comentado anteriormente.

Capítulo 4. La metodología TUNE-UP

Un proyecto de desarrollo o mantenimiento de software tiene por objetivo el conseguir una entrega exitosa del producto. En un proceso iterativo e incremental el trabajo asociado a una entrega del producto se divide en varias versiones (resultantes de cada iteración) siendo la última aquella que coincide con la entrega del producto.

La metodología "ideal" para la planificación y seguimiento de proyectos software debe tener en cuenta ciertas particularidades:

- Resulta más conveniente utilizar un proceso iterativo e incremental
- Gran parte del trabajo se realiza en paralelo
- El trabajo de los agentes es muy colaborativo y cambiante
- La coordinación entre agentes debería estar guiada por workflows
- Es importante conectar la herramienta con el trabajo del agente
- La planificación debe estar conectada con la gestión del producto, es decir, la gestión de requisitos del producto software.

La metodología TUNE-UP y su herramienta de apoyo TUNE-UP Process Tool, surgen como respuesta a estas particularidades sobre la gestión de proyectos software.

TUNE-UP es una metodología que incorpora aspectos ágiles y tradicionales con un sentido marcadamente pragmático. TUNE-UP se caracteriza fundamentalmente por combinar los siguientes elementos:

- **Modelo iterativo e incremental** para el desarrollo y mantenimiento del software. El trabajo se divide en unidades de trabajo que son asignadas a versiones del producto. Se realizan ciclos cortos de desarrollo, entre 3 y 6 semanas, dependiendo del producto.
- **Workflows flexibles** para la coordinación del trabajo asociado a cada unidad de trabajo. Los productos, según sus características, tienen disponibles un conjunto de workflows los cuales se asignan a cada una de las unidades de trabajo. Cada unidad de trabajo sigue el flujo de actividades del workflow para completarla. Bajo ciertas condiciones se permite saltar hacia adelante o hacia atrás en el workflow, así como cambios de agentes asignados e incluso cambio de workflow. Por ejemplo, las típicas situaciones de re-trabajo en desarrollo de

software ocasionadas por detección de defectos se abordan con saltos atrás no explícitos en el workflow.

- **Proceso de desarrollo dirigido por las pruebas de aceptación (Test-Driven)**. La definición de una unidad de trabajo es básicamente la especificación de sus pruebas de aceptación acordadas con el cliente. A partir de allí, todo el proceso gira en torno a ellas, se estima el esfuerzo de implementar, diseñar y aplicar dichas pruebas, se diseñan e implementan y luego se aplican sobre el producto para garantizar el éxito de la implementación.
- **Planificación y seguimiento continuo centrados en la gestión del tiempo**. En todo momento debe estar actualizado el estado de las versiones, de las unidades de trabajo, y del trabajo asignado a los agentes. El jefe del proyecto puede actuar oportunamente con dicha información, tomando decisiones tales como: redistribuir carga de trabajo entre agentes, cambiar los plazos de la versión, mover unidades de trabajo entre versiones, etc.
- **Control de tiempos**. Los agentes registran el tiempo que dedican a la realización de las actividades, el cual se compara con los tiempos estimados en cada una de ellas, detectando oportunamente desviaciones significativas. Esto permite a los agentes gestionar más efectivamente su tiempo, mejorar sus estimaciones y ofrecer al jefe del proyecto información actualizada del estado de la versión.

TUNE-UP es una metodología que incorpora aspectos de metodologías ágiles y de metodologías tradicionales. Las dos primeras características (proceso iterativo e incremental, y proceso centrado en las pruebas de aceptación) clasifican a TUNE-UP como metodología ágil, sin embargo, las otras características están más próximas de lo que sería una metodología tradicional.

Este trabajo se centra precisamente en las dos últimas características que veremos en capítulos posteriores.

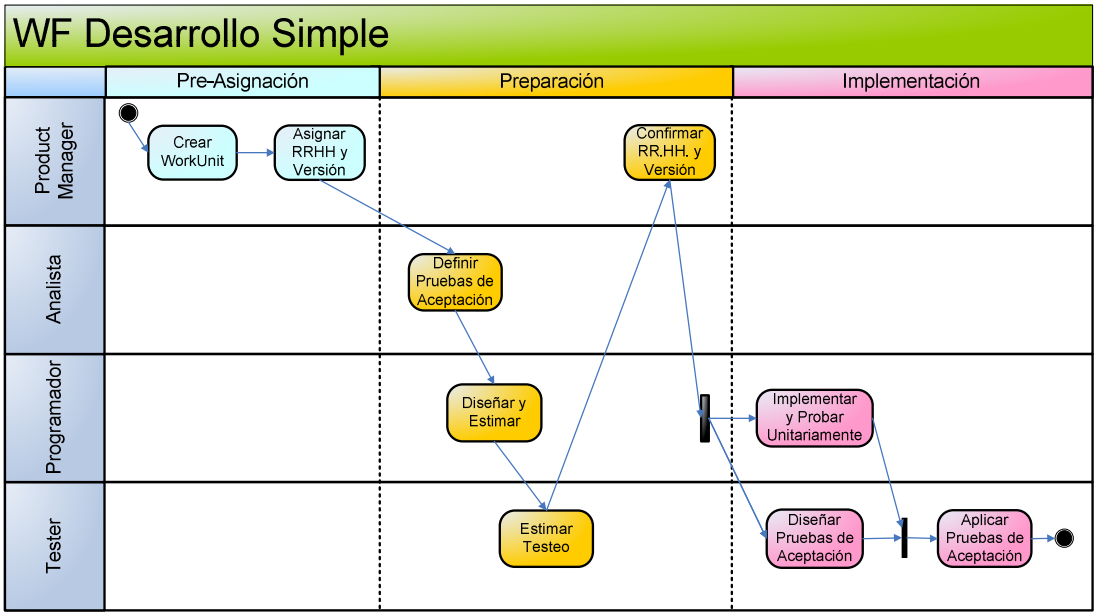


Figura 41 Un workflow de desarrollo simple para unidades de trabajo

La Figura 41 ilustra un workflow mínimo para el desarrollo de una unidad de trabajo. Un workflow, en general, incluye actividades asociadas a tres fases del proceso:

- Pre-Asignación: actividades realizadas hasta la asignación de los RRHH y versión. La unidad de trabajo ha sido identificada pero aún no tiene una prioridad como para asignarle los RRHH y versión.
- Preparación: se pueden realizar cuando la unidad de trabajo ha alcanzado cierta prioridad y se le han asignado los RRHH y una versión. Se comienza a trabajar en la preparación de la unidad de trabajo, y debería concluirse antes del inicio de la versión objetivo en la cual se implementará. Incluye el análisis, las revisiones y estimaciones para su implementación.
- Implementación: se realizan durante la versión objetivo. Incluye la implementación, aplicación de pruebas e implantación.

Las actividades de cada workflow pueden variar significativamente dependiendo de factores tales como: cantidad y especialización de agentes participantes, validaciones o negociaciones predeterminadas con el cliente, características del producto (necesidad de migración, traducción, etc.), niveles y actividades de pruebas (unitarias, de integración, de aceptación, pruebas de regresión, automatización de pruebas), etc. Cada unidad de trabajo en una iteración del proyecto podría tener su propio workflow. Sin embargo, en la práctica basta con disponer de un reducido conjunto de workflows que permitan cubrir los tipos de unidades de trabajo que se presentan en el proyecto.

Desde el punto de vista del agente responsable, el estado en el que se puede encontrar una unidad de trabajo asociada a la actividad que debe realizar puede ser:

- Por Llegar: el agente está asignado a la actividad pero aún no ha recibido la unidad de trabajo pues está en alguna actividad anterior en el workflow.
- Pendiente: el agente ha recibido la unidad de trabajo en la actividad pero aún no ha comenzado a trabajar en ella.
- Activa: el agente está trabajando en la actividad (y el sistema está registrando tiempo dedicado a ella).
- Pausada: el agente ha interrumpido su trabajo en la actividad (y se ha detenido el registro de tiempo.
- Finalizada: el agente ha terminado la actividad (la unidad de trabajo ha pasado automáticamente a las actividades siguientes en el workflow asociado).
- Omitida: la actividad ha sido omitida, es decir, se ha decidido no realizar la actividad (se ha saltado hacia adelante en el workflow). A menos que la unidad de trabajo dé un salto hacia atrás, la unidad de trabajo no pasará por la actividad.

La Figura 42 muestra los cambios de estados posibles de una unidad de trabajo en una actividad.

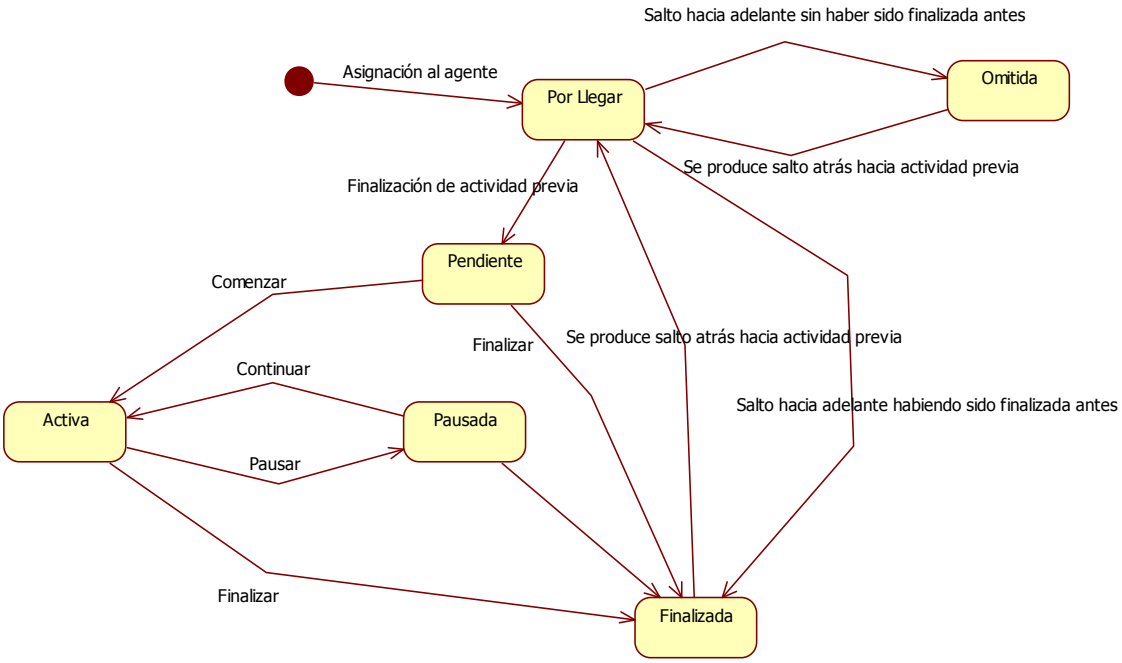


Figura 42 Posibles estados de una unidad de trabajo en una actividad

Capítulo 5. Gestión del tiempo en TUNE-UP

5.1. Control de tiempos

Según Humphrey (2001) [45], la forma de mejorar la calidad de nuestro trabajo comienza por entender lo que hacemos realmente. Esto significa que se deben de conocer las tareas que se hacen, cómo se hacen, y los resultados que se obtienen. Como primera aproximación, basta con conocer las actividades que se realizan y averiguar el tiempo que se gasta en cada una de ellas; es necesario medir el tiempo invertido. Humphrey también propone ideas de cómo registrar el tiempo de forma consistente y precisa, exactamente:

- Lleva siempre contigo el cuaderno de notas
- Cuando ocasionalmente olvides registrar la hora de comienzo, la hora de fin o la duración de la interrupción, haz una estimación tan pronto como lo recuerdes
- Puedes utilizar un cronómetro para controlar las interrupciones
- Resume tu tiempo puntualmente

Yendo más allá, en TUNE-UP se ha introducido una disciplina de registro de tiempo en la que los agentes registran los tiempos que invierten en las actividades de forma asistida. Cuando un agente inicia una actividad se contabilizan el tiempo dedicado a dicha actividad, y cuando la finaliza, el registro de tiempo se finaliza. Esta técnica permite conocer las duraciones de las actividades de forma mucho más precisa que la propuesta por Humphrey. TUNE-UP también permite realizar ajustes sobre los registros de tiempo, por ejemplo, cuando un agente olvida detener el registro de tiempo u olvida comenzar a registrar el trabajo en una actividad.

La metodología TUNE-UP utiliza para la estimación de las actividades la técnica denominada juicio del experto. Antes de que un agente inicie una actividad, debe estimar el tiempo que tardará en realizarla. Los workflows deben incluir actividades de estimación para estimar las actividades que se encuentran en la fase de Implementación de la metodología. Al igual que los registros de tiempo, TUNE-UP permite realizar ajustes sobre las estimaciones, por ejemplo, cuando el agente detecta que va a tardar más de lo estimado en realizar la actividad, o a lo contrario, estima que va a tardar menos de lo que preveía.

5.2. Planificación y seguimiento continuo centrados en la gestión del tiempo

En cualquier tipo de proyecto a priori es interesante contar con una previsión del ordenamiento de las actividades, de los tiempos máximos y mínimos de comienzo y finalización de las actividades, y de las holguras asociadas. Sin embargo, cuando se trata de un proyecto de desarrollo y mantenimiento de software, esto puede resultar muy complejo de establecer y puede que en la práctica no resulte todo lo efectivo que fuera de esperar. En un proyecto de desarrollo de software cobran protagonismo aspectos como los siguientes:

- Los agentes en general no cogen una actividad y la terminan. Frecuentemente es necesario discutir alternativas o analizar elementos previamente no considerados. Esto provoca que los agentes detengan actividades y continúen con otras, para posteriormente retomar aquellas detenidas. Además, es inevitable el retrabajo generado por saltos atrás en el workflow, como por ejemplo desde actividades de pruebas hacia actividades de implementación, cuando se detectan fallos.
- Para el ordenamiento y restricciones de tiempo entre actividades deben analizarse todas las actividades del proyecto (o incluso de otros proyectos si los recursos son compartidos) pues el trabajo en general es colaborativo y el retraso o anticipación en el trabajo de un agente afectará los tiempos de otros agentes.
- Cada iteración o versión del producto suele incluir muchas unidades de trabajo, cada una de ellas siguiendo un workflow específico y realizada por agentes específicos. Así, un grafo de precedencia de actividades podría llegar a tener cientos, incluso miles de nodos.
- Los workflows tienen una expresividad mucho mayor que la abordada en diagramas de red como PERT/CPM, uno de los métodos más populares utilizados en planificación de proyectos. Así, dichos métodos deben ser adaptados para poder aplicarse a workflows (Hyun et al., 2005) [22].

Por lo anterior, en TUNE-UP, como una primera aproximación a dicho cálculos de tiempos, hemos estado trabajando con lo que denominamos “holgura simple” del agente, la cual sólo considera el tiempo restante con respecto del tiempo disponible. Con la información que se recolecta en TUNE-UP (apartado 5.1 Control de tiempos de

este trabajo) es muy sencillo y rápido calcular dicha holgura simple. Así, cuando la holgura simple de algún agente es negativa (o positiva pero muy pequeña) podemos asegurar que el proyecto está desnivelado. Sin embargo, cuando el agente tiene teóricamente suficiente disponibilidad respecto de las horas restantes, no se puede asegurar que el agente se encuentre en situación de desbalanceo de carga puesto que no estamos considerando que el trabajo depende también de otros agentes. Aún así, la holgura simple nos ha resultado útil como indicador de sobrecarga del agente y síntoma de riesgo en los compromisos de una versión. La gestión del tiempo en TUNE-UP incorpora fundamentalmente las siguientes prácticas:

- Dividir el trabajo en unidades de trabajo que sean convenientes no sólo de cara a la captura y negociación de requisitos con el cliente sino también para planificación y seguimiento del proyecto. De forma orientativa las unidades de trabajo no deben superar una semana de trabajo para una actividad de un agente.
- Definición del workflow de cada unidad de trabajo previendo el flujo de actividades por las cuales pasará
- Asignación de unidades de trabajo a agentes balanceando la carga de trabajo de cada agente
- Estimación (y re-estimación) del esfuerzo para realizar las actividades asociadas a la realización de las unidades de trabajo
- Registro actualizado del tiempo invertido en actividades
- Seguimiento continuo del estado del proyecto posibilitando acciones correctivas oportunas

La Tabla 19 Abreviaturas de la fórmula de cálculo de la holgura simple presenta las abreviaturas utilizadas en las fórmulas de cálculo de la holgura simple de un agente.

Abreviatura	Significado
T_E	Tiempo estimado.
T_R	Tiempo registrado.
T_P	Tiempo del período, considerado como una fracción respecto al tiempo en un mes.
TA_M	Tiempo contractual del agente en un mes.
TN_P	Tiempo no considerado en el período, por festivos, ausencias, permisos, bajas, etc.
TE_P	Tiempo extra del agente en el periodo.

TR_P	Tiempo restante para finalizar actividades en el periodo.
TD_P	Tiempo disponible para abordar actividades en el periodo.
TH_P	Tiempo de holgura simple del agente en el periodo.

Tabla 19 Abreviaturas de la fórmula de cálculo de la holgura simple

La fórmula (1) permite obtener el tiempo restante del agente sumando las diferencias entre tiempo estimado y real para cada actividad asignada y que deba entregarse en el período analizado. La fórmula (2) determina el tiempo disponible del agente en un período a partir del día de hoy. Con el resultado de (1) y (2) conseguimos la holgura simple del agente reflejada en (3).

$$TR_P = \sum_{Actividades}(T_E - T_R) \qquad (1)$$

$$TD_P = T_P * TA_M - TN_P + TE_P \qquad (2)$$

$$TH_P = TD_P - TR_P \qquad (3)$$

Así, por ejemplo si quisiéramos conocer la holgura simple de un agente en las próximas dos semanas (T_P). Calcularíamos la suma de las diferencias entre estimaciones y registros de tiempo para todas sus actividades que tienen fecha de entrega en las próximas dos semanas, supongamos que esta suma nos da un tiempo restante de 65 hrs (TR_P). Suponiendo que el agente trabaja 120 hrs/mes (TA_M), y que no existe tiempo a descartar en el período ($TN_P = 0$) ni tiempo extra ($TE_P = 0$), entonces el tiempo disponible para el agente en el período son 60 hrs (TD_P = 0.5 * 120 – 0 + 0). Así, finalmente la holgura simple para el agente es – 5 hrs, es decir, a día de hoy el agente no podría cumplir con sus compromisos.

La implementación de dichos cálculos de holgura es más compleja pues debe considerar otros elementos (ya incorporados en TUNE-UP Process Tool) como por ejemplo:

- Un agente puede trabajar en varios productos a la vez y puede interesar fijar porcentajes de distribución de tiempo del agente a cada producto, con lo cual a nivel global el agente puede tener una cierta situación de holgura que puede ser diferente cuando se analiza a nivel de actividades de un determinado producto. De forma similar puede ser interesante que el agente tenga porcentajes prefijados de distribución de tiempo respecto de las actividades, es decir, su situación de holgura puede ser incluso relativa a una determinada actividad.

- Cuando el tiempo registrado sobrepasa al tiempo estimado el cálculo de tiempo restante se invalida pues se considerarían tiempo restante negativo para una actividad. En estos casos se descarta dicho tiempo y se alerta de la situación para que el agente proceda a una re-estimación.
- La flexibilidad del workflow de la unidad de trabajo obliga a hacer reajustes de estado de las actividades cuando se produce re-asignación del agente de una actividad, saltos hacia atrás o hacia adelante en el workflow o incluso cambio del workflow de una unidad de trabajo.
- Todas las actividades no finalizadas y con fecha de entrega anterior al período consultado deben ser consideradas en dicho período como actividades retrasadas.

Así, con estos mecanismos de cálculo, un agente puede en cualquier momento visualizar su estado de holgura con respecto a una cierta fecha futura, y en particular con respecto a las fechas de término de la versión de algún producto. El jefe de proyecto puede también visualizar la situación de holgura de cada agente y efectuar las acciones pertinentes, por ejemplo: reasignación de actividades entre agentes, reasignación de versiones de las unidades de trabajo, cambios en fechas de término de versión, cambios en distribución de tiempos de los agentes a los productos, etc.

Capítulo 6. TUNE -UP Process Tool: herramienta de apoyo a la metodología TUNE-UP

TUNE-UP Process Tool es una herramienta de apoyo para la aplicación efectiva de la metodología TUNE-UP. La herramienta está formada por tres módulos principales: Planificador Personal, Gestor de Unidades de Trabajo, y Planificador de Versiones. A continuación se describe cada uno de estos módulos.

6.1. Planificador Personal

Un aspecto clave para el proyecto es que los agentes se dediquen a las actividades acertadas de acuerdo con las prioridades. El agente decide qué actividad realizar dependiendo de los plazos del producto (fechas de inicio y término de versión), la prioridad de la unidad de trabajo en la versión, y a la posición de la actividad en el workflow.

El Planificador Personal (PP) presenta el trabajo que tiene un agente. Cuando un agente inicia su jornada laboral, accede al PP (Figura 43) para ver el trabajo que tiene asignado actualmente, y seleccionar qué va a realizar de acuerdo a sus prioridades. El PP ofrece una variedad de facilidades de filtrado y ordenamiento de información, además de datos de tiempos, para que el agente pueda determinar su elección de acuerdo a sus prioridades. La tabla de la izquierda de la Figura 43 resume las contabilizaciones de las unidades de trabajo según la actividad y estado en el que se encuentran, y la tabla de la derecha, muestra información de dichas unidades de trabajo incluyendo: producto, versión, descripción de la unidad de trabajo, tiempos calculados, estado de la actividad actual dentro del workflow, etc. Con los tiempos calculados el agente puede conocer el tiempo que lleva registrado en cada actividad (T.Real), los tiempos que ha estimado (T.Est. y T.Est.A., estimado y estimado ajustado), el porcentaje completado con respecto al estimado (%Com.) y las horas restantes (H. Res.) que le quedan para finalizar la actividad. En la Figura 43, al seleccionar una celda de la tabla de la izquierda, se muestran las respectivas unidades de trabajo en la tabla de la derecha.

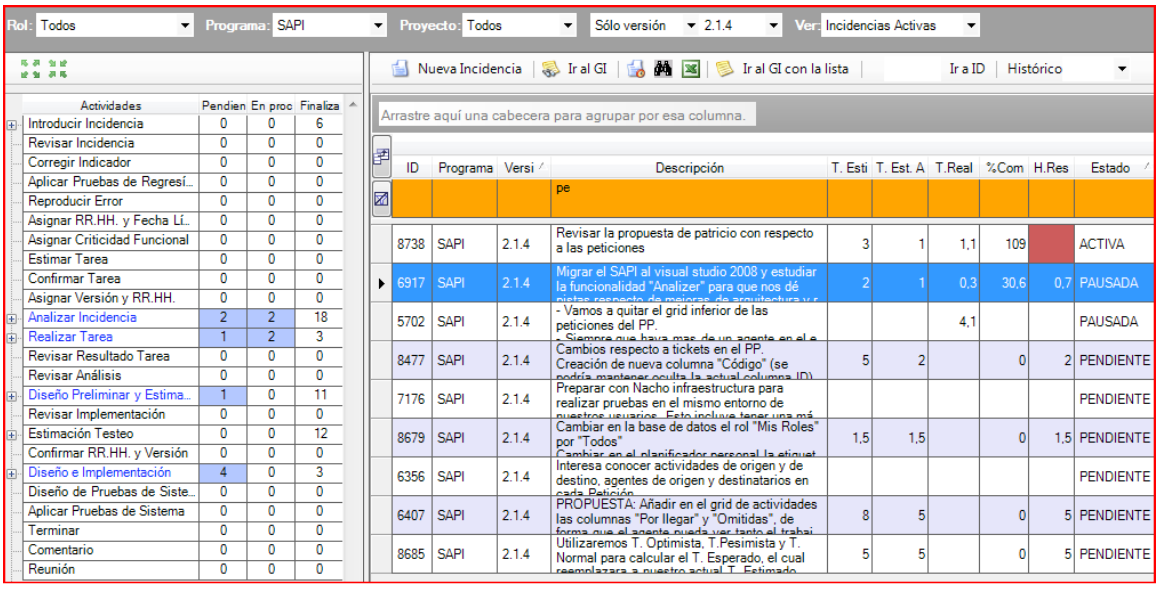


Figura 43 Fragmento de interfaz del Planificador Personal

6.2. Gestor de Unidades de Trabajo

Cuando el agente decide la unidad de trabajo que va a realizar, accede con ella al Gestor de Unidades de Trabajo (GUT) (ver Figura 44) como apoyo esencial para realizar su tarea. En la parte superior de la Figura 44 se observan los datos generales de la unidad de trabajo, y en la parte inferior, un conjunto de pestañas con la funcionalidad que ofrece el GUT. En la pestaña Seguimiento de la Figura 44 se observa el conjunto de actividades del workflow por las que ha pasado la unidad de trabajo. Cada actividad está asociada a un registro de seguimiento que incluye la actividad, el agente que la desarrolla, el estado en el que se encuentra, veces que se ha realizado la actividad (indicación del retrabajo), etc. En esta pestaña además, con los botones Empezar/Pausar/Continuar (es el mismo botón que cambia de nombre según el estado de la actividad) y Finalizar Actividad, el agente puede controlar el registro de tiempo, activando, pausando o finalizando la actividad.

Al finalizar una actividad, la unidad de trabajo pasa automáticamente a la siguiente actividad del workflow. El motor de workflows que incorpora TUNE-UP Process Tool permite realizar saltos a cualquier actividad del workflow, y soporta el paralelismo, secuenciación y bifurcación de las actividades.

Figura 44 Gestor de Unidades de Trabajo – Pestaña Seguimiento

La pestaña Tiempos de la Figura 45 ofrece funcionalidad específica de tiempos para la unidad de trabajo. En la parte superior los agentes establecen las estimaciones de las actividades que van a desarrollar. En la tabla inferior se muestran todos los registros de tiempo producidos automáticamente por los pares de acciones Comenzar/Continuar – Pausar/Finalizar, correspondientes a las actividades ejecutadas en la unidad de trabajo. Tanto en las estimaciones como en los registros de tiempo el agente puede realizar ajustes cuando, por ejemplo, prevea que la estimación no se va a cumplir, o cuando haya olvidado Activar, Pausar o Finalizar la actividad.

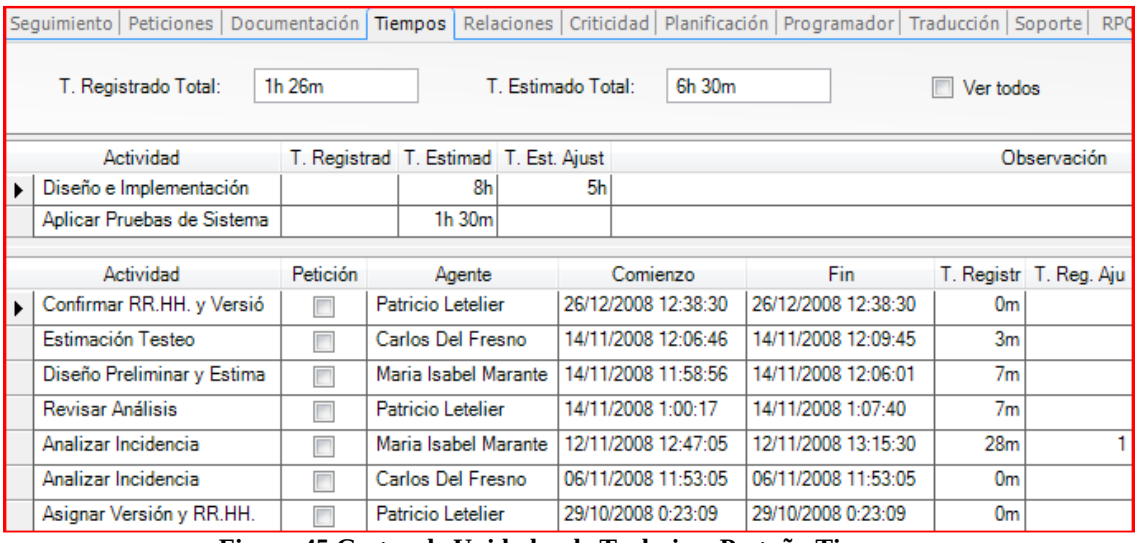


Figura 45 Gestor de Unidades de Trabajo – Pestaña Tiempos

La Pestaña Documentación de la Figura 46 incluye una biblioteca de documentos (especificaciones y material complementario) compartida entre los agentes que trabajan con la unidad de trabajo. En ella se ofrecen plantillas específicas y facilidades para llevar un control de versiones de los documentos.

guimiento | Peticiones | Documentación | Tiempos | Relaciones | Criticidad | Planificación | Programador | Traducción | Soporte | RPQ

Añadir archivo | Añadir imagen | Añadir plantilla | Check in | Check out | Checklist de programación

Ultima modificación	Agente	Tipo	Archivo	Carpeta	Sub doc	Observación	Check out
06/03/2009 9:22:51	Raquel García	Análisis	análisis.doc			Segunda Versión.	
03/03/2009 16:12:06	Elena Campos G	Otros	102_2.JPG			Despues de esto se cierra el programa.	
02/03/2009 9:52:49	Elena Campos G	Otros	id_102.avi			Error al acceder a la tabla maestra desde la	
24/02/2009 15:43:56	Tomislav Delalic	Pruebas de Si	testeo.doc				
24/02/2009 9:31:14	Tomislav Delalic	Otros	error_grid_102.avi			Error de grids al introducir linea fuera de	
29/01/2009 13:20:56	Pablo Fernandez	Diseño	diseño.doc				
01/02/2008 9:27:56	Raquel García	Otros	Anexo Ausencias.docx			REVISARLO, PORQUE HA CAMBIADO EL	
23/01/2008 11:59:18	Aliate Mohamed	Diseño	diseño.doc				

Figura 46 Gestor de Unidades de Trabajo – Pestaña Documentación

El Gestor de Unidades de Trabajo ofrece además otras pestañas entre las que destacan: Peticiones (mecanismo sencillo de comunicación entre los agentes para comentar respecto de la unidad de trabajo), Relaciones (relaciones de dependencia de la unidad de trabajo con respecto de otras) y Planificación (asignación de agentes a actividades de la unidad de trabajo y asignación a una versión).

6.3. Planificador de Versiones

El Planificador de Versiones (PV) permite gestionar los productos, sus versiones, los workflows disponibles para cada producto, los agentes por defecto asignados a las actividades de los workflows y realizar el seguimiento continuo del estado actual de las versiones.

Incidencias | Carga de Agentes en Versión

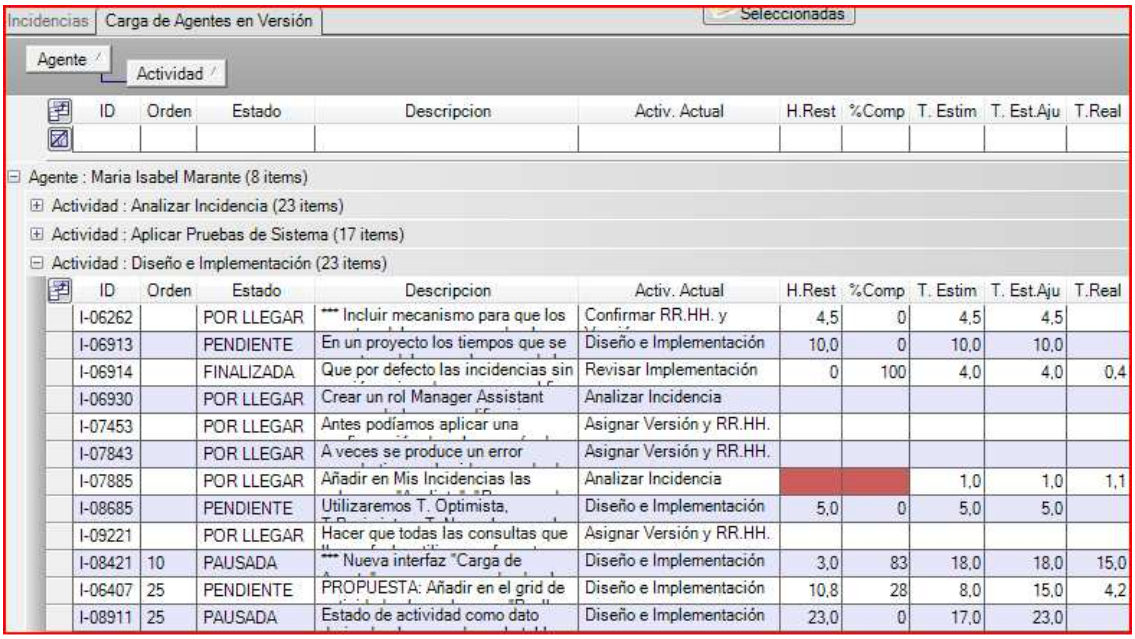
Arrastre aquí una cabecera para agrupar por esa columna.

ID	Version	Orden	Descripción	Proyecto	Activ. Actual	Analizar Incidencia
	2.1.4				Desestimar	
8679	2.1.4		Cambiar en la base de datos el rol "Mis Roles" por "Todos"...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
6261	2.1.4	10	Documentación de Ayuda - Permitir la edición de estos los documentos de...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8477	2.1.4	15	Cambios respecto a tickets en el PP. Creación de nueva columna "Código" (se podría...	Tickets SAPI	Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8395	2.1.4	20	Vamos a echarle un vistazo nuevamente para ver hasta qué punto sigue fallando y si encontramos...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
7110	2.1.4	30	Tener una pestaña de documentación para cada program, al estilo de la pestaña de las incidencia...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8194	2.1.4	40	Que aparezca en el PP la actividad "Desestimar" en el grid de actividades.		Analizar Incidencia (0) / Maria Isabel Marante	Maria Isabel Maran...
8635	2.1.4	40	Poner la funcionalidad "Ir al GI con la Lista" en el grid Detalles de Versión		Analizar Incidencia (0) / Maria Isabel Marante	Maria Isabel Maran...
6264	2.1.4	50	Mantenimiento de Tablas Maestras en el SAPI: Agentes, Roles, Actividades y Programas. Se po...		Diseño e Implementación (1) / Maria Isabel Marante	Maria Isabel Maran...
5702	2.1.4	100	PROPUESTA: Incluir las peticiones (con su estado) en el grid de actividades. El agente vería...	Gestión de Tiem...	Diseño e Implementación (1) / Maria Isabel Marante	Maria Isabel Maran...
6407	2.1.4	100	PROPUESTA: Añadir en el grid de actividades las columnas "Por llegar" y "Omitidas", de forma que	Gestión de Tiem...	Diseño e Implementación (0) / Maria Isabel Marante	Maria Isabel Maran...

Figura 47 Planificador de Versiones – Pestaña Incidencias

La ventana de la Figura 47 muestra la lista de unidades de trabajo que están asignadas a una versión de un determinado producto. El jefe de proyecto puede consultar los datos resumidos de cada unidad de trabajo en la versión, en particular, puede conocer la actividad actual en que se encuentra dentro del workflow, el orden de prioridad, el esfuerzo que implica elaborar la unidad de trabajo, el agente asignado a las actividades principales del workflow, etc. Además, cuenta con potentes mecanismos que ayudan a explotar esa información: filtros, ordenamiento, agrupamiento de información, etc. Con toda esta información, el jefe de proyecto puede planificar las versiones, cambiando de versión las unidades de trabajo y asignando las prioridades a cada una de ellas de acuerdo con sus compromisos.

La pestaña Carga de Agentes de la Figura 48 permite a cualquier miembro del equipo conocer el trabajo que tiene asignado en la versión y el estado en el que se encuentra. Por otro lado, el jefe del proyecto puede realizar un seguimiento continuo del estado de la versión. La información está agrupada por Agente y Actividad. La columna Estado indica el estado en que se encuentra la unidad de trabajo en la actividad. Además, se muestra el tiempo que el agente ha registrado en la actividad hasta el momento (T.Real), el tiempo estimado (T.Estim), el tiempo estimado ajustado que corrige al anterior (T.Est.Aju), el porcentaje de completado de la actividad (%Comple) y las horas restantes que le quedan al agente para completar la actividad (H. Rest). Para que este último dato sea fiable, las horas registradas del agente no pueden sobrepasar el tiempo estimado, en este caso, el sistema indicaría al agente que debe realizar un ajuste en su estimación coloreando la celda de rojo.



ID	Orden	Estado	Descripcion	Activ. Actual	H.Rest	%Comp	T. Estim	T. Est.Aju	T.Real
Agente : Maria Isabel Marante (8 items)									
Actividad : Analizar Incidencia (23 items)									
Actividad : Aplicar Pruebas de Sistema (17 items)									
Actividad : Diseño e Implementación (23 items)									
I-06262		POR LLEGAR	*** Incluir mecanismo para que los	Confirmar RR.HH. y	4,5	0	4,5	4,5	
I-06913		PENDIENTE	En un proyecto los tiempos que se	Diseño e Implementación	10,0	0	10,0	10,0	
I-06914		FINALIZADA	Que por defecto las incidencias sin	Revisar Implementación	0	100	4,0	4,0	0,4
I-06930		POR LLEGAR	Crear un rol Manager Assistant	Analizar Incidencia					
I-07453		POR LLEGAR	Antes podíamos aplicar una	Asignar Versión y RR.HH.					
I-07843		POR LLEGAR	A veces se produce un error	Asignar Versión y RR.HH.					
I-07885		POR LLEGAR	Añadir en Mis Incidencias las	Analizar Incidencia			1,0	1,0	1,1
I-08685		PENDIENTE	Utilizaremos T. Optimista,	Diseño e Implementación	5,0	0	5,0	5,0	
I-09221		POR LLEGAR	Hacer que todas las consultas que	Asignar Versión y RR.HH.					
I-08421	10	PAUSADA	*** Nueva interfaz "Carga de	Diseño e Implementación	3,0	83	18,0	18,0	15,0
I-06407	25	PENDIENTE	PROPUESTA: Añadir en el grid de	Diseño e Implementación	10,8	28	8,0	15,0	4,2
I-08911	25	PAUSADA	Estado de actividad como dato	Diseño e Implementación	23,0	0	17,0	23,0	

Figura 48 Planificador de Versiones – Pestaña Carga de Agentes

Toda esta información permite conocer la Holgura Simple del agente, la cual considera el tiempo restante (T.Rest.) con respecto el tiempo que dispone el agente para terminar su trabajo. Así, cuando la holgura simple de algún agente es negativa, puesto que no dispone de tiempo para terminar su trabajo, podemos asegurar que la carga del agente en la versión está desbalanceada, y por tanto, esto afectará a toda la versión. Gracias a esta información el jefe de proyecto puede tomar las acciones correctivas oportunas, como por ejemplo, cambiar la unidad de trabajo de versión, dividir la unidad de trabajo, pasar parte del trabajo a otro agente, modificar los plazos de entrega de la versión, negociar tiempo extra con el agente, etc. La holgura simple nos ha resultado útil como indicador de sobrecarga del agente y síntoma de riesgo en los compromisos de una versión.

Capítulo 7. Conclusiones y trabajo futuro

La clave para el éxito de un proyecto es su correcta planificación y seguimiento. Tanto el jefe del proyecto como cada uno de los participantes (incluyendo también al cliente) deben tener una visión actualizada del estado de sus actividades y del proyecto en su conjunto.

La adecuada gestión de los tiempos en un proyecto permite tomar acciones correctivas oportunas. Tanto el jefe del proyecto como cada uno de los agentes pueden detectar desajustes en el cumplimiento de sus compromisos observando las holguras. Sin embargo, estos beneficios sólo se alcanzan cuando se dispone de la información actualizada y completa de los tiempos del proyecto. Por otra parte, no existe un sistema prefecto en cuanto a precisión pues en los cálculos influyen muchos factores no predecibles ni controlables totalmente, por ejemplo: la fiabilidad de las estimaciones, trabajo no considerado inicialmente y que hay que incluir en la planificación, cambios de prioridades, imprevistos del agente que afectan su disponibilidad planificada, etc. Aún con estos inconvenientes, las mejoras en la gestión de tiempos de proyectos reportan beneficios significativos para el desempeño del equipo de desarrollo.

Modelos de referencia como CMMI y estándares como ISO 90003, así como otras guías para la gestión de proyectos tales como PMBOK y PSP, sólo recomiendan prácticas muy generales en cuanto a gestión de proyectos. Son las metodologías las encargadas de definir mecanismos concretos para aplicar dichas prácticas.

Las metodologías ágiles aciertan respecto de la granularidad en la definición de las unidades de trabajo, su asignación y estimación, el seguimiento continuo, y promueven informalmente en cierto grado el registro actualizado de tiempos invertidos. Sin embargo, la definición de roles tan genéricos, acompañada del prácticamente inexistente concepto de workflow, suele ser incompatible con otros esquemas de desarrollo basados en mayor especialización (analistas, programadores, testers, etc.) y workflows que coordinan el trabajo. Cuando el mismo agente desempeña varios roles se reduce la comunicación y coordinación con otros agentes (al menos respecto de las tareas de dichas roles). Pero encontrar agentes que satisfagan un perfil múltiple es difícil. Además, cuando el trabajo es colaborativo existe una tendencia natural hacia la especialización de los agentes. Por último, el seguimiento en un enfoque ágil se delega en gran medida a los agentes (confiando en su propia disciplina para abordarlo).

Por otra parte, curiosamente cuando se utiliza una metodología tradicional en los planes se suele asumir un modelo de proceso en cascada, lo cual resulta fácil de elaborar y entender, algo positivo para cerrar un acuerdo con el cliente, pero contraproducente para el control y seguimiento si dicho modelo de proceso no es el más apropiado para el proyecto. Además, aunque las metodologías tradicionales ponen más énfasis en el trabajo colaborativo basado en workflows, no ofrecen mecanismos para su aplicación. Tampoco promueven la asignación ni estimación detallada respecto de esfuerzos estimados o registrados por los agentes.

Existe una gran cantidad de herramientas genéricas para gestión de proyectos, sin embargo, éstas presentan inconvenientes importantes para la gestión de tiempos en proyectos software. Por una parte, no consideran las particularidades de un proyecto de software, especialmente no soportan adecuadamente un proceso iterativo e incremental (usando versiones) basado en unidades de trabajo que se implementan de forma colaborativa. Por otra parte, no promueven/soportan un registro actualizado de estimaciones y tiempos invertidos pues en la práctica se usan desconectadas del trabajo diario de los agentes, los cuales periódicamente deberían actualizar los datos del proyecto para asegurar que la información del estado del proyecto está actualizada y completa. TUNE-UP hace que la planificación y seguimiento del proyecto se integre en el trabajo diario de los agentes.

En los últimos años han aparecido diversas herramientas para planificación ágil de proyectos software, las cuales, a diferencia de las herramientas tradicionales, asumen que el proceso de desarrollo es iterativo e incremental y basado en unidades de trabajo más que en actividades. Entre las herramientas que hemos estudiado destacan: TargetProcess y VersionOne. Estas herramientas presentan un especial atractivo respecto de la interfaz y plataforma Web, sin embargo, en cuanto a gestión de tiempo los mecanismos ofrecidos son muy rudimentarios. Además no tienen el concepto de workflow asociado a cada unidad de trabajo (usan simplemente un atributo estado, por ejemplo: pendiente, finalizada, etc.).

Es reconocido que la implantación de una metodología es un proceso largo y que presenta dificultades, tales como la formación y entrenamiento del equipo o la resistencia al cambio por parte de algunos agentes. Una gestión de tiempo como la ofrecida por TUNE-UP conlleva además otros obstáculos:

- Disciplina de estimación y registro de tiempos. Hay que ofrecer pautas para realizar la estimación y entrenar a los agentes en ello. Las acciones asociadas al registro de tiempo deben ser lo menos molestas para los agentes.
- Desconfianza de los agentes al control basado en los tiempos. Es difícil convencer a los agentes de que las ventajas de esta gestión de tiempos van mucho más allá del posible uso para su control. Sin embargo, a la larga el agente termina valorando la información de sus registros de tiempos y el análisis que puede hacerse de ella. Es fundamental proporcionar a cada agente toda la información de tiempos que de él recolecta el sistema.
- Evitar confundir la gestión de tiempos con el reloj para fichar entrada y salida de la empresa. El sistema no pretende que el tiempo registrado mensualmente sea igual al tiempo contractual, aunque podría llegar a serlo sería demasiado molesto tener que registrar tiempos para cada tipo de tarea posible de desarrollar. Por esto es importante definir distribuciones de dedicación de los agentes a un producto y así acotar el ámbito de tareas para las cuales es relevante registrar tiempos. Por otra parte, el jefe de proyecto debe reservarse un margen respecto de las estimaciones de los agentes según los antecedentes que vaya teniendo respecto de la precisión de las estimaciones en comparación con los tiempos reales.
- Introducir la disciplina de planificación y gestión de tiempos en la rutina de trabajo de los agentes. En término de herramientas esto exige que el agente tenga en todo momento en ejecución TUNE-UP Process Tool, lo cual en un principio no era apreciado. Pero en la medida que la herramienta fue incorporando funcionalidad respecto a planificación y seguimiento del trabajo, comunicación y coordinación entre agentes, espacio compartido de documentos, etc., se ha transformado en un apoyo imprescindible para el trabajo de los agentes. Un elemento muy importante respecto de la comodidad para los agentes fue la instalación de dos monitores en cada puesto de trabajo, esto les permite tener las ventanas de la herramienta en un monitor y utilizar el otro para el trabajo específico según sus roles.

TUNE-UP ha evolucionado en un contexto de una PYME de desarrollo de software bajo el marco de varios proyectos universidad-empresa. Comenzamos con la modificación del proceso centrándolo en pruebas de aceptación y fuimos

paulatinamente desarrollando la herramienta de apoyo TUNE-UP Process Tool. Esta herramienta con sus principales funcionalidades para gestión de tiempos comentadas en este trabajo lleva 3 años en funcionamiento y satisface las necesidades de desarrollo y mantenimiento de un producto de tamaño considerable (un ERP dirigido al sector socio-sanitario) y de cinco productos más pequeños, extensiones de dicho ERP. En el ERP trabaja un equipo con 2 analistas, 4 testers y 6 programadores, y en las extensiones trabajan equipos más pequeños que cuentan con 1 analista, 1 tester y 1 programador. En el desarrollo del ERP se realizan versiones de alrededor de un mes, las cuales incluyen entre 50 y 100 solicitudes de cambio.

TUNE-UP Process Tool es en sí misma también otro proyecto de la empresa, que sigue la metodología TUNE-UP y utiliza la herramienta. En el desarrollo de TUNE-UP Process Tool participan 2 programadores. Actualmente tenemos definidos 20 workflows, unos específicos para un determinado producto, otros compartidos entre diferentes productos, unos con pocas actividades, otros con hasta 30 actividades. Cada producto utiliza alrededor de cinco workflows.

Hemos comenzado a implementar refinamientos en el cálculo de la holgura de los agentes con la intención de proponer un ordenamiento orientativo de las actividades y fechas límite para el comienzo o finalización de actividades. Sin embargo, de antemano sabemos que en el contexto de desarrollo de software dicha precisión, en la práctica, no constituye una mejora sustancial. Resulta más conveniente y sencillo que los agentes organicen por sí mismo el trabajo que puedan tener en paralelo, llevando a la vez varias actividades y respondiendo con agilidad al dinamismo del proyecto.

Respecto de las estimaciones estamos estudiando mecanismos complementarios al “juicio del experto”. Con el gran volumen de datos recolectados estamos evaluando la utilización de técnicas de minería de datos para predecir el tiempo de realización de una actividad, basándose en métricas de tamaño de la unidad de trabajo (por ejemplo número y tipo de pruebas de aceptación, número y tipo de elementos de interfaz implementados o modificados, etc.).

Finalmente quisiera destacar la experiencia profesional que ha significado para mí el desarrollo de esta tesis. Durante más de un año he estado trabajando con un equipo de trabajo desarrollando y mejorando la metodología TUNE-UP. Nos hemos enfrentado a un marco de trabajo en el que cada 3 semanas realizábamos entregas de funcionalidad

de la herramienta TUNE-UP Process Tool. Además, hemos podido convivir con los usuarios de la herramienta, y ofrecerles el soporte, entrenamiento y apoyo necesario, hemos convivido dentro de un entorno real de empresa. La implantación de la gestión de tiempo en TUNE-UP ha sido difícil puesto que nos hemos encontrado con muchos agentes reacios al cambio y preocupados por el control que esto suponía sobre ellos. Actualmente, tanto la herramienta como la metodología están fuertemente implantadas y resultan indispensables para organizar el trabajo de los agentes.

Además del resultado práctico de esta tesis en cuanto a la implantación de la propuesta de gestión de tiempos se han generado dos publicaciones de investigación directamente asociadas a este trabajo, ellas son:

- Marante M, Letelier P, Suarez F. *TUNE-UP: Seguimiento de proyectos software dirigido por la gestión de tiempos.* XIV Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2009). San Sebastián, Septiembre del 2009.
- Marante M, Letelier P, Suarez F. *TUNE-UP: Un enfoque pragmático para la planificación y seguimiento de proyectos de desarrollo y mantenimiento de software*. I Congreso Iberoamericano SOCOTE -Soporte al Conocimiento con la Tecnología Universidad Politécnica de Valencia (SOCOTE 2009). Valencia, Noviembre del 2009.

Tanto los trabajos de investigación como el desarrollo de esta tesis han motivado a la autora a continuar su trabajo de investigación relacionado con la gestión de proyectos software y la mejora de procesos de desarrollo.

Referencias

1. Abrahamsson P, Warsta J, Siponen M, Ronkainen J. New directions on agile methods: a comparative analysis. Proceedings of the 25th international conference on software engineering (ICSE'03). Portland, Oregon: 2003.

2. Ambler S. *Agile Modeling*. New York: John Wiley, 2002.

3. Atkinson R. *Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria*. Elsevier Science Ltd and IPMA, 1999. International Journal of Project Management Vol. 17, No. 6, pp. 337 - 342.

4. Beck K. *Extreme Programming Explained. Embrace Change*. Pearson Education, 1999.

5. Beck K. *Extreme Programming Explained: Embrace Change*. Reading, Massachusetts: Addison-Wesley, 2000.

6. Berander P, Damm L, Eriksson J, Gorschek T, Henningsson K, Jönsson P, Kågström S, Milicic D, Mårtensson F, Rönkkö K, Tomaszewski P. *Software quality attributes and trade-offs*. Blekinge Institute of Technology, 2005.

7. Charette R. *Software engineering environments: concepts and technology*. New York: McGraw-Hill, Inc., 1986.

8. Chrissis MB, Konrad M, Shrum S. *CMMI®: Guidelines for Process Integration and Product Improvement*. Boston: Addison-Wesley, 2003.

9. CMMI Product Team. *CMMI for Development, Version 1.2.* Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.

10. CMMI v1.2. *Capability Maturity Model Integrated*. SEI–Software Engineering Institute. Carnegie Mellon: 2009. www.sei.cmu.edu/cmmi/models/ (visitado junio de 2009).

11. Cockburn A. *Agile Software Development.* Reading, Massachusetts: Addison Wesley Longman, 2001.

12. Derniame JC, Kaba BA, Wastell D. *Software process: principles, methodology, and technology*. Lectures Notes in Computer Science. Verlag, Berlin, Heidelberg, New York: Springer, 1999.

13. DSDMConsortium. *Dynamic Systems Development Method, version 3*. Ashford, Eng.: DSDM Consortium, 1997.

14. Hamilton J., *Modern Times Series Analysis,* New York: Wiley and Sons, 1996.

15. Henderson JC, Cooprider JG. *Dimensions of I/S Planning and Design Aids: A Functional Model of CASE Technology*. Information Systems Research, 1990. Vol. 1, No. 3, pp. 227-254.

16. Highsmith J. *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. New York, NY: Dorset House Publishing, 2000.

17. Highsmith J. *Agile project management: Principles and tools.* Agile Project Management, 2003. Vol. 4 No. 2. Cutter Consortium

18. Highsmith J. *Agile Project Management*. Addison-Wesley, 2004.

19. Hiranabe K. *Kanban Applied to Software Development: from Agile to Lean*. www.infoq.com/articles/hiranabe-lean-agile-kanban (visitado Junio de 2009)

20. Humphrey WS. *Managing the Software Process*. Reading MA: Addison-Wesley, 1989.

21. Hunt A, Thomas D. *The Pragmatic Programmer*. Addison Wesley, 2000.

22. Hyun J, Sun J, Ho M. *Extracting the workflow critical path from the extended well-formed workflow schema*. Journal of Computer and System Sciences. Academic Press, Inc., 2005. Vol. 70 pp: 86 –106.

23. Ilivari J. *Why are CASE tools not used?* Communications ACM, 1996. Vol.39 No. 10 pp: 94–103.

24. Institute of Electrical and Electronics Engineers (IEEE). *IEEE Std 1490 – 2003. Guide to the Project Management Body of Knowledge*. 2003

25. ISO-International Organization for Standardization. *ISO/IEC 15504, SPICE: Software Process Improvement and Capability dEtermination*. 1998.

26. ISO-International Organization for Standardization. *ISO/IEC 90003, Software engineering-Guidelines for the application of ISO 9001:2000 to computer software*. 2004.

27. Kamatar J, Hayes W. *An experience report on the personal software process.* IEEE Software, 2000. Vol.17, No. 6, pp. 85-89.

28. Karner G. *Metrics for Objectory*. Diploma thesis, University of Linköping, Sweden, 1993.

29. Kroll P., Kruchten P., Booch G. *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP*. Addison-Wesley Professional. 2003.

30. Letelier P, Penadés C, *Metodologías ágiles para el desarrollo de Software: eXtreme Programming (XP)*, Universidad Politécnica de Valencia, 2006.

31. MAP-Ministerio de Administraciones Públicas de España. *Métrica Versión 3, Metodología de planificación, desarrollo y mantenimiento de sistemas de información*. 2005. www.csi.map.es/csi/metrica3/ (visitado en junio de 2009)

32. Microsoft Office. *Ayuda Microsoft Office Project 2007*. http://office.microsoft.com/es-es/project/ (visitado Septiembre 2009)

33. Nerur S, Mahapatra RK, Mangalaraj G. *Challenges of migrating to agile methodologies.* Communications of the ACM, 2005.

34. Palmer SR, Felsing JM. *A Practical Guide to Feature-Driven Development*, 2002.

35. Paulk MC, Weber CV, Curtis B, Chrissis MB. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Boston: Addison-Wesley, 1995.

36. Poppendieck T, Poppendieck M. *Lean Software Development*. Addison-Wesley, 2003.

37. Poppendieck T, Poppendieck M. *Implementing Lean Software Development*. Addison-Wesley, 2006.

38. Project Management Institute. *Guía de los fundamentos para la dirección de proyectos*. Newtown Square, PA: PMI 3ª edición, 2004.

39. Project Management Institute (PMI) Standards Committee. *A Guide to the project Management Body of Knowledge Exposure Draft*. Upper Darby, PA: Project Management Institute, 1994.

40. Putnam L. H., Myers, W. *Measures for Excellence-Reliable Software on Time, within Budget*. U.S.A: Prentice-Hall Inc., 1992.

41. Rodrigues A. *The role of system dynamics in project management: a comparative analysis with traditional models*. International Journal of Project Management. Elsevier Science Ltd, 1996. Vol. 14 No. 4 pp. 213-220.

42. Sakata A. *Niko-niko calendar*. http://www.geocities.jp/nikonikocalendar/index_en.html (visitado Octubre 2009)

43. Schwaber K., Beedle M. *Agile Software Development with Scrum*. Prentice Hall, Series in Agile Software Development. 2001.

44. Vessey I, Sravanapudi A.P. *CASE tools as collaborative support technologies*. Communications of the ACM archive, 1995. Vol. 38, No. 1, pp. 83 – 95.

45. Watts, H. *Introducción al proceso de software personal*. Traducido por Pearson Education S.A. Addison-Wesley Iberoamericana. 2001.