Técnicas avanzadas para la gestión de proyectos *software*

Javier Navascués Fernández Victorio, 29.737.751Q

jnavascues@us.es

Supervisado por la Prof. Dra. Isabel Ramos Román

Trabajo remitido al Departamento de Lenguajes
y Sistemas Informáticos de la Universidad de Sevilla
para la obtención del Diploma de Estudios Avanzados.
(Informe de Investigación)

Índice general

Índice de figuras

Índice de cuadros

Agradecimientos

A Paulita que ha cargado toda la bibliografía A Manuel, Juan y Kevin por su ayuda. A Isabel Ramos y Miguel Toro por su paciencia y buenos consejos. A Paula por muchos motivos que no caben aquí.

Resumen

El presente informe de investigación presenta los resultados de una revisión y posterior análisis de la literatura en dos campos diferenciados: la gestión de proyectos y la simulación de procesos *software*. El objetivo es identificar los modelos y herramientas del primero de los dos campos para su empleo en relación con el segundo.

La finalidad de este trabajo es servir de base para la implementación de un modelo de simulación de *la producción de* software *a medida en un entorno multiproyecto*. El entorno multiproyecto en este caso es muy relevante según todas las referencias tanto en el campo de la producción de *software* como en de la gestión de proyectos en general. Esto se debe a que la complejidad inherente a la gestión del desarrollo del *software* se multiplica extraordinariamente en presencia de varios proyectos simultáneos.

Dado que la gestión multiproyecto ha sido estudiada desde hace tiempo por la disciplina de la gestión de proyectos, es oportuno evaluar las aportaciones que ese campo puede hacer al problema que nos ocupa. En este sentido, el trabajo que se presenta consigue identificar metodologías y modelos para descomponer jerárquicamente el problema multiproyecto, generar planes en condiciones de limitación de recursos y modelar y hacer frente al riesgo y la incertidumbre.

El trabajo realiza una revisión de los modelos de simulación del proceso *software* para seleccionar las técnicas de modelado que mejor se adapten al objetivo. Se presta especial atención a los modelos que tienen en cuenta la mejora de procesos y la implementación de las reglas y políticas de la gestión de proyectos.

A partir de aquí se puede abordar un modelo de simulación capaz de servir para ayudar a la toma de decisiones en la gestión y para comprender mejor la dinámica de interacción entre recursos, tareas, tiempo y calidad en este entorno. Dicho modelo será validado con su aplicación a un caso real.

Capítulo 1

Introducción: la producción de *software* como la gestión de proyectos

1.1. Presentación del problema

Este trabajo se enfoca en el problema de la planificación y gestión de la producción de una organización dedicada al desarrollo de *software* a medida. El resultado final esperado es poner las bases para construir un modelo dinámico de simulación de un sistema de producción multiproyecto sobre el que se puedan evaluar los siguientes aspectos:

- La planificación y el seguimiento de los proyectos desde el punto de vista de la gestión del riesgo
- La aplicación de políticas dinámicas de asignación de personal y secuenciación de actividades en relación con lo anterior
- El impacto potencial de la mejora de procesos en la controlabilidad y predecibilidad del sistema
- Las necesidades de medición y control que todo ello plantea

El presente informe de investigación tiene por objeto presentar los resultados de la primera fase del trabajo comentado. Dicha primera fase ha consistido en las tareas siguientes:

- Caracterizar el problema de la producción de *software* a medida como un problema de gestión multiproyecto
- Revisar las aportaciones desde la disciplina de la gestión de proyectos que puedan ser relevantes para el problema en cuestión
- Revisar las propuestas existentes de modelado y simulación de la producción de *software* para identificar las que sean más adecuadas al problema que se quiere modelar
- Identificar los desarrollos posteriores necesarios para alcanzar el fin previsto

1.2. Producción de *software* a medida

El enfoque de este trabajo se dirige a organizaciones de *tamaño medio*, dedicadas al desarrollo, implantación y mantenimiento de sistemas de información bajo pedido. La acotación, por tanto, se produce en dos planos: el producto y el tamaño.

En cuanto al *producto*, el trabajo no tiene en cuenta la producción de software denominado "comercial" que, si bien tiene características en común con el segmento seleccionado, presenta especificidades tanto en el proceso productivo como en el ciclo de vida del producto que escapan de la investigación realizada. Lo mismo ocurre con el software especializado de comunicaciones, control de procesos en tiempo real, y otras aplicaciones en el que las similitudes que indudablemente existen no pueden obviar el peso de los determinantes técnicos que la especialidad impone.

En cuanto al *tamaño* de la empresa, se enfoca lo que comúnmente se conoce como PYME, dejando a un lado las microempresas (pequeños grupos de programadores, consultores independientes, ...) y las grandes empresas, con más de 300 empleados.

El ámbito objeto del estudio pues, viene a corresponderse con el subsector empresarial de pequeños y medianos productores independientes bajo pedido de software de gestión y de almacenamiento, tratamiento y difusión de información. El tipo de producto y el tamaño de empresa fija unos límites al número máximo de productos simultáneos que se pueden estar desarrollando, lo que a su vez permite acotar la dimensión de los problemas a modelar.

1.2.1. Relevancia del sector en Andalucía

En Andalucía, el sector organizado según la Asociación de Empresarios de Tecnologías de la Información y Comunicaciones de Andalucía, ETICOM. (`www.eticom.org`) de las empresas de las Tecnologías de la Información y las Comunicaciones (TIC) supone más de 5.500 trabajadores y casi 300 empresas con una facturación próxima a los 500 millones de euros. En estas cifras se incluyen tanto ventas de hardware, como de software "comercial", servicios informáticos, consumibles, cánones y otros por lo que a la hora de estimar el volumen de producción del subsector potencialmente interesado en el resultado de este trabajo se pueden dar los siguientes datos: Los "servicios informáticos" se puede estimar en torno a un 50 % del volumen de negocio total de las TIC, aplicando a Andalucía los datos a nivel nacional del Ministerio de Ciencia y Tecnología. De este 50 %, la mayor parte son servicios de procesado y almacenamiento de información, tratándose pues de servicios de un relativamente bajo valor añadido. Limitando a un 10 % la producción de software bajo pedido, podemos estimar la producción andaluza unos 2'5 millones de euros en Andalucía. Esta facturación es, en su casi totalidad, valor añadido.

El mercado europeo se servicios informáticos referido a EUR-15 asciende a 233.000 millones de euros, con un valor añadido ligeramente inferior al 50 % del total y ocupa a unos dos millones de personas. A nivel nacional, las cifras se reducen a unos 11.000 millones de euros con un valor añadido de unos 4.900 y algo más de 133.000 personas empleadas. Dado que se trata de un mercado de servicios, el mercado exterior es relativamente bajo. A nivel nacional, la importación es inferior al 7 % de la demanda y la exportación inferior al 4 %. Evidentemente en lo que se refiere al mercado andaluz, la importación de otras comunidades autónomas sí es una cifra relevante.

1.3. Modelos de simulación y modelos normativos

1.3.1. Simulación continua y discreta

La aplicación de modelos de simulación al estudio de diversos problemas relacionados con la gestión de los proyectos de desarrollo de productos *software* es un campo al que se dedica un significativo esfuerzo de investigación.

La razón para ello es doble: de una parte la necesidad de hacer más predecible y controlable una actividad que cada vez tiene más importancia; de otra, la naturaleza compleja y no lineal del problema.

Una parte muy importante del trabajo que se ha realizado y que se sigue realizando se basa en dos metodologías diferentes y complementarias. La primera es la Dinámica de Sistemas, la segunda es la simulación de evetos discretos. Ambas metodologías se prestan específicamente a representar y estudiar facewtas diferentes del sistema. La primera permite analizar la estructura de interrelaciones complejas entre las diferentes variables de estado del sistema en estudio, especialmente cuando estas relaciones son fuertemente no lineales y los métodos analíticos resultan poco adecuados. Una organización dedicada al desarrollo de productos *software* es un sistema sociotécnico complejo,en los que las interacciones humanas, los flujos de comunicación y la presencia de objetivos diferentes y, hasta contradictorios, requieren este tipo de representación.

La metodología orientada a eventos discretos posibilita el estudio del ciclo de vida de entidades individuales diferenciadas y su circulación a través de las diferentes componentes del sistema. La producción de software está plagada de este tipo de entidades: desde el propio producto, que no es siplemente un "bloque de código" más o menos grande sino una unidad con su funcionalidad propia y diferente de la de otros productos, hasta la multiplicidad de entregables y otro tipo de artefactos que se generan en la producción de software. Los propios recursos que se emplean en la producción no son simplemente acumulables; el recurso principal son personas, cada una con sus diferentes capacidades, conocimientos y habilidades. Si bien el tratamiento del ciclo de vida de objetos individuales en un sistema es objeto de estudio desde hace mucho tiempo, aquí lo mismo o más que en el análisis de relaciones estrcuturales en los sistemas continuos, los métodos analíticos no son suficientes para tratar suficientemente el problema. Es por tanto otra metodología candidata natural a la simulación.

1.3.2. Modelos normativos

La simulación no es la única aproximación a los problemas que plantea la producción de *software*. Desde diversas disciplinas se postulan modelos normativos orientados a responder a la pregunta: ¿Cómo se debe llevar adelante un proyecto? En este trabajo nos interesan dos grandes líneas que intentan responder a esta pregunta. De un lado esta el amplio campo de la gestión de

proyectos, *project management*, y en particular las aplicaciones de modelos y métodos de la Investigación Operativa, enriquecidas en las últimas décadas con métodos provenientes de la Inteligencia Artificial. Del otro lado está el pujante terreno de la *ingeniería del software* y, dentro de él, la *ingeniería de procesos software* y las propuestas para definir, estudiar, gestionar y controlar la compleja red de actividades que comprende la producción de software.

El interés en estudiar el proceso software nace de la necesidad de mejorarlo. La idea que subyace es que un proceso mejorado da lugar a un mejor producto. Un éxito que sólo se basa en la disponibilidad de unas personas específicas no constituye una base para mejorar la productividad y la calidad a largo plazo en todo el ámbito de la organización. Para extender a toda la organización esta capacidad, el paradigma vigente en materia de gestión de la calidad se basa en el supuesto de que la calidad del producto está directamente vinculada a la calidad de los procesos que conforman su ciclo de vida y estos, a su vez, a la madurez de la organización que los desarrolla. En este campo, los resultados actualmente más operativos son los estándares de madurez. Los estándares de madurez en la producción de software son tecnología de dominio público a escala internacional difundida a través de diferentes instituciones y aplicadas con carácter general, en particular en grandes organizaciones. La aplicación de estos estándares a las PYMEs es objeto de in-vestigación, tanto mediante el estudio de casos como mediante las construcción de propuestas metodo-lógicas que tengan en cuenta las características específicas de este tipo de empresas .

Estos estándares orientados a las capacidades a nivel de organización (CMM) o a nivel de proceso (ISO 15504) proporcionan guías para la evaluación que señalan objetivos genéricos para la mejora de la calidad de los procesos, si bien no indican vías unívocas para su puesta en aplicación en un caso concreto. La razón de ello se en la gran variedad de tecnologías, modelos de desarrollo y tipologías organizacionales existentes determina el carácter dependiente del camino (*path dependent*) de las estrategias posibles de mejora, incluso aunque se utilice un modelo estándar como referencia.

1.4. El problema de la gestión de la producción de software a medida: dos perspectivas

El proceso de negocio en las organizaciones objeto de este trabajo puede describirse desde el siguiente esquema básico:

- la organización es, básicamente y en primer lugar, un conjunto de recursos especializados con una capacidad productiva, un nivel tecnológico y unos procedimientos operativos determinados.

- la organización capta pedidos a través de su acción comercial, que se convierten en proyectos a desarrollar e implementar.

- la organización planifica la producción de esos proyectos asignándoles recursos para su ejecución en función de su capacidad, de los compromisos adquiridos con los clientes y de la carga global de trabajo.

- la organización dispone de un sistema de control interno a través del cual intenta gestionar la ejecución y modificar las asignaciones para hacer frente a las perturbaciones externas o internas que se produzcan en el curso de la misma.

- la organización lleva a cabo otros procesos de infraestructura y apoyo al proceso principal dentro de los cuales es crítica la actualización constante de la capacidad productiva vía el reclutamiento y la formación de nuevos recursos humanos.

Este esquema se ve afectado de características específicas del tipo de producto que lo diferencian de otras actividades productivas:

- La primera es que, a diferencia de otros negocios, el resultado final del proyecto, el producto, es un *artefacto evolutivo*, en el sentido de que sus características no están totalmente definidas a priori sino que se van conformando a lo largo de su propio ciclo de vida. De este modo, la propia evolución del producto es fuente de perturbaciones que afectan a la definición del proceso pendiente y, por tanto, obliga a la revisión constante de la planificación.

- La segunda, que es consecuencia de la anterior, es que no *es fácil determinar cuando ese proceso se acaba*. El momento de la implementación y puesta en operación de un sistema no es nunca el de la entrega final, sino un hito más en la evolución del sistema. Junto a la corrección de errores imputables a las fases anteriores aparecen nuevas perturbaciones posibles derivadas de la integración con otros sistemas, de la incorporación de bases históricas, de cambios en las operativas de usuario, ...

Como consecuencia de todo ello, los tres objetivos que corrientemente se señalan a un proyecto; *plazos*, *coste* y *calidad*, son mutuamente contradictorios en la dinámica de su propia ejecución.

1.4.1. La perspectiva de la gestión de proyectos

El problema de conjugar plazos, costes y calidad es un problema común en la gestión de proyectos. Desde un cierto punto de vista, el proceso de producción de *software* a medida puede verse como el de un *proyecto unitario*; es decir un conjunto de actividades y tareas lógicamente ordenadas a la obtención de un fin único. Desde ese punto de vista, la gestión de la producción de este tipo de software es fundamentalmente una *aplicación especializada de la gestión de proyectos unitarios*.

Desde otro punto de vista, el problema crucial de la gestión de la producción es la administración de los *recursos* de la organización por cuyo uso "compiten" varios proyectos concurrentes. Estamos, pues, ante un problema de gestión *multiproyecto* en un entorno de *restricciones de capacidad y de plazos*.

Por otra parte, las perturbaciones que se detectan en la ejecución de un proyecto pueden venir inicialmente ocasionadas por problemas específicos de ese proyecto - como típicamente ocurre con la volatilidad de los requisitos u otras razones ligadas a la relación con el cliente -, pero la respuesta a esas perturbaciones se suele traducir en presiones sobre la asignación de recursos a esas tareas, las cuales se transmiten al resto de proyectos por efecto de la restricción global de recursos.

Esta *realimentación* entre un entorno de trabajo sujeto a variaciones debidas al propio proyecto y a las interrelaciones de este con otros que se estén desarrollando simultáneamente es una fuente de complejidad de la gestión de la producción de software en la práctica.

A la *complejidad relacional* del problema así planteado debe añadirse la

complejidad intrínseca del trabajo que subyace a las actividades que configuran un proyecto software: dependencia del factor humano, divisibilidad limitada de las tareas, problemas de falta de homogeneidad de los recursos, no-linealidad en la productividad, ...

1.4.2. La perspectiva de la ingeniería del software

Una perspectiva complementaria parte de la hipótesis de que en una organización los proyectos están formalmente constituidos por actividades que ofrecen - o pueden ofrecer - un grado significativo de similaridad y repetitiblidad . El esfuerzo desde este punto de vista se enfoca en la definición de *proceso* como unidad repetible y, por tanto, mejorable. en su articulación interna y susceptible de programarse en relación con otros procesos.

Este es el punto de vista de la *ingeniería de los procesos software*, un área de conocimiento del campo de la *ingeniería del software* que se orienta a la aplicación de métodos y modelos para conocer, racionalizar y en último extremo mejorar la gestión de las actividades que forman parte del ciclo de vida de un producto software. Desde el punto de vista que nos interesa, se trata de una aplicación a la ingeniería del software del paradigma de la reingeniería de procesos y de la mejora continua.

Dentro del campo de la ingeniería de los procesos software, este trabajo se centra en el modelado de procesos software con fines descriptivos, orientados a la simulación y eventual optimización o, al menos, evaluación de las mejoras del proceso [Sca01].

1.5. El caso de aplicación

A fin de validar el trabajo aquí propuesto, se utilizará como caso de aplicación una empresa con una plantilla de unas 50 personas dedicada al desarrollo de software para grandes clientes - instituciones públicas, sobre todo - en tres áreas: sistemas de soporte a la gestión administrativa, sistemas de información y comunicación a través de la www y sistemas distribuidos de adquisición y transmisión de datos para el mantenimiento de sistemas de información georreferenciada.

El caso de aplicación presenta las siguientes ventajas:

- Se trata de una empresa representativa del sector objetivo

- Se conocen los procedimientos actuales de planificación y programación del trabajo

- La empresa está certificada según el estándar ISO 9001:2000 y en los momentos actuales está en proceso de obtener la acreditación CMMI nivel 2, por lo que existe motivación favorable a este trabajo desde la dirección

- Posee un registro histórico de cargas de trabajo en el desarrollo de más de un centenar de proyectos

Esta tercera faceta es la que se considera más interesante para validar empíricamente los resultados que se alcancen en este trabajo pues la mayoría de los trabajos de investigación de los que se tiene referencia, o bien se basan en datos de organizaciones muy diferentes - y geográficamente alejadas - o bien se basan en "evidencia empírica" obtenida mediante ejercicios académicos o modelos de simulación.

En nuestro caso, y desde hace más de diez años se dispone de una aplicación desarrollada por la misma organización para el seguimiento y análisis de costes de producción. La organización comercializa la mayoría de sus servicios mediante el sistema de asistir a concursos de ofertas competitivas, en los que la oferta económica tiene un peso muy importante a la hora de obtener un contrato. De ahí que considere crítico el conocimiento del coste real de los contratos que consigue.

La información disponible está organizada en un sistema relativamente complejo que controla aspectos de facturación, control de pagos, seguimiento de expedientes de concurso y otros aspectos comerciales y administrativos que progresivamente van evolucionando hacia un sistema de CRM.

Incluido en el sistema está también un subsistema de imputación de costes a los diferentes contratos en el que se distinguen los costes derivados de suministros (equipos, software, edición de soportes, ...) y, lo que interesa más a este trabajo, los costes de desarrollo. Salvo tareas muy especializadas, fundamentalmente de diseño gráfico, y tareas mecánicas de digitalización de imágenes, la empresa no subcontrata al exterior por lo que los costes de desarrollo son los costes de personal de la empresa.

Es precisamente este subsistema el que interesa a efectos de este trabajo. Las bases de datos disponibles deben depurarse y normalizarse, tareas que también se incorporan a este trabajo.

1.5.1. Descripción general del caso

El caso de aplicación es una organización de tamaño mediano y mercado regional. La mayoría de sus productos son aplicaciones desarrolladas a medida que se incorporan a sistemas de información que van evolucionando con las sucesivas incorporaciones.

En algunos casos, los sistemas han sido creados a partir de aplicaciones *ex-novo* desarrolladas por la propia organización u otros competidores. En otros casos se trata de la traslación a tecnologías más actuales o la incorporación de nuevas funcionalidades en sistemas preexistentes.

La mayoría de los sistemas son, a su vez, componentes informatizados de sistemas de información de carácter administrativo, es decir, deben integrarse en sistemas administrativos regulados por procedimientos reglamentarios y acomodarse a los mismos. Dado que los clientes son básicamente instituciones y administraciones públicas, los requisitos que esto impone son relativamente rígidos.

Dentro del esquema general del ciclo de vida, tal y como se establece en la norma ISO-IEC 12207, la organización interviene fundamentalmente en los procesos principales de Desarrollo y Mantenimiento y en los procesos de soporte asociados. La política de las administraciones públicas de separar la adquisición de hardware e incluso de normalización del software básico - SGBD, por ejemplo - excluye, generalmente a la organización de los procesos de suministro y adquisición. Los condicionantes del proceso administrativo determinan rígidamente los procesos de la organización.

La singular mecánica de la contratación pública da lugar a un ciclo de vida del producto basado en contratos - denominados"proyectos" por la organización que pueden referirse a un sistema o a varios, y en el primer caso, a un único bloque de funcionalidades o a varias.

Otro elemento digno de destacarse es la peculiaridad de la acción comercial que da lugar a que fases formales del ciclo de desarrollo, como las que genéricamente se recogen en la fase EVS de la metodología MÉTRICA, deban acometerse con anterioridad a la puesta en marcha de los correspondientes contratos-proyectos. Algo que formalmente podría considerarse como un coste de la acción comercial pero que realmente constituyen costes de desarrollo que se incurren asumiendo el riesgo de que no sea posible obtener el correspondiente contrato.

La limitación presupuestaria en la que operan los clientes es otro factor determinante tanto en las condiciones de competencia a la hora de conseguir

un contrato como en las circunstancias en las que se desarrolla el ciclo de vida de un producto. En primer lugar, la competencia en precios es determinante, puesto que, asegurando unos niveles de competencias y conocimiento del problema exigibles, el coste del producto es la variable de decisión principal de los clientes. En segundo lugar, la existencia de planes de desarrollo de sistemas de proyección mayor no evita la mecánica presupuestaria de contratación de realizaciones parciales de esos planes, realizaciones parciales que, además, suelen tener que ajustarse a periodos anuales. Una práctica nada inhabitual, por tanto, es la de presentar ofertas que para justificarse deben contemplarse en conjunción con las expectativas de obtener otros contratos posteriormente.

Un último factor relevante es la presión de los plazos, que en el caso de los clientes institucionales no sólo implica riesgos de cara a futuros contratos sino que está a su vez determinada por la presión de los plazos que impone el ciclo de ejecución presupuestaria. Como cabe pensar, la naturaleza del tipo de contratos y clientes da lugar a un coste de financiación del desarrollo nada despreciable que tiene que contemplarse en función del periodo de pago típico de estos clientes.

1.5.2. La dinámica del caso de estudio a largo plazo

La dinámica en el largo plazo del caso de estudio viene gobernada por el progresivo incremento del consumo de TIC por parte de las administraciones públicas, por la adaptación de la capacidad productiva a ese consumo creciente y por el seguimiento constante de la innovación tecnológica, lo que permite mantener y hacer crecer la cuota de mercado.

El incremento progresivo del consumo de TIC por parte de las administraciones públicas es un hecho conocido. Si bien las oscilaciones del ciclo económico modifican el perfil, sin embargo no han invertido la tendencia que - previsiblemente - seguirá siendo creciente en el medio plazo. Otro tanto puede decirse del ciclo"político" provocado por los cambios en los máximos niveles de responsabilidad de los organismos clientes que se producen cuando hay procesos electorales. Una de las fortalezas de una organización como es la del caso de estudio es la capacidad, una vez que ha alcanzado una cierta cuota de mercado, de influir en que esa tendencia le beneficie lo más posible gracias a su conocimiento de las necesidades del cliente, si es capaz de mantener un dominio de la tecnología suficiente.

El ajuste de la capacidad de producción a esta tendencia, en el caso objeto de estudio, ha obligado a resolver varios problemas comunes en el sector.

Las oscilaciones en la demanda en el plazo corto son importantes, aunque se producen con cierta regularidad. Existe un efecto "estacional" ligado al propio ciclo de ejecución presupuestaria. Literalmente, existen momentos del año en los que "hay que hacer ofertasz momentos del año en los que "hay que entregar". Una estrategia de ajuste de la capacidad productiva basada en la rotación del personal presenta serios inconvenientes en una organización en la que la capacitación y la familiaridad con los procedimientos internos es crucial. Una estrategia de plantillas inamovibles a medio plazo da lugar a costes excesivos y a corto plazo ocasiona tensiones financieras.

La estrategia adoptada incluye, por una parte, alcanzar un grado de autofinanciación suficiente para eludir esas tensiones financieras; por otra, dedicar la capacidad ociosa a mejorar la organización y sus condiciones para seguir siendo competitiva; y, por último, el recurso a personal en formación - estudiantes en prácticas, becarios, ...- lo que, además de reducir los costes permite formar un amplio *pool* de personal cualificado que puede pasar a incorporarse a la plantilla progresivamente, regulando así la capacidad productiva de fondo a la tendencia. Esta estrategia, que hasta ahora se ha visto como adecuada, se complementa con una ventaja adicional desde el punto de vista comercial y es que una parte de ese personal joven formado en la empresa se incorpora posteriormente a trabajar con clientes actuales o potenciales.

En cada ocasión de descenso en el ciclo económico general, o coincidiendo con la difusión de una innovación tecnológica importante o la implantación de unos criterios de estandarización concretos, el sector de la oferta ha experimentado una profunda reorganización. Lo que inicialmente era un sector poblado por múltiples pequeñas empresas ha ido progresivamente convirtiéndose en un conjunto mucho más limitado de competidores - aunque siguen existiendo nichos de mercado "de proximidad" - donde grandes organizaciones obtienen cada vez más cuota de mercado. Es evidente que mantener el paso con la innovación tecnológica o normativa es una condición necesaria para no retroceder. Pero junto con esto, la flexibilidad y la capacidad de responder a las demandas del cliente compitiendo en precio siguen siendo una ventaja diferencial que puede mantenerse y acrecentarse como demuestra la trayectoria pasada del caso de estudio.

Si bien no es objeto de este trabajo entrar en los aspectos estratégicos, si conviene poner de relieve que una adecuada gestión de los proyectos en el corto plazo permitirá a la empresa mejorar sus factores competitivos en el medio y largo plazo.

1.5.3. El proceso actual de gestión de proyectos

El proceso actual de gestión de los proyectos, entendiendo por estos normalmente el conjunto de actividades incluidas en un único contrato, se basa en unas reglas muy simples que combinan la elaboración de previsiones de plazos y recursos necesarios para completar un proyecto con el control del estado de empleo de los recursos en cada momento.

Inicialmente se designa un Jefe de Proyecto que elabora una estimación del programa de desarrollo del proyecto con base en su experiencia previa.

Dicho programa incluye unas demandas a plazo de determinados recursos durante determinados periodos de tiempo. Con base en esas demandas se constituye un equipo para el desarrollo del proyecto asignándose personas al mismo, personas que entrarán a trabajar en el proyecto en los plazos estimados y por el periodo de tiempo igualmente estimado.

Esto proporciona una programación del empleo de los recursos humanos en el tiempo y permite, por exclusión, ir asignando recursos a proyectos que vayan apareciendo posteriormente.

Semanalmente se evalúa el cumplimiento de los programas y se actualizan las previsiones, de modo que se pueden producir reasignaciones de recursos a tareas eventuales o a acelerar el cumplimiento de las previsiones. Los recursos ociosos se destinan a tareas internas, salvo cuando estas son repetitivas - en cuyo caso la capacidad utilizada se descuenta de la total - o de magnitud tal que son consideradas como un proyecto más. Con este heurístico tan simple se acomoda la producción a la capacidad y se regula - en último extremo - el flujo de las operaciones en la empresa.

1.5.4. El sistema de control de gestión actual

El sistema de control de gestión implantado tiene por objetivo principal la determinación del coste de los contratos que realiza la empresa con sus clientes en relación con el coste presupuestado en la oferta. El coste se determina por la imputación del coste directo de mano de obra más un coeficiente de absorción de los gastos indirectos y generales predeterminado. En el caso de que el proyecto incorpore suministros de equipos o cualquier otro elemento adquirido en el exterior - incluso la subcontratación de partes del desarrollo - se acumulan al coste así determinado.

Para ello la organización mantiene un sistema de bases de datos que se orienta, fundamentalmente, a la estimación del coste total de un proyecto. El

sistema contiene mucha información sobre aspectos administrativos, clientes, ... y, de hecho alguna de sus funcionalidades evoluciona hacia un sistema CRM. Sin embargo, a los efectos de este trabajo la información relevante es el registro de *esfuerzo invertido* en las tareas de desarrollo de lod proyectos de la empresa.

La captura de esta información proviene de un procedimiento implantado en la organización que obliga a cada persona a registrar semanalmente el uso del tiempo entre los distintos proyectos en los que está participando. Junto con ese cómputo del tiempo se indica también la "categoría", es decir, la función que ha realizado.

La información disponible se extiende a un periodo muy dilatado pero sólo es relativamente fiable desde 1999. El número de registros elementales trabajador×proyecto asciende a más de 40.000 con un total de varios centenares de proyectos.

En 2004, el autor de este trabajo dirigió un proyecto fin de carrera de Ingeniería de Telecomunicaciones sobre la aplicación de la técnica de los puntos-función a la estimación de las cargas de trabajo. Se desarrollo una aplicación en ORACLE para analizar los registros, depurar los errores y ajustar el esfuerzo real invertido. Aplicando la metodología de los puntos función se obtuvo un ajuste logarítmico bastante aceptable a un grupo de proyectos realizados con la misma plataforma tecnológica.

Se considera que esta información puede servir de soporte a la investigación a la que pertenece este informe.

1.6. Conclusión y propuesta de investigación

La propuesta de investigación a la que pertenece este trabajo intenta conjugar las aproximaciones reseñadas de la forma siguiente: se trata de modelar el problema de la planificación y gestión de la producción de una organización dedicada al desarrollo de software a medida para evaluar la aplicación de las técnicas y métodos de la ingeniería de organización y de la ingeniería del software a dicho problema.

El resultado final esperado es un *modelo dinámico de simulación de un sistema de producción multiproyecto.*

El modelo de simulación será de tipo híbrido por la necesidad de capturar con la granularidad necesaria procesos y sucesos de naturalezas muy diferentes. Se pretende con ello poder modelar simultáneamente:

- El carácter discreto de:
  - El ciclo de vida de un producto software caracterizado por la sucesión de fases definidas y caracterizadas cada una de ellas por sus correspondientes entregables e hitos
  - Las decisiones de asignación (y reasignación) de recursos y programación (y reprogramación) de actividades
  - El proceso de registro para la obtención de métricas necesarias para la adopción de decisiones
  - Los eventos externamente determinados o derivados de las decisiones del punto anterior que pueden afectar al ciclo de vida o a cualquiera de sus fases
- El carácter continuo de:
  - El proceso de trabajo con el correspondiente consumo de esfuerzo y la eventual generación de errores
  - El entorno general de la empresa (movimiento laboral, ciclos de capacitación del personal, desarrollo de infraestructuras de soporte a los procesos de producción, etc.)
  - La mejora de las capacidades de los procesos

Desde el punto de vista de la implementación, el modelo se basará en un formalismo del tipo DEVS propuesto por Ziegler ([ZKP00]) que posibilita representar con naturalidad un sistema híbrido al gestionar conjuntamente una simulación continua con un manejo muy flexible de la lógica de eventos y los flujos de información asociados al inicio y finalización de las actividades de un proyecto y a la monitorización de las mismas.

La primera parte de este trabajo, a la que corresponde el presente informe, consiste en una revisión y análisis de los modelos, métodos y propuestas para responder a las siguientes preguntas:

1. ¿Cómo se modela, o se puede modelar, específicamente un entorno de gestión multiproyecto desde el punto de vista normativo más allá de agregar todos los proyectos en un único *metaproyecto*?

2. ¿Cómo se toman las decisiones de asignación de recursos a actividades y proyectos para cumplir plazos y minimizar costes?

3. ¿Cómo se tienen en cuenta los factores de riesgo e incertidumbre en estas propuestas normativas?

4. ¿Qué propuestas hay de modelos de simulación de los procesos software que tengan en cuenta el carácter a la vez continuo y discreto de los mismos?

5. ¿Cómo se pueden implementar las técnicas del punto 3 anterior en esos modelos?

6. ¿En qué medida la mejora de procesos puede reflejarse y evaluarse con esos modelos?

Capítulo 2

Caracterizando el problema desde la perspectiva de la gestión de proyectos

2.1. Objetivo y contenidos

Este capítulo tiene como objetivo caracterizar el problema de la producción de *software* a medida dentro del amplio campo de los problemas de gestión de proyectos. Para ello, tras una presentación de la disciplina, de su alcance y sus limitaciones, se hace una referencia a diversos estudios sobre los factores que determinan el éxito en la gestión de proyectos.

A continuación se plantea la cuestión del entorno *multiproyecto* y de las repercusiones que implica dicho entorno en la estructura de la organización que desarrolla simultáneamente varios proyectos. El siguiente paso es establecer una tipología de situaciones multiproyecto en términos de *variabilidad* y *dependencia* con el fin de posicionar el problema en estudio dentro de esa tipología.

Una vez que se ha posicionado el problema en términos de variabilidad y dependencia, se plantea una discusión sobre los niveles de decisión en tres categorías: *estratégica, táctica y operacional*. Esta discusión posibilita relativizar y separar las causas de variabilidad y dependencia. El ámbito de decisión implica el grado de libertad de los responsables de dicho ámbito.

Dejando a un lado el nivel estratégico, a un nivel táctico se dispone de todos los recursos de la empresa para asignarlos a los diferentes proyectos. La

dependencia entre proyectos es, por tanto, total a este nivel. Según se asigne la capacidad productiva habrá más o menos probabilidades de cumplir con los plazos de entrega de los productos, plazos de entrega que se suponen fijos pues vienen asignados de partida a los proyectos que se admiten en el sistema, algo que a su vez depende de consideraciones estratégicas y que, por tanto, quedan fuera del problema.

En el nivel operacional, en cambio, la situación es diferente. El responsable de un proyecto cuenta con una asignación de recursos más restringida. Su problema es cumplir los plazos con los recursos con los que cuenta, sujeto a la variabilidad propia del proyecto - en el caso de los proyectos *software*, por ejemplo, la volatilidad de los requisitos aparece como fuente principal de riesgo - y a la variabilidad provocada por la eventual disrupción en la disponibilidad de recursos. Esta última puede obedecer a factores externos o a decisiones del nivel táctico. En este último caso, típicamente las decisiones de retirar eventualmente recursos de un proyecto para hacer frente a otro más urgente. En base a esta discusión el capítulo concluye proponiendo descomponer el problema en dos: uno *táctico* y el otro *operacional*.

2.2. La disciplina de la gestión de proyectos (*Project Management*)

2.2.1. Introducción: la actualidad de la gestión de proyectos

Un proyecto es "*un esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único*"[Ins04]. Otra definición podría ser la que proponen Shenhar y Dvir [SD07] para quienes un proyecto es "*una organización y un proceso dispuestos para alcanzar un objetivo específico bajo restricciones de tiempo, presupuesto y otros recursos*". La gestión de proyectos serían las"*actividades de gestión necesarias para conseguir el éxito del proyecto*" según estos autores, o "*la aplicación del conocimiento, habilidades, herramientas y técnicas a las actividades de un proyecto a fin de conseguir los requisitos del mismo*" según la primera referencia.

La gestión de proyectos - *project management* - es una de las disciplinas de mayor crecimiento en la actualidad dentro del área de la gestión empresarial. Ello es debido a que la propia idea de proyecto ha trascendido a su

interpretación tradicional asociada típicamente a la construcción de un producto o un sistema material singular, y se extiende a subconjuntos cada vez mayores de la actividad empresarial hasta el punto de que se habla, a veces, de una *orientación a proyectos* de todos los procesos de trabajo [MBCDH06] e incluso de la *sociedad orientada a proyectos*[Gar01].

La mejora de productos, el desarrollo de sistemas de información, la reingeniería de procesos, las campañas de marketing,... todas ellas son actividades que se abordan desde una perspectiva de proyecto. El dinamismo creciente del entorno de los negocios obliga a ciclos de vida del producto cada vez más cortos lo que determina que a su vez el desarrollo de nuevos productos sea una actividad cada vez más recurrente. Hace ya tiempo que estos procesos se enfocan como proyectos [Cho05]. Incluso ramas de actividad tradicionalmente organizadas en torno a procesos de negocios estables y rutinarios como el sector financiero y de seguros, el comercio al menor, etc. está cada vez más afectado por esta dinámica. La incorporación progresiva de las TIC en todas las áreas de negocio es una fuente de nuevos proyectos de implantación con la consiguiente modificación de los procesos empresariales. Como es lógico, la construcción de esos sistemas es en si misma un proyecto.

La creciente generalización de la orientación a proyectos da lugar a entidades agregadas y compuestas de proyectos que se denominan comúnmente *programas* y *portafolios* o *carteras de proyectos*. Convencionalmente se denomina programa a un conjunto de proyectos que tienen una finalidad común, "*un grupo de proyectos relacionados cuya dirección se realiza de manera coordinada para obtener beneficios y control que no se obtendrían si fueran dirigidos de forma individual* [Ins04].

Una Cartera de Proyectos es un conjunto de proyectos independientes unidos únicamente por el hecho de que son desarrollados por una misma empresa, lo que eventualmente conlleva el que deben compartir determinados recursos. Según el PMI,

> un portafolio es un conjunto de proyectos o programas y otros trabajos, que se agrupan para facilitar la gestión efectiva de ese trabajo, a fin de cumplir con los objetivos estratégicos de negocio. Los proyectos o programas del portafolio no necesariamente tienen que ser interdependientes o estar directamente relacionados. La recaudación y el respaldo pueden asignarse sobre la base de categorías de riesgo / recompensa, líneas de negocio específicas o tipos generales de proyectos, como la mejora de la infraestructura

y del proceso interno. [Ins04].

La gestión de programas y de carteras de proyectos plantea problemas adicionales a la simple superposición de los proyectos. Un grado superior de complejidad, derivado de las dependencias que se establecen, bien por los objetivos en el primer caso, bien por la necesidad de hacer compatible el empleo de los recursos, en ambos. En ambos casos, ese entorno más complejo lo denominaremos un *entorno multiproyecto*.

2.2.2. Los factores críticos del éxito de los proyectos

El objetivo convencionalmente definido de la gestión de proyectos es alcanzar el resultado previsto, en términos de funcionalidad y características, dentro de los plazos y los costes establecidos. El éxito de un proyecto debe medirse, pues, en términos de desviaciones de ese objetivo. Sin embargo, el número de casos de retrasos, sobrecostes, frustración de expectativas e incluso fracasos rotundos en la consecución de los fines de los proyectos registrados en la literatura es muy significativo. Casos extraordinariamente notables son los del aeropuerto de Denver [Don02], la Ópera de Sydney [Hal80] o el proyecto Iridium de Motorola [FS00]. En el campo de los proyectos de sistemas de información es muy renombrado el informe Chaos de la Standish Corporation [Cor03].

Al margen de estos casos más sonados, la investigación publicada documenta que la frecuencia de variaciones en costes y plazos es muy alta. Las razones para esta variabilidad que más se citan es muy diversa: falta de experiencia o capacitación inadecuada de los directores de proyecto; expectativas mal definidas y gestionadas; defectos de liderazgo a varios niveles; fallos en la identificación, documentación y seguimiento de requisitos; procesos de planificación deficientes; errores en la estimación de esfuerzos; diferencias culturales y éticas; desajustes entre el equipo del proyecto y la organización a la que sirve; métodos inadecuados o mal empleados; seguimiento y comunicación inadecuada; etc.

Varios autores han intentado construir un marco para clasificar los factores críticos para el éxito o el fracaso de un proyecto. Según varios trabajos de Pinto y otros [PM90, PP90] esos factores críticos caen dentro de dos grandes grupos:

- Factores asociados a la *planificación inicial*, tales como la definición de

la misión del proyecto, su programación adecuada, la corresponsabilización del cliente, ...

- Factores asociados al *entorno y la forma en la que se desenvuelve el desarrollo del proyecto*, tales como el apoyo de la alta dirección, la comunicación, la gestión de recursos y la solución de problemas

Estos estudios llegan a la conclusión de que los factores situados en primer lugar son más importantes al inicio del ciclo del proyecto mientras que los segundos van cobrando cada vez más importancia conforme dicho ciclo va avanzando. De hecho, la falta de estrategias o planes de contingencia acaban revelándose como la principal causa de fracaso en los proyectos revisados por estos estudios.

Belassi y Tudek [BT96] en lugar de analizar los factores en función de su ubicación en el ciclo de vida utilizan una clasificación de tipo relacional, que les permite diferenciar entre:

- *Factores relacionados con el propio proyecto*: tamaño y valor, singularidad de las actividades del proyecto, densidad o complejidad del mismo, ciclo de vida y urgencia o perentoriedad.

- *Factores relacionados con el equipo del proyecto*: capacidades de delegación y coordinación, competencia, nivel de corresponsabilización y compromiso, capacitación técnica, habilidades comunicativas y capacidad de reacción.

- *Factores relacionados con la organización*: apoyo de la alta dirección, estructuración del proyecto, coordinación de recursos comunes y apoyo de las unidades funcionales.

- *Factores relacionados con el entorno del proyecto*: clientes, subcontratistas y competidores y el entorno socioeconómico y tecnológico en general.

Los resultados de la investigación de campo en relación con los factores citados hablan de un significativo desplazamiento de los factores críticos hacia los dos primeros grupos como consecuencia de la progresiva importancia que la orientación a proyectos de las organizaciones va cobrando.

En el terreno específico de los proyectos de Sistemas de Información, en un estudio publicado por [WKR04] se analiza aplicando una correlación

logística la relación entre las desviaciones al alza en plazos y costes y una serie de instrumentos de gestión. La conclusión principal es que la variable que mejor distinguía entre proyectos desviados y no desviados era *la existencia y calidad de los procedimientos de seguimiento y control, incluyendo la anticipación de medidas correctivas*. Le seguían en relevancia las variables relativas a la existencia de métodos ajustados de estimación y el contar con buenas especificaciones y requisitos. El estudio concluye que la inversión en herramientas, técnicas y capacitación especialmente en las áreas citadas (obtención de requisitos, estimación, seguimiento y control), es la mejor defensa frente a la variabilidad de los proyectos.

Sin embargo, del mismo modo que se acepta que el éxito o el fracaso de un proyecto depende en último extremo de entregar en condiciones de calidad, coste y plazo se sigue constatando que las herramientas que teóricamente garantizan alcanzar esos objetivos y que constituyen uno de los núcleos de la investigación académica más visitados se emplean poco en la práctica [Her05].

2.2.3. La profesión y la academia : los fundamentos teóricos y prácticos de la disciplina

La gestión de proyectos es una disciplina relativamente joven evolucionada formalmente a partir de mediados del siglo XX con la aparición de la técnica PERT. El Project Management Institute creado en 1969 es la organización profesional más importante dentro de esta rama de actividad. El libro "*A Guide to the Project Management Body of Knowledge (PMBOK Guide)*" [Ins04] es el estándar principal en términos de práctica profesional.

Dada la naturaleza esencialmente práctica del trabajo de dirección de proyectos, generalmente acosado por las contingencias, la disciplina en su conjunto está orientada a la solución de problemas. Sin embargo, es posible que ese enfoque conduzca al tratamiento sintomático en lugar de acudir a las causas.

Por otro lado, la profesión y la formación y la investigación en gestión de proyectos se mueven en dos terrenos relativamente separados. De una parte la capacitación profesional en gestión de proyectos debe mucho a lecciones aprendidas, experiencia adquirida y apreciación de buenas prácticas. Se trata pues de una actividad muy empírica. De otra parte, la formación académica en gestión de proyectos descansa sobre todo en herramientas y aplicaciones, la mayoría de ellas provenientes del campo de la Investigación Operativa y

la Estadística. En [SD07] se afirma que sólo el 2 % de los casos de estudio de la Harvard Business School mencionan a los proyectos y apenas una docena están centrados en ellos.

Investigaciones recogidas en [Her05] indican que el grado de utilización de las herramientas basadas en los modelos de redes provenientes de la Investigación Operativa es muy elevado entre los directores de proyecto, pero que *su empleo se limita a fines relativamente poco sofisticados* en comparación con el potencial que dichos modelos teóricamente encierran. En términos prácticos, el uso principal es la planificación temporal y posterior seguimiento en el mejor de los casos, aunque a veces se limita a la representación y comunicación entre las partes interesadas. Los paquetes Microsoft Project y Primavera Project Planner, que son los más populares en la profesión, poseen funcionalidades para la gestión de las restricciones ligadas a los recursos y para el equilibrado del empleo de los mismos que raramente se emplean.

Paradójicamente el desarrollo de procedimientos para la programación de actividades bajo diversos tipos de restricciones de recursos, que es objeto es objeto de una intensa actividad de investigación académica, recibe poca atención por parte de los profesionales. La propia guía del PMI citada apenas destina un párrafo de 20 líneas al problema del equilibrado y secuenciación de recursos. De hecho, Goldratt [Gol97] llega a argumentar que los problemas de secuenciación y programación tienen escasa importancia debido a que "*el impacto en la duración de los proyectos es mínima*". Herroelen y Leus en el artículo arriba citado por su parte argumentan que este tipo de razonamiento es erróneo y conduce a subestimar los problemas ocasionados por los conflictos en el empleo de recursos.

La existencia de estas diferencias entre los practicantes y los investigadores es llamativa en un contexto en el cual los proyectos cobran cada vez más importancia como se ha indicado más arriba y sugiere la necesidad de una reflexión sobre los conceptos en los que se apoya la disciplina. En este sentido algunos investigadores ya citados [SD07] sostienen que el cuerpo teórico y conceptual de la gestión de proyectos es relativamente escaso. Afirman la necesidad de enriquecer la visión de la gestión de proyectos con paradigmas que enfoquen la investigación. Por paradigmas entienden literalmente una visión global sobre cuál es, o debe ser, el objeto de atención de la disciplina.

En esta línea, plantean tres posibilidades: la visión operacional centrada en el proceso, la visión organizacional centrada en el equipo y el liderazgo y la visión estratégica centrada en el negocio.

Visión operacional Se concentra en el proyecto como conjunto de actividades que sirven a una finalidad. Es la visión más tradicional, la que arranca con la aparición de la técnica PERT y se relaciona con una clase general de problemas ampliamente estudiados por la Investigación Operativa. El éxito se mide por la capacidad de ajustarse a especificaciones, plazos y costes. El papel del director del proyecto es asegurar ese ajuste.

Visión organizacional Entiende a los proyectos básicamente como equipos humanos que deben ser coordinados, dirigidos y motivados para contribuir a una finalidad común. La finalidad es el éxito como equipo: productividad, moral, aprendizaje, realización personal. El director del proyecto es responsable de construir las condiciones que permitan alcanzar esas finalidades.

Visión estratégica . Entiende que el proyecto es una actividad que contribuye a objetivos de mayor alcance de la empresa. El éxito del proyecto se mide por la creación de valor para la empresa del mismo pero también por su contribución a otras necesidades estratégicas relacionadas con los clientes y la evolución a largo plazo de la empresa.

Esta indicación sobre la necesidad de una visión más integrada de la dirección de proyectos surgida de la propia profesión no es la única posición crítica, ni siquiera la más radical. Por ejemplo [WS07] hablan de un *ethos* o espíritu de la gestión de proyectos que limita su desarrollo como disciplina. En el contexto de los proyectos de sistemas de información, proyectos software, que son el objeto de este trabajo, la hegemonía de una visión "ingenieril", de sistema "duro", frente a una visión más "conversacional" o "blanda", podría estar en el origen de muchos de los fracasos de los proyectos [Day00].

2.3. Caracterización del problema multiproyecto

2.3.1. La organización de la empresa y la gestión de proyectos

Una vez que una empresa tiene que acometer un proyecto debe decidir de qué manera se relaciona el mismo (entendido en el sentido organizacional)

con el conjunto de la compañía. Según [Mer06] hay básicamente tres maneras:

- *El proyecto como parte integrada dentro de una división funcional de la empresa.* Este caso sólo se da cuando esa división funcional es capaz de asumir íntegramente las actividades del proyecto (entendido ahora en el sentido operacional) con sus propios recursos.

- *La organización por proyectos.* Cuando el proyecto contiene todos los recursos necesarios para desarrollar las actividades. En este caso se obtiene la ventaja de que el proyecto es autónomo y no plantea conflictos con otros proyectos o con departamentos funcionales. Por contra, se corre el riesgo de duplicar esfuerzos en áreas funcionales e infrautilizar los recursos que podrían ser compartidos.

- *La organización matricial.* En este caso se produce una solución intermedia entre los dos casos extremos expuestos previamente. El proyecto posee una cierta autonomía pero emplea recursos compartidos. Este es el caso más general, si bien el rango de opciones es elevado. Este caso puede modelarse desde un punto de vista de procesos como un taller: el trabajo es realizado por una serie de estaciones/servidores que se corresponden con las unidades funcionales mientras que los proyectos son esos trabajos fluyendo entre estaciones [AMNS95].

2.3.2. Caracterización del entorno multiproyecto en términos de variabilidad y dependencia

En [Her05], los autores proponen un marco para establecer diferentes categorías de entornos multiproyecto en función de las características de los proyectos y su relación entre si. Señalan para ello dos características determinantes, la *variabilidad* y la *dependencia*.

Por variabilidad se entiende la incertidumbre que se deriva de la falta de información suficiente sobre las actividades requeridas al inicio del proyecto, incertidumbre que se irá desvelando conforme avanza el desarrollo, compuesta con la propia incertidumbre operacional en la ejecución de las propias actividades. La dependencia sería la medida en la que las actividades de un proyecto se ven afectadas por influencias externas al mismo. Estas influencias pueden deberse a actores externos (clientes, proveedores, ...) o a factores internos, típicamente los conflictos sobre recursos compartidos.

El marco resultante es el que recoge el cuadro 2.1. Se plantea en términos de cuatro casos extremos si bien cabe todas las combinaciones intermedias de manera continua. Esos cuatro casos son:

- Caso BB (*baja variabilidad, baja dependencia*). Se trata de una situación en la que los proyectos son rutinarios y están bien especificados los procedimientos por lo que la incertidumbre es baja. Un ejemplo típico serían trabajos de mantenimiento preventivo "in situ". Los recursos están asignados desde el nivel jerárquico superior al que gestiona el proyecto, normalmente en una organización orientada a proyectos que mantiene compartimentos relativamente estancos.

- Caso BA (*baja variabilidad, alta dependencia*). Se trata de una situación en la que recursos de propósito general se emplean por diversos proyectos por otra parte bien especificados y de baja incertidumbre pero que pueden colisionar en sus demandas. Un ejemplo típico sería la producción o el ensamblaje de conjuntos de poca complejidad bajo pedido. Típicamente se trata de una organización matricial en la que se busca equilibrar la presión de cada proyecto sobre las diferentes unidades funcionales.

- Caso AB (*alta variabilidad, baja dependencia*). Se trata de una situación en la que proyectos relativamente complejos y con incertidumbre derivadas de variables externas no controladas como las condiciones meteorológicas o ambientales de proyectos complejos de ingeniería civil o la volatilidad de las especificaciones. En una empresa orientada a proyecto, los recursos están dedicados a los proyectos individuales por lo que se producen pocos conflictos en torno a su asignación.

- Caso AA (*alta variabilidad, alta dependencia*). Se trata de una situación en la que se producen junto con las incertidumbres propias de cada proyecto las que se componen por la dependencia entre proyectos que compiten por recursos. Se trata de una situación característica de una organización matricial.

	Baja dependencia	*Alta dependencia*
Baja variabilidad	**BB**	**BA**
Alta variabilidad	**AB**	**AA**

Cuadro 2.1: Variabilidad y dependencia en el entorno multiproyecto según [HL04b]

2.3.3. Jerarquía de decisiones en la gestión multiproyecto

La clasificación anterior en términos de dependencia y variabilidad resulta incompleta sino se tiene en cuenta que una parte de las decisiones a tomar en la gestión multiproyecto precisamente conduce a acotar, en la medida de lo posible, esas contingencias. Así, la opción por una organización matricial conduce a una mayor dependencia mientras que la adopción de una organización por proyectos tiene como finalidad precisamente reducir aquella. Estas configuraciones no son "naturales"sino que son consecuencia de decisiones que se adoptan en los niveles estratégicos. La primera opción estratégica, precisamente, es la configuración que adopta una organización multiproyecto.

Adler et al. [AMNS95] en un artículo muy citado sugieren adoptar un punto de vista de procesos para tratar el problema de la gestión multiproyecto. Sostienen que la actitud que habitualmente prevalece está enfocada sobre los proyectos individuales en lugar de considerar la complejidad de asignar cargas de trabajo en un sistema con una capacidad dada. Se trataría, según ellos, de introducir un enfoque de *gestión por proyectos* que, partiendo de los objetivos y entregables concretos de cada proyecto individual, los integrara la operación global de la empresa. A pesar de que el enfoque de gestión por proyectos es muy citado, en [HHLW07] se sostiene que las aproximaciones reales son escasas.

La jerarquía de las decisiones

A juicio de Hans *et al* [HHLW07] el problema se debe en parte a la *confusión entre los diferentes niveles de responsabilidad y decisión*. En términos de la gestión de una cartera de proyectos es necesario diferenciar entre los criterios empleados para priorizar y seleccionar de proyectos, que corresponde a los niveles *estratégicos* de la dirección, y las decisiones *tácticas* y *operacionales* sobre la asignación de capacidad y el control de la programación de tareas.

Así en [LK02] se proponen métodos para la aceptación de proyectos en un modelo dinámico que considera la asignación de recursos en función de diversos criterios. En [AIG03] se proponen mecanismos de control de proyectos que limitan el número de proyectos activos con el objetivo de asegurar una carga de trabajo constante (CONWIP, *constant work in process*).

Según los mismos autores de [HHLW07], es preciso también distinguir el caso de los programas, como un caso particular de la gestión multiproyecto, en la medida en que los componentes de un programa concurren a un objetivo común y pueden verse como un *proyecto único complejo*, en lugar del caso más general de la coexistencia de proyectos independientes, cada uno de ellos con sus propios objetivos.

En el caso multiproyecto más general diversos proyectos compiten por los mismos recursos limitados. Para eludir los conflictos que se plantean en la práctica sobre la asignación de los mismos se puede recurrir a una planificación agregada con capacidad fija. Sin embargo esto hace perder flexibilidad al nivel operacional. Cuando los conflictos se plantean en este nivel, es corriente que los planes agregados se subviertan debido al diferente tamaño de los proyectos, la incertidumbre propia de cada uno de ellos, la naturaleza dinámica de la propia cartera de proyectos y el hecho de que cada proyecto puede tener directores diferentes, con criterios distintos y poder diferente en la organización.

Por estos motivos, sostienen, la gestión multiproyecto adecuada requiere ser considerada simultáneamente en los diferentes niveles de planificación teniendo presente que dichos niveles tienen diferentes objetivos, niveles de agregación, restricciones y flexibilidad. Por ejemplo, a nivel operacional es prioritaria la adecuada secuenciación de tareas para cumplir los compromisos de plazos. Sin embargo, en el nivel táctico lo es el empleo adecuado de los recursos sin tener momentos de capacidad ociosa frente a momentos de sobrecarga, el típico problema del equilibrado de recursos considerado a escala

de la empresa. En todos los niveles, disponer de planes robustos y estables es una necesidad. Así, la gestión multiproyecto debe ser capaz de tratar con estos objetivos de manera diferenciada y teniendo en cuenta la jerarquía de los mismos.

Planificación jerárquica

La planificación con un enfoque jerárquico ha sido muy estudiada en el ámbito de la planificación de la producción. Los sistemas MRP y todos sus sucesores se basan en la idea de la obtención de un *Plan Maestro de Producción*, correspondiente al nivel táctico de decisión, que determina los límites de la programación detallada al nivel de taller [LOL88].

Una primera extensión al terreno de la gestión de proyectos aparece en [HL89]. En ella, los autores proponen una planificación jerarquizada en dos niveles. En el primero se agregan las actividades de los proyectos que presentan perfiles de empleo de recursos semejantes y se asignan fechas de entrega. Dadas las fechas de entrega de los proyectos en curso y las fechas tentativamente asignadas a los nuevos proyectos, a través de un programa lineal se minimiza el coste descontado de los recursos no utilizados. Una vez se obtiene un perfil adecuado de empleo de recursos, las fechas asignadas se emplean para secuenciar cada proyecto individualmente.

Otras aproximaciones al problema pueden verse en [866] donde se minimizan los costes totales derivados de los retrasos. En [SV93] se agregan las actividades definidas a nivel operacional en programas a nivel táctico. Este enfoque es criticado por [HS96] que demuestra con contraejemplos que se alcanzan soluciones sub-óptimas. Yang y Sum [YS97, YS93] proponen una estructura dual para gestionar el uso de recursos en un entrono multiproyecto. Una autoridad central negocia los plazos con los clientes y asigna los recursos a los proyectos de modo que los proyectos críticos tienen prioridad, determinado así un calendario de inicio de proyectos. Los responsables operacionales programan las actividades de sus respectivos proyectos dentro de las restricciones así impuestas. Examinan el comportamiento de reglas heurísticas de asignación de recursos y fijación de plazos, incluyendo el caso en el que el desplazamiento de recursos entre uno y otro proyecto impone costes.

En [Boe98] se propone un marco jerárquico para la planificación en las organizaciones orientadas a proyecto. Argumenta que es necesaria una descomposición jerárquica para llegar a un proceso mejor gestionado. Menciona igualmente que la incertidumbre es un factor relevante en un entorno mul-

tiproyecto. Si esta es demasiado amplia, los canales de información en la estructura jerárquica se saturan. Para evitarlo propone cuatro estrategias: a) la creación de holguras relajando objetivos; b) la creación de tareas "auto contenidas", es decir, grandes agrupaciones de tareas a desempeñar por equipos multidisciplinarios; c) la creación de vínculos laterales empleando diseños organizativos matriciales o equipos especializados; y, d) la inversión en sistemas de información verticales. Asegura que estas vías son adecuadas para enfrentar los problemas de la incertidumbre en la organización orientada a proyectos. No obstante, y en la misma línea que la mayoría de los autores antes que él, acaba desembocando en un marco determinista de planificación en el cual no hay comunicación explícita para hacer frente a las variaciones entre los diversos niveles de la jerarquía.

En [NS98] se propone un modelo más sofisticado que se desarrolla en tres planos jerárquicamente organizados suponiendo una cartera de proyectos que deben acometerse dentro de un horizonte de planificación de 2 a 4 años. Cada proyecto posee una fecha de inicio, una fecha máxima de compleción y una descomposición en tareas y actividades (*work breakdown structure -WBS*). En el nivel más alto, a largo plazo, todos los proyectos se integran en una única red que los contiene como actividades agregadas. Las fechas de inicio y final se programan utilizando relaciones de precedencia generalizada (ver más adelante). Las actividades se programan restringidas por los recursos clave. La duración estimada de una de tales actividades agregadas equivale a la duración del camino crítico del proyecto correspondiente más un buffer que anticipa la duración calculada al tercer nivel de desagregación y que se estima utilizando métodos de la teoría de colas. La capacidad en términos de recursos clave viene dada por la estrategia general de la empresa. La función objetivo es la maximización del valor actual neto de la cartera de proyectos.

Esto proporciona una programación en la que se le asigna una duración máxima a cada proyecto y el perfil de empleo de recursos resultante que actúa como restricción de los proyectos individuales al segundo nivel de planificación. En éste segundo nivel, referido al medio plazo, cada proyecto se secuencia en términos de actividades desagregadas considerando los recursos no-clave como ilimitados. Aquí el objetivo es equilibrar el empleo de dichos recursos. Todo lo anterior proporciona el marco de restricciones para el tercer y último nivel en el que se minimiza la duración de los proyectos, ahora descompuestos en las tareas elementales.

En [Kol00] se propone un marco jerárquico para diferenciar entre los diferentes procesos de gestión en la fabricación bajo pedido, MTO- *manufac-*

turing to order. Se diferencia entre tres niveles: la aceptación de pedidos, la planificación de la producción y la secuenciación de operaciones. Un marco que se corresponde con los que ya se han comentado.

2.4. Conclusión: una propuesta de descomposición según los niveles de decisión del problema de planificación multiproyecto

De todo lo anterior se concluye la necesidad de establecer por una parte la distinción entre el tipo de decisiones a adoptar y su alcance, y de otra la comunicación y realimentación entre los diferentes niveles. Se pueden distinguir al menos tres problemas de planificación en relación con la capacidad:

- La decisión de admitir un nuevo proyecto, decisión que depende de la capacidad global de la empresa y del valor del nuevo proyecto para la misma. Se trata de una decisión guiada por consideraciones *estratégicas*.

- La decisión de fijar fechas de inicio y, eventualmente, compromisos de entrega para un proyecto ya aceptado - nótese que la decisión de aceptarlo puede conllevar la aceptación de un plazo de entrega - lo que conlleva decidir sobre la capacidad que se va a dedicar a su desarrollo y compleción, lo cual es una decisión de orden *táctico*.

- La decisión de programar las tareas concretas y asignarles los recursos concretos, que corresponde al ámbito *operacional*.

Puesto que las decisiones se toman en un entorno de incertidumbre en los tres niveles, es preciso prever los flujos de información entre los diferentes niveles con la finalidad de verificar el cumplimiento de los objetivos que en cada uno de ellos se ha señalado y arbitrar las medidas para hacer frente a las desviaciones que pueden comprometer dicho cumplimiento.

El nivel estratégico proporciona al nivel táctico las restricciones básicas del problema: decide la "admisión" de los proyectos al sistema. Los proyectos admitidos se caracterizan por su valor, lo que limita el coste total y, por tanto, la dedicación de recursos, y los hitos principales, normalmente negociados con, o requeridos por, el cliente. El nivel táctico a su vez deberá informar al

nivel estratégico sobre la evolución de la cartera y sus componentes, normalmente a través de una metodología de síntesis como por ejemplo la del *valor alcanzado - EVM* [DF00].

Las organizaciones que desarrollan proyectos en entornos de alta dependencia (BA y AA del punto anterior) suelen corresponder a una organización matricial. Para este tipo de organizaciones [HHLW07] recomiendan un flujo de información entre los niveles táctico y operacional del siguiente tenor: al inicio del proyecto, cuando sólo se dispone de información limitada sobre el contenido de trabajo de un proyecto, la salida más importante que proporciona la planificación de capacidad es la fijación de hitos temporales internos y externos y la capacidad requerida. Esta información permite adquirir los recursos necesarios y cerrar fechas. Conforme el proyecto avanza la información gradualmente es más completa. Con esta información junto con la del diseño y planificación de los procesos puede pasarse a la fase de planificación operacional. Esta consiste básicamente en resolver un problema de programación multiproyecto con restricción de recursos.

En las organizaciones orientadas a proyecto, el marco es del tipo BB y BA. En tal caso, los autores recomiendan una interacción diferente. Además de la información citada, la planificación táctica permite asignar recursos a cada proyecto por lo que estos se pueden gestionar a nivel operacional como proyectos individuales relativamente autónomos, es decir como un problema de programación de proyectos con restricción de recursos.

Una organización dedicada al desarrollo de *software* a medida es típicamente una organización por proyectos, por tanto *orientada a reducir la dependencia entre proyectos a nivel operacional asumiéndola en el nivel táctico.* Es decir, en la medida de lo posible se intenta independizar en términos de recursos cada proyecto individual. Esta independencia será siempre relativa, en función del recurso del que se trate. Habrá por tanto recursos que se pueden compatibilizar entre proyectos y recursos exclusivos de cada uno de ellos, por tanto preservaremos inicialmente los dos enfoques: multiproyecto y proyectos sigulares.

La decisión de aceptar proyectos queda fuera del ámbito del problema en estudio. Se deriva de criterios estratégicos y de la acción comercial, si bien la realimentación entre el proceso comercial y el productivo no debe olvidarse [CB07]. A los efectos que nos interesan, los proyectos "aparecen" previamente "marcados" con fechas límite que pueden deberse a condiciones internas o externas.

Los objetivos de nuestro estudio entonces se plantean como:

- En el nivel *táctico*; asignar a los proyectos los recursos necesarios para asegurar el cumplimiento de plazos de entrega. Estos recursos deben obtenerse empleando la capacidad productiva de la empresa de la forma más económica posible y dentro de los requisitos de calidad de los proyectos. Se trata de un problema de **planificación táctica de capacidad** que debe considerar, aparte de las restricciones temporales, las asignaciones previas y el coste de los recursos.

- En el nivel *operacional*; programar los proyectos dentro de los márgenes impuestos desde el nivel táctico de forma que se pueda hacer frente a la variabilidad propia de cada uno de ellos. Se trata de un problema de **programación de proyectos con restricción de recursos** en entorno de incertidumbre y con ventanas de tiempo. Según el grado de dependencia mutua estaremos hablando de proyectos **individuales** o **multiproyecto**.

Decisión	Nivel	Información
Admisión de proyectos	Estratégico	Valor e hitos externos
Planificación de capacidad	Táctico	Recursos e hitos internos
Programación con restricción de recursos	Operacional	WBS, gestión del riesgo, ...

Cuadro 2.2: Descomposición del problema multiproyecto por niveles jerárquicos

Capítulo 3

El problema de la planificación táctica de capacidad

3.1. Objetivo y contenidos

En el capítulo anterior se ha planteado una perspectiva para abordar el problema de la gestión multiproyecto basada en la descomposición en dos niveles: uno táctico y el otro operacional. En este capítulo se revisa el primer nivel, el de la *planificación táctica de capacidad*.

Como más adelante se indica, éste es un problema relativamente poco tratado, especialmente en comparación con el problema operacional que se discute ampliamente en el siguiente capítulo. La principal característica del problema de planificación táctica de capacidad es que se basa en una estimación agregada, y sujeta a incertidumbre, de la carga de trabajo que implican los proyectos. En la literatura se denomina *rough cut capacitiy planning*, puesto que se trata de una planificación basada en una estimación *a grandes rasgos*. Lo que se busca es *reservar* los recursos que requerirá un proyecto para poder planificarlo detalladamente.

Tras una discusión sobre los antecedentes del problema, se pasa a caracterizarlo en base a sus diferentes facetas.

En primer lugar, y adoptando el criterio de que la previsión de carga de trabajo es determinista, se repasan las soluciones existentes en la literatura al problema es dos variantes: cuando la restricción "dura" son los plazos y, por tanto, la variable de decisión es la cantidad de recursos que se adquieren, y cuando el factor más limitante son los recursos y, por tanto, son los plazos lo

que no puede alterarse. EL problema se ha planteado en las referencias que se han identificado como un problema de programación matemática, resoluble bien por métodos exactos, bien por métodos aproximados.

Lo más característico del problema, sin embargo, es que se aborda cuando el conocimiento que se tiene de cuáles van a ser las cargas de trabajo reales es muy aproximado. En el caso de los proyectos *software*, se basa en una estimación que puede ser analítica (COCOMO, puntos-función, ...)o bien estar basada en la experiencia o por analogía. Por eso se debe tratar como un problema con incertidumbre. Las aproximaciones a esta cuestión que se han identificado son:

- a través de una técnica de *escenarios* que permiten acotar, dentro de ciertos límites, la variación posible de la realidad respecto de las estimaciones centrales
- *relajando* las restricciones para dar cabida a un margen de variación en plazos y/o recursos
- tratando el problema como un problema esencialmente *dinámico* en el que las previsiones se ven alteradas en tiempo de ejecución por los nuevos datos que la misma va ofreciendo y por la aparición de nuevos proyectos

EL capítulo termina proponiendo precisamente este tercer punto de vista por considerarse que es el que mejor refleja la realidad que se pretende modelar.

3.2. Aproximaciones al problema de planificación táctica de capacidad

El problema de planificación táctica de capacidad es un problema relativamente poco estudiado. En el plano estratégico existe una literatura muy abundante, en general en relación con el extenso campo de las cadenas de suministro, la producción flexible, las redes de transporte, etc. La planificación de capacidad en el plano estratégico se basa en estimaciones agregadas de la demanda sin tener en cuenta la especificidad de los pedidos de los clientes.

En el plano operacional, la secuenciación on-line ha sido ampliamente estudiada como una aplicación del problema RCPSP estocástico [MSSU03],

[Uet01], [MS00] en el mismo contexto de las cadenas de suministro, la secuenciación de tareas en talleres *job-shop* y *flow-shop* y también en aplicaciones relacionadas con la gestión de redes y recursos informáticos (procesadores). En este plano, el principal factor de incertidumbre es el ritmo de llegada de las tareas o proyectos.

El problema de la etapa de planificación táctica que aquí se investiga no es que el ritmo de aparición sea una fuente de disrupción permanente de la programación puesto que en el nivel estratégico previo existe un filtro que "admite" o no nuevos proyectos. El problema es que se dispone sólo de una aproximación muy general al contenido de trabajo de los proyectos que van llegando.

3.2.1. EL problema táctico como un problema de programación dinámica

Una aproximación al problema táctico se recoge en [AT08], en el que se emplea un modelo de programación dinámica para un horizonte finito de intervalos discretos de tiempo y se busca minimizar los costes totales de hacer frente a una demanda estocástica mediante una política de recursos permanentes y recursos contingentes (personal eventual y horas extraordinarias). Se trata de producir para inventario, por lo que el modelo se adapta poco al caso de estudio. También en un contexto de producción para inventario se pueden reseñar [NS98] y [CLT02].

3.2.2. El problema táctico como un problema de asignación

Otro conjunto de aproximaciones proviene del campo del problema generalizado de asignación (GAP) en el que se trata de determinar una asignación óptima (generalmente de coste mínimo) de una serie de tareas a agentes. Se trata de un problema derivado de un problema tradicional de taller, la asignación de m trabajos a n máquinas, con restricciones de todo tipo: limitación de recursos, ventanas de tiempo, etc. Un problema parecido se trata a la hora de confeccionar horarios, asignar personal a flotas de transporte o programar intervenciones quirúrgicas en un hospital [BJN$^+$98, Nag02, GC03, CR06, LTN06, SW06, TGR05, FCMA08].

En un contexto de proyectos las principales contribuciones detectadas en

esta línea corresponden a [HGdVZ02b, GS05, HGdVZ02a, Boe98] y [BS99]. Se trata de modelos de programación lineal entera en los que se minimiza el coste de la capacidad no regular empleada cumpliendo con una serie de restricciones temporales. El horizonte de planificación es finito y definido en términos discretos.

3.3. El problema de planificación táctica de la capacidad como un problema de asignación

El punto de partida es un horizonte temporal finito, T, dividido en intervalos - *buckets* - regulares de duración normalizada a la unidad (por ejemplo, una semana). La empresa tiene una cartera de proyectos definidos en términos de actividades agregadas, con una duración estimada cada una de ellas. La empresa emplea recursos diferentes , disponiendo de unidades del recurso en el intervalo de tiempo t. Cada actividad agregada i requiere un empleo del recurso R_j que viene dado por q_{ij}.

Las actividades agregadas pueden tener relaciones de precedencia entre si. De hecho, cada uno de los proyectos no es más que la subred resultante de esas relaciones de precedencia. Las actividades agregadas además se realizan en modo continuo, es decir, la duración de las mismas en términos de intervalos de tiempo depende de la cantidad de recursos de los que disponga. Debe disponer de todos los recursos que requiere para poder completarse. En general una actividad agregada dura varios periodos de tiempo. A lo largo de cada periodo se completa una fracción del contenido en trabajo total de una actividad agregada. De un periodo a otro esa fracción puede cambiar.

Sea x_{jkt} la fracción de la actividad agregada j que se realiza en el periodo t empleando el recurso k. Suponemos que esa fracción es igual para todos los recursos lo que equivale a decir que la proporción de recursos requeridos de cada tipo es constante en una actividad, o lo que es lo mismo, cada actividad agregada implica una combinación fija de recursos. Esta visión de la actividad a un nivel agregado lo permite. De no ser así se puede descomponer una actividad agregada en varias. Eso nos permite referirnos en lo sucesivo a x_{jt} como la fracción de la actividad j que se realiza en el periodo t. La intensidad del trabajo puede ser variable de un periodo a otro representándose como un cambio en esa fracción.

Supongamos además que la máxima fracción de trabajo que puede hacerse de una actividad en un intervalo de tiempo es j es $\frac{1}{p_j}$. Eso equivale a decir que la duración mínima de cada actividad es p_j periodos. Una tarea está completa en el intervalo $[t_1, t_2]$ si la suma de todas las x_{jt} en ese intervalo es igual a la unidad.

Cada tarea tiene asignada una ventana de tiempo $[r_j, d_j]$. No puede empezarse antes de r_j ni terminarse después de d_j. Por tanto x_{jt} debe ser cero para $t < r_j$ y $t > d_j$. Además $d_j - r_j$ tiene que ser mayor o igual que p_j, el tiempo mínimo en que se pude hacer la tarea.
Las relaciones de precedencia las representaremos como la exigencia de que si la actividad i precede a la actividad j, en el momento τ para que $x_{j\tau}$ sea mayor que 0 es necesario que la suma de x_{it} desde cero hasta τ sea igual a 1.

Una solución está formada por una lista ordenada de $n \times T$ componentes, x_{jt}, con t de 1 a T y j de 1 a n que indica la fracción de la actividad j que se ejecuta en el periodo t. Para ser factible requiere:

- Respetar las relaciones de precedencia y de tiempo
- No ser mayor que $\frac{1}{p_j}$
- Que $\sum_{t=1}^{T} x_{jt} = 1$

Por último existe la posibilidad de emplear recursos extraordinarios o no regulares en cantidad U_{kt} para el recurso k en el intervalo t a un coste c_{kt}. Suponemos inicialmente que el empleo de recursos no regulares (horas extraordinarias, empleados eventuales, ...) no está limitado.

La función objetivo a considerar puede adoptar dos formas:

- *Problema restringido por el tiempo:* se trata de cumplir con un horizonte temporal para cada proyecto al menor coste de oportunidad que es el derivado del empleo de recursos no regulares.
- *Problema restringido por los recursos:* se trata de minimizar la duración de los proyectos sin emplear recursos irregulares.

3.3.1. El problema restringido por el tiempo

El problema restringido por el tiempo, sin tener en cuenta las relaciones de precedencia, puede representarse como:

$$min \sum_{t=1}^{T} \sum_{k=1}^{K} c_{kt} U_{kt} \tag{3.1}$$

Sujeto a:

$$\sum_{t=1}^{T} x_{jt} = 1 \tag{3.2}$$

$$x_{jt} \leq \frac{1}{p_j} \tag{3.3}$$

$$U_{kt} \leq \sum_{j=1}^{n} q_{jk} x_{jt} - Q_{kt} \tag{3.4}$$

$$x_{jt}, U_{kt} \geq 0 \tag{3.5}$$

En el que se busca minimizar el coste de los recursos irregulares asegurando que se completan los proyectos respetando el límite a la duración mínima de las actividades pero sin tener en cuenta las relaciones de precedencia; es decir es una variante relajada del problema.

Para introducir las relaciones de precedencia, se define una ventana de tiempo VT_j para una actividad j como un intervalo $[S_j, C_j]$ tal que la actividad no puede comenzar antes de S_j ni concluir después de C_j. Un conjunto de ventanas de tiempo CVT es una lista de n elementos donde cada actividad tiene asignada una ventana de tiempo. Un conjunto de ventanas de tiempo *factible* es un conjunto de ventanas de tiempo que cumple las relaciones de precedencia y la restricción 3.3. Existe un problema 3.1 para cada CVT factible.

Si ahora sustituimos x_{jt} en la restricción 3.3 por $\frac{s_{jt}}{p_j}$ donde s_{jt} vale 1 si la actividad j se está ejecutando en el tiempo t y 0 de lo contrario, cada problema 3.1 nos aparece como un problema de programación binaria que se puede descomponer en dos grupos de restricciones:

- *Restricciones que afectan a todos los proyectos*, las relacionadas con los recursos

- *Restricciones de cada proyecto*, las relacionadas con las relaciones de precedencia

Esto permite descomponer la matriz de las restricciones en dos bloques, uno primero que recoge las restricciones relativas a recursos y otro segundo relativo al conjunto CVT factible. Una solución cualquiera del problema (continuo) general es una cota inferior del problema restringido (mixto continuo-entero) en el tiempo. A su vez, una solución para un problema restringido para un CVT factible es una solución factible del problema general y puede servir, por tanto, de punto de partida para un heurístico de mejora. A partir de aquí los autores consultados proponen soluciones heurísticas que permiten obtener resultados de calidad en tiempos de computación razonables.

3.3.2. El problema restringido por los recursos: *forward resource loading.*

La variante restringida por los recursos se denomina problema de carga de recursos hacia adelante, *forward resource loading* (FRL). Aquí la función objetivo debe incorporar un término que penalice los retrasos. Si no es posible emplear recursos extraordinarios, basta con hacer 0 las variables U_{kt} que representan su consumo. Sin embargo la formulación se complica ya que para evaluar (y penalizar) los retrasos, se hace necesario determinar la fecha de finalización de cada proyecto y ello no puede hacerse sin incluir en la función objetivo alguna medida relacionada no ya con los tiempos de ejecución de cada actividad sino con la duración de los proyectos en su conjunto, lo que obliga a introducir explícitamente nuevas variables. Nótese que en el caso anterior esto no era necesario en la formulación del problema básico, bastaba con resolverlo en el sub-espacio de los CVT admisibles.

Una forma de simplificar el problema es suponer que la estructura de actividades agregadas de un proyecto particular es simplemente una cadena, es decir, una secuencia lineal. Esta simplificación al nivel que estamos considerando de planificación táctica de capacidad no supone una limitación grave al realismo de los modelos. En un proyecto de desarrollo *software*, salvo que se adopte un modelo de ciclo de vida evolutivo, las grandes fases de Análisis, Diseño, Construcción y Pruebas pueden descomponerse en *cascada* sin perjuicio de que se adopte un modelo de desarrollo incremental o de otro tipo a un nivel más fino de análisis.

Supónganse n proyectos formados por actividades agregadas *(b,j)* donde

(b,j) es la actividad b-ésima del proyecto j. Los recursos vienen definidos como en el caso anterior como $R_1, R_2, \ldots, R_K$ disponiéndose de Q_{kt} unidades del recurso R_k en el intervalo de tiempo t. Cada proyecto debe empezar en la fecha r_j y concluir antes de la fecha d_j. Cada actividad agregada *(b,j)* requiere un empleo del recurso R_k que viene dado por q_{bjk}. Como en el caso anterior consideramos un empleo de recursos en proporciones constantes, de manera que p_{bj} es la fracción de la actividad *(b,j)* que se realiza en el periodo t.

Llamaremos un plan π para un proyecto j a un vector a^{π}_{bjt} con elementos binarios que contiene un 1 si la actividad *(b,j)* se puede realizar en el periodo t y un 0 de lo contrario. Sólo consideramos planes factibles, es decir, planes en los que se cumplen las relaciones de precedencia entre actividades, las fechas finales y la duración mínima de las actividades. La generación de planes factibles es un problema relativamente sencillo puesto que hasta este punto los proyectos son independientes unos de otros. Un plan indica cuando se permite iniciar una actividad.

Llamaremos una *secuencia de carga* a un vector Y_{bjt} cuyos elementos son la fracción de la actividad b del proyecto j que se realiza en el tiempo t. Multiplicando cada elemento de ese vector por el correspondiente p_{bj} sabemos el consumo de recursos de cada actividad en cada momento.

El tiempo de compleción de un proyecto j, CT_j, es el último intervalo de valor 1 de un plan π para ese proyecto j.

$$CT_j = \text{máx}_{\pi factibles} t \cdot a^{\pi}_{bjt} \tag{3.6}$$

La tardanza de un proyecto δ es la diferencia entre la fecha señalada de finalización de un proyecto, d_j, y su tiempo de compleción, CT_j. El retraso de un proyecto vale 0 si la tardanza es negativa y $CT_j - d_j$ si la tardanza es positiva. Esta variable es la que se incluye en la función objetivo. Nótese que como la tardanza negativa no se considera, se puede llegar a una solución de coste mínimo que puede mejorarse en términos de duración de otras actividades y proyectos.

La solución de este problema puede abordarse de diversas maneras a partir de la obtención de una primera solución admisible. Esto se consigue empleando un heurístico tradicional del problema RCPSP (ver 4.4 y 4.5) o bien la fase I del modelo tradicional Simplex en dos fases para generar una solución inicial o demostrar la inexistencia de las mismas [Han01]. A partir de esta solución se pueden seguir métodos exactos o métodos heurísticos.

Solución exacta

Se trata de un algoritmo denominado *Branch-and-Price*: [BJN+98], [Han01], [SS03][SW06] y [BD07]. Este método, utilizado para resolver problemas de asignación, combina el método de exploración dirigida a través de un árbol de soluciones con la inclusión progresiva de columnas, que corresponden a planes factibles, en función de un criterio derivado del precio sombra, es decir, evaluado a través de los costes reducidos.

Solución aproximada

El método exacto, a pesar de explotar la posibilidad de descomponer la matriz del problema agregando columnas progresivamente, cada una de las cuales es un plan factible, proporciona soluciones que no son factibles, al resultar planes que son combinaciones convexas de planes factibles (es decir, el proyecto debe hacerse en un 20 % de acuerdo con un plan y en un 80 % de acuerdo con otro, lo cual obviamente es imposible). Por ello [GS05] proponen un método heurístico de mejora que sólo evalúa soluciones factibles. Para ello se parte de un único plan factible (una columna) y se miran los precios sombra para ver que modificaciones deben hacerse a los proyectos, bien adelantando o demorando las fechas de inicio y/o finalización. Tras eliminar las modificaciones inviables eligen aquella que más mejora la solución. El heurístico prosigue resolviendo problemas restringidos de una sola columna por plan hasta que no se puede mejorar.

3.4. Planificación táctica de capacidad e incertidumbre

3.4.1. El uso de escenarios

Si bien el problema del riesgo y la incertidumbre se tratan con más detalle más adelante en el capítulo 5, en este apartado se describe una aproximación al problema del riesgo en la planificación táctica de capacidad.

En este nivel de decisión, la fuente de incertidumbre más relevante es la falta de información sobre el contenido de trabajo real de cada actividad agregada. Si bien la simplificación de que cada actividad agregada se desenvuelve con una productividad constante y que la proporción de recursos también se

mantiene constante es admisible a este nivel del análisis (no así en la programación operacional), no es evidente que el contenido real de trabajo se conozca con suficiente precisión. Eso es así porque los compromisos de fechas contraídos y el valor del proyecto se han fijado teniendo en cuenta una estimación grosera del volumen de trabajo implicado, sin que el contenido real se conozca a priori sino que más bien irá desvelándose conforme el proyecto avance.

En el mismo trabajo antes descrito [GS05] plantean una aproximación a este problema basada en escenarios. Un escenario es un supuesto razonable del contenido de trabajo de las actividades agregadas que conforman los proyectos que se están planificando. Para cada actividad, en analogía con el PERT clásico, se definen tres hipótesis en términos de contenido de trabajo: optimista, promedio y pesimista. Esto incrementa la dimensión del problema extraordinariamente, pues genera escenarios para n actividades con incertidumbre.

Para reducir el espacio de búsqueda proponen una estrategia basada en obtener muestras de los escenarios, bien de manera aleatoria bien mediante criterios preestablecidos

En este segundo caso elaboran escenarios pesimistas, promedio y optimistas para distintas instancias del problema variando el número de actividades, de recursos, la holgura promedio y el grado de utilización de los recursos (ver más adelante). Tras un extensivo estudio combinando escenarios y aplicando los dos modelos de solución indicados (truncando el método exacto cuando las iteraciones no han concluido al cabo de 10 minutos) concluyen que con tres escenarios y números razonables de recursos (entre 3 y 10) y de actividades (entre 10 y 40), el heurístico basado en soluciones factibles y precios-sombra proporciona buenos resultados.

3.4.2. La incertidumbre acotada como restricción: planificación robusta

Una solución robusta es una solución que admite la variación en los datos sin incurrir en la necesidad de reprogramar las actividades. Si, como en el caso anterior, se puede acotar el intervalo de variación del contenido de trabajo, se pueden establecer cotas inferiores y superiores a la duración de las actividades. En [BTN00] se muestra un procedimiento para obtener soluciones robustas, entendidas como soluciones capaces de absorber la variación de

las restricciones dentro de ciertos límites. Este método se ha descrito para su empleo en diferentes dominios como puede verse en [LJF04], [KL05], [AP06] y [PS06].

En el caso del problema que nos ocupa, orientado a la minimización de retrasos sujeto a la capacidad de recursos, este método puede aplicarse a una o ambas series de restricciones, las de recursos y las de plazos. En el primer caso, en lugar de autorizar el empleo de recursos extraordinarios se puede dar una tolerancia al exceso de consumo de recursos. Es un modelo adecuado para una situación en la que hay que acudir a horas extraordinarias, las cuales sólo deben representar un porcentaje muy limitado de las horas regulares so pena de afectar a la productividad. En el caso del tiempo, se pueden relajar bien los plazos de entrega, bien la superposición de actividades agregadas en cadena.

El método parte definiendo una tolerancia δ (que puede ser homogénea para todas las restricciones o singular para cada una de ellas o cada tipo) como el porcentaje máximo de desviación admisible. en el cumplimiento de las restricciones. Esa tolerancia δ y la desviación unitaria ϵ de cada coeficiente de la matriz de restricciones o de los términos independientes de cada una de ellas. Nótese que también es posible establecer diferentes valores de ϵ, uno por coeficiente o término independiente incierto.

Los autores citados demuestran que se puede construir un programa lineal al que denominan *contrapartida robusta en el intervalo* δ, ϵ del problema original, que es determinista. Este problema se construye incorporando las restricciones derivadas del cumplimiento de las cotas de los coeficientes y los términos independientes y las derivadas de la tolerancia de admisibilidad. Si el problema presenta características de simetría en la variabilidad de los datos, se puede aún reducir más a un problema más simple de programación no-lineal.

3.4.3. Formulando explícitamente la incertidumbre: el problema de planificación táctica de capacidad como un problema dinámico y estocástico

En [CRL04] se propone una aproximación alternativa a la solución del problema de planificación de capacidad enfocándolo directamente como un problema estocástico y dinámico. El modelo está basado en un problema de desarrollo de producto en la industria farmacéutica en el que las diversas acti-

vidades que conducen al desarrollo de un producto están sujetas a resultados inciertos.

La empresa posee una cartera de productos en desarrollo. Estos productos en desarrollo compiten por unos recursos limitados. Su valor para la empresa es una función decreciente con el tiempo. El desarrollo de un producto es una secuencia de actividades de duración incierta y que pueden condicionar el desarrollo de las actividades posteriores. El objetivo es maximizar el valor para la empresa de la cartera de productos de forma dinámica.

Para ello se modela en tiempo discreto cada proyecto como un proceso markoviano. El estado del proyecto en un momento del tiempo viene dado por las actividades completadas, en curso y por iniciar; los recursos ocupados los determinan las actividades y el propio tiempo. La conclusión y el inicio de una actividad dan lugar a la transición de estados. Las transiciones pueden darse de acuerdo con determinadas probabilidades conforme al modelo convencional de cadenas de Markov.

Un algoritmo de programación dinámica que maximice el valor para la empresa resulta prohibitivo dado el enorme número de combinaciones que admite el problema. Existen en la literatura diversas técnicas para reducir el problema basadas en propiedades de la función objetivo (monotonicidad, separabilidad) del espacio de estados (dominancia estocástica), ...

En el artículo de referencia se propone como alternativa el empleo de reglas de prioridad de los heurísticos convencionales del problema de programación de proyectos con recursos limitados que se tratan en el capítulo siguiente para obtener un espacio de estados más reducido que el original.

Atribuyendo valores prohibitivos a la función objetivo en los estados que no han sido generados por los heurísticos se confina el espacio de búsqueda para el algoritmo de programación dinámica. De esta forma el algoritmo converge con más rapidez a una solución satisfactoria. Esta aproximación tiene la ventaja de que se pueden simular las políticas heurísticas para completar la cartera de proyectos a partir de cualquier estado previamente alcanzado. De este modo se presta a servir de base para modelos de apoyo a la decisión en tiempo de ejecución.

3.5. Conclusiones y propuesta

Dada la naturaleza táctica del problema, el problema que se le plantea a una organización que desarrolla a la vez varios proyectos *software* es el de

dimensionar los equipos necesarios para abordar esos proyectos de manera que se asegure que se cumplen los plazos. En ese sentido el problema se plantea en los términos en que se expone en el apartado 3.3.1.

La empresa cuenta con un "pool" de recursos ordinarios y cabe la posibilidad de añadir más. Puesto que los recursos ordinarios son el personal de plantilla, el coste relevante es, como en el modelo que se ha indicado, el de los recursos adicionales. La principal variable de decisión es precisamente el número de personas que habrá que contratar por encima de los recursos ordinarios para cumplir los compromisos de tiempo.

La representación de las actividades agregadas de manera que la proporción entre los distintos recursos que emplea cada una de ellas es también adecuada al problema puesto que en la etapa de planificación táctica no cabe una desagregación tan fina de las tareas que permita singularizar los recursos individuales tanto en función de su especialización. El modelo admite que los recursos sean compartidos por varios proyectos, algo que a este nivel de análisis se corresponde con la realidad del problema en estudio.

La formulación de las ventanas de tiempo admisibles es relativamente sencilla si, como suele ser el caso en los proyectos de *software* a medida, existen hitos externos caracterizados por entregables que deben hacerse llegar al cliente. Luego las restricciones temporales resultan fáciles de implementar.

Los problemas que presenta el modelo para el caso de estudio son:

- El modelo asume una productividad lineal, es decir, que el tiempo de desarrollo es inversamente proporcional al número de personas empleadas en las tareas, algo que choca con la asunción generalizada implícita en la ley de Brooks. En una primera aproximación, una transformación del tamaño de los equipos a una variable relacionada linealmente con la productividad (del tipo equipos "equivalentes") puede resolver el problema a costa de la apareción de una función objetivo no lineal con la primera derivada creciente en función del tamaño "equivalente". En todo caso un estudio más detallado puede servir para ver cuanto se distorsiona asumiendo que la productividad es lineal con el número de recursos efectivos.

- El modelo es estático, en el sentido de que la aparición de un nuevo proyecto obliga a una reprogramación. Sin embargo, el ritmo de aparición de nuevos proyectos hace que no sea computacionalmente excesivamente costosa la reprogramación.

- De lo visto en la literatura examinada, el tratamiento más adecuado al riesgo parece el que se hace en la modalidad indicada en el apartado 3.4.2 puesto que se pueden relajar las restricciones

Por tanto, la propuesta es emplear el modelo presentado en el apartado 3.3.1. Admitiendo la linealidad se trata de:

1. Obtener una cota inferior del coste de la planificación téctica de capacidad a partir de la solución del problema continuo; será una asignación de recursos en general no compatible con las relaciones de precedencia y los plazos

2. Generar una solución factible del problema restringido a partir de alguno de los algoritmos de generación de secuencias que se describen en el capítulo siguiente

3. Utilizando el coste reducido, iterar sustituyendo esa solución factible por otra igualmente factible que tenga el menor coste reducido hasta que la aproximación a la cota inferior sea suficiente

4. La inclusión de un nuevo proyecto generará una nueva solución factible simplemente añadiendo recursos extraordinarios para cumplir los plazos. Este es el punto de partida para una nueva exploración ahora con el nuevo proyecto.

Capítulo 4

Gestión operacional de proyectos con restricción de recursos

4.1. Objetivo y contenidos

Al contrario que el problema táctico revisado en el capítulo anterior, el problema operacional es objeto de mucha investigación y experimentación, si bien más en el terreno académico que en el profesional, donde se siguen aplicando mayoritariamente métodos menos sofisticados que los que presenta la literatura.

El objetivo de este capítulo es presentar al menos una visión de conjunto de los tipos de soluciones que se proponen para el problema de la programación operacional de proyectos en un contexto de restricción de recursos. Necesariamente esta visión de conjunto apenas puede señalar los modelos principales y hacer referencia a las técnicas de solución propuestas más recientemente. La cantidad de investigación y conocimientos acumulados sobre esta materia excede con mucho el alcance de este trabajo. Algunos problemas, como los de la topología de las redes o la intensidad del empleo de los recursos son apenas mencionados, a pesar de que por si mismos son el objeto de mucho trabajo y dedicación. Pero lo que aquí se persigue es identificar aquellos modelos y técnicas que pueden ser útiles para implementar en un modelo de

simulación, lo que descarta muchos aspectos dificilmente integrables en esta perspectiva.

En el capitulo se presentan, por orden de complejidad creciente, los modelos de gestión de *redes de proyectos*, introduciendo progresivamente los siguientes factores:

- El problema con *restricción de recursos*
- El problema *multimodal*, donde existen diferentes modos de realizar las actividades que suponen diferentes consumos de recursos y duraciones
- EL problema de las *precedencias generalizadas* que permite modelar la existencia de ventanas de tiempo permitidas y, por tanto, de plazos de entrega
- El problema*multiproyecto*

Una vez presentados los modelos, se plantean los diferentes modos de solución que se exponen en la literatura.

En primer lugar, los métodos *exactos*, que corresponden a modelos de programación lineal mixta y que se resuelven de manera más eficiente a través de algoritmos de *enumeración implícita*, del tipo *btanch-and-bound*. En segundo lugar los métodos *aproximados*. El problema en su formulacióm más simple es un problema *NP-duro*. Conforme se va complicando, incluso el encontrar soluciones admisibles es un problema NP-duro. Por eso se aplican técnicas heurísticas y metaheurísticas de muchos tipos para abordarlo.

Dentro de los métodos aproximados, se describen primero los algoritmos de *generación de secuencias* que sirven para construir soluciones viables al problema. Posteriormente se describen las formas de representación de esas soluciones para que sea posible su tratamiento algorítmico. A partir de esto se describen las *reglas de prioridad* que son el punto de partida de los procedimientos heurísticos convencionales. Estos se basan, bien en procedimientos de búsqueda local, bien en la aplicación de sucesivas pasadas de los esquemas de generación de secuencias, o de combinaciones más o menos sofisticadas de ambos.

A continuación se describen brevemente las metaheurísticas más utilizadas, para concluir con una referencia al rendimiento computacional

de las diferentes técnicas. El capítulo concluye con una propuesta de heurísticos a retener para su posible implementación en el problema de referencia.

4.2. El modelo básico

El modelo clásico de un proyecto [Tav02] es un modelo relativamente simple de Investigación Operativa, un *grafo dirigido y acíclico* definido por:

- Un conjunto finito de *actividades* o tareas que hay que programar o secuenciar a fin de completar el proyecto
- Unas *relaciones de precedencia* entre dichas actividades representadas unívocamente por el conjunto de las inmediatamente anteriores a cada una de ellas. A esta relación se le denomina Final-Inicio aunque en ocasiones esas relaciones pueden ser *generalizadas*, es decir, no necesariamente vinculan a la actividad precedida con la precedente por la fecha de terminación de esta que debe ser menor o igual a la de iniciación de aquella, sino que pueden ser de diversos tipos como:
  - *Inicio-Inicio*; cuando el inicio de la actividad sucesora requiere que se ha iniciado la predecesora
  - *Final-Final*: cuando la predecesora debe terminar en tiempo menor o igual que la sucesora
  - *De realización parcial*: cuando la sucesora sólo pude iniciar una vez que la predecesora se ha completado en un porcentaje arbitrario
  - *Con retrasos*: cuando debe satisfacerse un retraso arbitrario entre los inicios y finales en cualquiera de sus combinaciones
- Un conjunto finito y discreto de *atributos de cada actividad*, representando propiedades relevantes del proyecto a efectos de su gestión, tales como duración, coste, consumo de recursos, ...
- Un conjunto discreto y finito de *criterios* que expresan las preferencias del (de los) decisor (decisores) tales como la duración total, el coste total, la relación coste-beneficio, el valor actual, ...

Este modelo básico se ha refinado y enriquecido progresivamente con las aportaciones de la comunidad académica, especialmente del campo de la Investigación Operativa y, más recientemente, de la Inteligencia Artificial. Dichos avances se han producido básicamente en tres terrenos:

- *El modelo de la red*; incorporando topologías más complejas que la de simples actividades relacionadas de manera determinista por relaciones de precedencia. De este modo hay precedencias basadas en operaciones lógicas, estocásticas, incluso ciclos en la red.
- *Los modos de ejecución*; tanto en lo relativo a los atributos de las actividades y sus restricciones temporales como a los recursos que pueden ser deterministas, estocásticos o borrosos y que pueden estar funcionalmente relacionados como en el caso del problema denominado *multi-modo* en el que el consumo de recursos determina la duración de una actividad.
- *Los métodos de solución*; optimización algorítmica por métodos exactos o aproximados, basados en simulación, en agentes, en el tratamiento analítico del espacio de las soluciones, en el análisis de la topología de la red, etc.

Una descripción detallada del estado de la cuestión bastante actualizado puede encontrarse en [Tav99b] y [DH02]. Textos básicos de Investigación Operativa sobre el problema son [AMO93], [ME78] y [Elm70]. De profesores de la Universidad de Sevilla, [Lar87] y [Lar60].

4.3. Gestión de proyectos con recursos limitados

La tipología de problemas que pueden plantearse con estos modelos es extensísima. A fin de identificarlos de forma unívoca se han propuesto diversas *taxonomías* [BDM$^+$99, DH02] . Básicamente consisten en generalizaciones de la clasificación tradicional de los problemas de secuenciación de tareas en talleres de máquinas. Existe además una nomenclatura basada en las siglas del nombre estándar del problema,

algo menos precisa que las propuestas en las referencias, pero que es la que seguiremos en este trabajo por tratarse de la más extendida.

En términos generales se formula como un problema de programación matemática en el que:

- Las *variables de decisión* son las fechas en que se programa el inicio de las actividades
- Las *restricciones* incorporan tanto las *relaciones de precedencia* como la *disponibilidad* en términos de cota superior (e inferior) de los recursos.
- La *función objetivo* describe criterios tales como la minimización de la duración total, la nivelación del empleo de recursos, la minimización del riesgo de retraso, la maximización del valor actual neto así como otros indicadores coste-beneficio. En muchas ocasiones la función objetivo es una combinación ponderada de estos criterios. A su vez puede implementarse en términos deterministas o como funciones estocásticas en términos de valor esperado o cuantiles extremos.

Una función objetivo se denomina *regular* si no es decreciente con la duración de las actividades. Los problemas de minimización de la duración total, minimización del retraso de las actividades, etc. son problemas con una función objetivo regular. Una función objetivo es *no-regular* en caso contrario. Los problemas ligados a la minimización del coste o del valor actual y aquellas que contemplan una relación inversa entre duración y empleo de recursos, son ejemplo de funciones objetivo no regulares [DH02].

Existe una amplísima variedad de modelos a partir de las diferentes combinaciones posibles cómo ya se ha señalado. Pero siguiendo a [Tav99a] se puede caracterizar dicha variedad de una manera simplificada atendiendo a tres dimensiones:

- El criterio que se sigue
- Las características de las actividades
- Las características de los recursos

Conviene señalar alguna de las cuestiones de entre estas variantes que resultan clave a la hora de tratar los diversos problemas:

Dimensión	**Indicador**	**Variantes**
Criterios	Temporal Empleo de recursos Nivelado de recursos Valor actual neto u otro criterio económico	Cualquier composición de los anteriores
Actividades	Duración Precedencias Modo Posibilidad de interrupción	Deterministas o estocásticas
Recursos	Monetarios No monetarios	Renovables o no Deterministas o estocásticos

Cuadro 4.1: Elementos que determinan los diferentes tipos de problemas de programación de proyectos según [Tav02]

- La implementación de las actividades puede seguir un único patrón o se admiten variaciones en la intensidad de la misma en cada intervalo de tiempo. El primer caso es el problema *unimodal*, el segundo el *multimodal* y el tercero el caso *continuo*.
- La segunda variante respecto de las alternativas es la *posibilidad de suspender y posteriormente retomar* la implementación de las actividades (*preemptive*) o no (*non-preemptive*).
- En tercer lugar, las variables formuladas (duración y/o requerimiento de recursos pueden ser magnitudes *deterministas* o, alternativamente, variables aleatorias o *estocásticas*.
- Por último, el proceso de toma de decisiones puede ser *estático* o *dinámico*, es decir empleando una programación y asignación de tareas y recursos predeterminada desde el inicio del proyecto o posibilitando modificaciones a lo largo del ciclo de vida del proyecto.

Evidentemente, los casos más complejos contemplan actividades multimodo o en modo continuo, duraciones y/o disponibilidades de recursos no deterministas y la posibilidad de la reprogramación dinámica a lo largo del ciclo de vida.

4.3.1. El problema RCPSP determinista con un solo modo de ejecución

Comenzaremos por el problema más simple: el problema de la secuenciación de proyectos con recursos limitados, *resource constrained project scheduling problem*, RCPSP en la nomenclatura tradicional que ha sido y sigue siendo objeto de muchísima atención y trabajo. El modelo que se presenta es una extensión del utilizado en la programación sin limitación de recursos con la notación *AEN* (actividades en nodos). Variables de decisión:

- t_i tiempo de comienzo de la actividad i.
- $S(t)$ conjunto de actividades que se procesan durante el instante t.

Geometría de la red:

- N, conjunto de nodos de la red, corresponden a las actividades. Existen dos actividades ficticias, la *0* predecesora de todas las demás y la n que es la última.
- $A = \{(i, j)\}$ conjunto de arcos de la red que indican las relaciones de precedencia - en este caso i precede a j.

Datos:

- p_i duración de la actividad i. Se consideran $n+1$ actividades, luego $\forall i \in N$ se tiene que $i = 0, \ldots, n$. Además, las actividades ficticias inicial y final cumplen que $p_0 = p_n = 0$
- r_{ik} consumo de recurso tipo k al realizar la tarea i.
- b_k cantidad disponible de recurso k.

Parámetros:

- K, número de recursos que intervienen en la realización del proyecto.
- T, horizonte temporal.

Con todo ello se puede escribir el modelo siguiente:

$$\text{mín } t_e \tag{4.1}$$

Sujeto a:

$$t_i - t_j \geq p_j \quad \forall (i,j) \in A \; \forall i \in N \tag{4.2}$$

$$\sum_{i \in S(t)} r_{ik} \leq b_k \quad t = 1, \ldots, T \quad k = 1, \ldots, K \tag{4.3}$$

$$t_i \geq 0 \tag{4.4}$$

Esta formulación del modelo es conceptual pero conlleva la dificultad para hacerlo operativo de la definición del conjunto de actividades que se realizan en el instante *t*, *S(t)*.

Con el objeto de resolver este problema, se plantea una variación sobre la formulación del modelo covirtiéndolo en un problema de *programación mixta* [PWW69]. Para ello se define la variable *binaria* x_{it} que toma el valor de 1 si la actividad i se completa en el instante t (téngase en cuenta que el tiempo está discretizado) y 0 en caso contrario. El problema queda como:

$$\text{mín} \sum_{t=EFT_n}^{LFT_n} t \cdot x_{nt} \tag{4.5}$$

Sujeto a:

$$\sum_{t=EFT_i}^{LFT_i} x_{it} = 1 \quad i = 1, \ldots, n \tag{4.6}$$

$$\sum_{t=EFTi}^{LFT_i} x_{it} \leq \sum^{LFT_j} t = EFT_j t \cdot x_{jt} - p_j \quad \forall (i.j) \in A \tag{4.7}$$

$$\sum_{i=1}^{n} \sum_{q=max\{t,EFT_i\}}^{min\{t+d_i-1,LFT_i\}} r_{ik} \cdot x_{iq} \leq b_k \quad k = 1, , K \; t = 1, \ldots, T \tag{4.8}$$

$$x_{it} \in \{0,1\} \; i = 1, \ldots, n \quad t = EFT_i, \ldots, LFT_i \tag{4.9}$$

Donde EFT_i y LFT_i son respectivamente la fecha más temprana, *earliest start*, y más tardía, *latest start*, de la actividad i-ésima.

Las restricciones respectivamente imponen, por el orden en el que se presentan, la necesidad de que todas las actividades se hayan completado, las relaciones de precedencia y el límite de consumo de recursos.

4.3.2. El problema RCPSP determinista con diferentes modos de ejecución - MRCPSP

El problema se basa en los mismos supuestos del caso anterior, pero también incluye la posibilidad de que el modo de realizar el procesamiento de la actividad haga variar la cantidad consumida de recurso. El modelo presentado en [DH02] emplea una variable de decisión $x_{i_{mt}}$ que cumple: $x_{imt}=\begin{cases} 1 & \text{si } i \text{ comienza a realizarse en modo } m \text{ en } t \\ 0 & \text{de lo contrario} \end{cases}$
El modelo es el siguiente:

$$\text{mín} \sum_{t=EFT_n}^{LFT_n} t \cdot x_{n1t} \tag{4.10}$$

Sujeto a:

$$\sum_{m=1}^{M_i} \sum_{t=EFT_i}^{LFT_i} x_{imt} = 1 \quad i = 1, \ldots, n \tag{4.11}$$

$$\sum_{m=1}^{M_i} \sum_{t=es_i}^{ls_i} (t + d_{im}) x_{imt} \leq \sum_{m=1}^{M_j} \sum_{t=es_j}^{ls_j} t \cdot x_{jmt} \quad \forall (i,j) \in A \tag{4.12}$$

$$\sum_{i=1}^{n} \sum_{m=1}^{M_i} r_{imk} \sum_{s=max\{t-d_{im}, es_i\}}^{min\{t-1, ls_i\}} x_{ims} \leq b_k \quad k = 1, \ldots, K \tag{4.13}$$

$$x_{imt} \in \{0,1\} \; i = 1, \ldots, n \quad m = 1, \cdots, M_i \quad t = es_i, \ldots, ls_i \tag{4.14}$$

En este caso, las fechas de referencia son es_i, el inicio más temprano, *earlier start*, y ls_i, el inicio más tardío, *latest start*. r_{imk} es el consumo del recurso k-ésimo por parte de la actividad i en el modo m (de entre los $1, \ldots, M_i$ posibles para esa actividad). La duración de las actividades depende del modo a través de d_{im}. En el modelo aquí presentado, el recurso es *renovable* y las actividades no pueden suspenderse una vez iniciadas.

4.3.3. El problema RCPSP con ventanas de tiempo

El problema RCPSP con ventanas de tiempo, tambien denominado TCPSP - *time constrained project scheduling problem-* es una variante del problema RCPSP en el que las actividades solo pueden ejecutarse en un intervalo de tiempo determinado, la *ventana de tiempo*, de manera análoga a como se ha descrito el problema de planificación táctica de capacidad restringido por el tiempo en 3.3.1.

A su vez es un caso particular del problema *RCPSP/max* o también denominado como problema con relaciones de *precedencia generalizadas* (página 53). Cualquier problema con relaciones de precedencia generalizadas se puede modelar empleando un *retraso mínimo y/o máximo* entre las fechas de inicio de dos actividades que se encuentran ligadas por una de estas relaciones. Como en cualquier proyecto se incorporan dos actividades ficticias, una inicial y otra final, la imposición de una ventana de tiempo absoluta a cualquier actividad se puede resolver simplemente imponiendo un retraso mínimo y uno máximo (descontando la duración de la actividad implicada) respecto de la actividad ficticia inicial.

Para modelar el problema con ventanas de tiempo se definen:

- N, conjunto de nodos de la red, corresponden a las actividades. Existen dos actividades ficticias, la *0* predecesora de todas las demás y la$n$ que es la última.
- $[r_i, d_i]$ ventana de tiempo, fecha de inicio y plazo máximo (*release* y *deadline*), de la actividad i
- $A = \{(i, j)\}$ conjunto de arcos de la red que indican las relaciones de precedencia - en este caso i precede a j.
- l_{ij} retraso entre el inicio de i y el de j si $(i, j) \in A$
- p_i duración de la actividad i. Se consideran $n+1$ actividades, luego $\forall i \in N$ se tiene que $i = 0, \ldots, n$. Además, las actividades ficticias inicial y final cumplen que $p_0 = p_n = 0$
- q_{ik} consumo de recurso tipo k por intervalo al realizar la tarea i.
- b_{kt} cantidad disponible de recurso k en el intervalo discreto t.

- K, número de recursos que intervienen en la realización del proyecto.

Además se definen dos variables binarias:

- x_{it}, vale 1 si la tarea i se realiza en el intervalo discreto t y 0 de lo contrario
- s_{it}, vale 1 si la tarea i se realiza en el intervalo discreto t y 0 de lo contrario

El espacio de las soluciones admisibles vendrá definido por:

$$\sum_{d_i-1}^{t=r_i} x_{it} = p_i \ \forall n \tag{4.15}$$

$$\sum_{n}^{0} x_{it} q_{kt} \leq b_{kt} \ \forall k, t \tag{4.16}$$

$$\sum_{d_i-1}^{t=r_i} s_{it} = 1 \ \forall n \tag{4.17}$$

$$x_{i0} = s_{i0} \ \forall i \tag{4.18}$$

$$x_{it} \leq x_{it-1} + s_{it} \ \forall i, t > 0 \tag{4.19}$$

$$s_j \geq s_i \ \forall (a, j) \in A \tag{4.20}$$

$$s_{it} = 0 \ \forall t \notin \{r_j, \ldots, d_j - p_j\} \tag{4.21}$$

$$x_{it} = 0 \ \forall t \notin \{r_j, \ldots, d_j - 1\} \tag{4.22}$$

La restricción 4.15 requiere que todas las actividades se completen; la restricción 4.16 que se respeten los límites de recursos; la 4.17 que todas las actividades comiencen, y las demás que se respeten las reglas de precedencia y las ventanas de tiempo.

La función objetivo en este problema puede adoptar varias formas, de modo semejante a como ocurría con el problema de la planificación táctica. Con la formulación expuesta, minimizar la duración de todo el proyecto se formula simplemente como:

$$\text{mín} \sum_{T}^{t=0} t \cdot s_n t \tag{4.23}$$

El problema de esta formulación es que el límite es la propia solución. El horizonte temporal se puede acotar, on obstante, con la suma de la duración de todas las actividades del proyecto,$\sum_{n-1}^{1} p_i$.

En el entorno de una programación operacional, otros objetivos podrían ser:

- minimizar el plazo de terminación de algunas, o todas, las tareas para asegurar la robustez de la programación ganando holguras respecto al plazo final
- equilibrar el empleo de recursos
- e incluso relajando las restricciones de recursos, minimizar el coste del empleo de recursos extraordinarios

4.3.4. El problema multiproyecto - RCMPSP

El problema multiproyecto admite básicamente dos tratamientos diferentes:

- La incorporación de todos los proyectos en una única red, es decir, modelar el conjunto de proyectos como un *metaproyecto* en el que por medio de actividades ficticias se conectan entre si las redes parciales de cada proyecto independiente. Este tratamiento es adecuado en un entorno de *programas* y desde el punto de vista de la existencia de un objetivo común a todos los proyectos.
- La consideración explícita de los proyectos de manera individualizada. Este tratamiento es adecuado cuando los objetivos de los proyectos son distintos, al menos al nivel operacional. En este caso la principal diferencia con los casos de un sólo proyecto se establece en la formulación de *la función objetivo*.

Este segundo caso es el que nos interesa pues el primero se reduce a los casos ya descritos anteriormente.

Entre las formulaciones de función objetivo adecuadas para el problema multiproyecto [BY06] señalan las siguientes:

- la minimización del tiempo de compleción de los proyectos
- la minimización del retraso acumulado de los proyectos

- la minimización del retraso medio de los proyectos
- la minimización del coste total
- la minimización del coste (penalización) del retraso
- maximizar el empleo de los recursos (penalizar los recursos ociosos)
- maximizar el valor descontado de los proyectos
- etc.

Como es lógico, la selección de la función objetivo depende básicamente del problema en estudio y también de la posición jerárquica desde la cual se aborda. En el caso que nos ocupa, la posición jerárquica es el nivel operacional. Se parte de una previa asignación de recursos y fechas límite que proviene de la planificación táctica. Por ese motivo los objetivos pueden verse a dos niveles:

- Desde la dirección de cada proyecto individual el objetivo es minimizar el retraso relativo de *cada* proyecto
- Desde la dirección conjunta el objetivo es minimizar el retraso de *todos* los proyectos de la cartera

Puesto que en la etapa de planificación táctica se han atribuido a los proyectos una fechas límite en función de la asignación de recursos establecida, los retrasos relativos deberán ser con relación a las duraciones establecidas en esas fechas límite. De esta manera, podemos plantear los objetivos de minimización de los retrasos como:

- El retraso relativo de cada proyecto es la desviación respecto a la duración prevista en la planificación táctica como un porcentaje de la misma
- El retraso relativo de la *cartera* de proyectos es el retraso máximo de los proyectos en curso

Así pues, en una cartera con P proyectos, se define una actividad ficticia para cada uno de ellos y_{pt} que vale 1, si el proyecto p termina en el momento$t$ y vale 0 de los contrario. Cada proyecto tiene una fecha asignada de finalización f_p que proviene de la planificación táctica.

El problema de minimización del retraso relativo de cada proyecto p es:

$$\text{mín}\ \frac{t \cdot (y_{pt} - f_p)}{f_p} \tag{4.24}$$

El problema de minimización del retraso *medio* es:

$$\text{mín}\ \frac{1}{P} \sum_{p=1}^{p=P} \frac{t \cdot (y_{pt} - f_p)}{f_p} \tag{4.25}$$

Y el problema de la minimización del máximo retraso en la cartera:

$$\text{mín}\ \underset{p=1}{\overset{p=P}{\text{máx}}} \left[\frac{t \cdot (y_{pt} - f_p)}{f_p} \right] \tag{4.26}$$

4.3.5. Otras variantes del problema

El número de variantes del problema es prácticamente interminable. Existen:

- El problema multiproyecto con múltiples modos de realización, *MRCMPSP*
- El problema de ventanas de tiempo con múltiples modos de ejecución, *MRCPSP/max*
- El mismo que el anterior pero en contexto multiproyecto, *MRCMPSP/max*
- ...

De la literatura revisada, el problema con más complejidad y opciones encontrado es el MRCMPSP-GPR/EXP, siglas que corresponden a *multi-mode resource constrained multi-project scheduling problem with generalized precedence relations and expediting resources*en el cual se añade la posibilidad del empleo de recursos extraordinarios[Fre01].

4.4. Métodos exactos para la resolución de los problemas RCPSP

El problema RCPSP es un problema combinatorio de extraordinaria complejidad computacional. Se trata de un problema *NP-duro* como se demuestra en [DH02]. Los métodos exactos para la solución del problema RCPSP se basan generalmente en procedimientos de enumeración implícita (*Branch-and-Bound*). Estos procedimientos parten de una solución inicial y van generando un árbol de particiones del espacio de las soluciones (ramas) y acotándolo. Existen diversas estrategias para generar el árbol de exploración y las cotas que aparecen descritos en la literatura. Básicamente se agrupan en dos tipos: búsqueda en profundidad (*depth-first*) y búsqueda en la frontera (*best-first* o *frontier search*). Los algoritmos van seleccionando los nodos que se exploran y ramificando a partir de ellos de acuerdo con la estrategia adoptada.

En [HD98] se analizan los diversos criterios para acotar y las reglas de dominancia que permiten reducir el espacio de exploración. Se presentan tres alternativas para la solución del MRCPSP. En [HRD98][LWT01][FMK01] se revisan también los diferentes métodos aplicados al problema RCPSP.

Entre las aplicaciones más recientes relacionadas con la aplicación de estos métodos a la gestión de proyectos que se han localizado se encuentran las siguientes:

- En [BdOBW08] se aplica los métodos branch and bound a generar equipos de trabajo para proyectos de desarrollo software
- En [SM07b] se construye un modelo para decidir sobre la aceptación o no de pedidos para su posterior secuenciación
- En [BZ06] se toma los métodos branch and bound como punto de partida para un heurístico de secuenciación de tareas en un entorno de manufactura virtual

Esta última aplicación, en la que el método es el punto de partida de un procedimiento heurístico, es significativa porque, si bien la potencia del *branch and bound* es grande, la complejidad de los problemas RCPSP ha llevado a que la investigación se centre en métodos heurísticos en lugar de perseguir los métodos exactos. Esto es lo que se trata

en los siguientes apartados. Las variantes más complejas del problema (MRCPSP, MRCMPSP, ...) son todavía más complejos, por lo que prácticamente todos se abordan con metodologías heurísticas.

4.5. Métodos heurísticos para la solución de los problemas RCPSP

Los métodos heurísticos aplicados en el problema RCPSP caen básicamente dentro de dos categorías, los métodos *constructivos* y los métodos de *mejora*.
Los métodos constructivos se centrar en construir una secuencia de actividades factible, es decir que cumpla las restricciones tanto de precedencia como de recursos. Para ello se parte de una primera secuencia "vacía" a la que se van incorporando actividades de acuerdo con una serie de *reglas de prioridad* y cumpliendo con las restricciones para así contar con una *secuencia viable*.

Los métodos de mejora se centran en obtener mejores soluciones a partir de contar con esa secuencia viable de actividades. Para ello realizan una *búsqueda local* en torno a la misma que conduzca a un óptimo local. Con el fin de eludir que éste sea subóptimo en términos del espacio total de soluciones, a veces los procedimientos de búsqueda permiten "deteriorar" temporalmente la solución. Esto, a su vez, puede dar lugar a la aparición de *ciclos*. Con el fin de evitarlo se emplean técnicas *metaheurísticas* desarrolladas en el terreno de la inteligencia artificial como algoritmos genéticos, búsqueda tabú, recocido simulado, etc.

Ejemplo 1 *Para ilustrar los puntos que se exponen a continuación se emplea el ejemplo siguiente, tomado de [WL77]:*

Suponemos un proyecto, cuya duración queremos minimizar, con 8 actividades más dos ficticias, de duración 0, que señalan el inicio y el final del proyecto. El cuadro 4.2 representa la duración, el consumo de recursos por unidad de tiempo y las relaciones de precedencia inmediata de cada una de las actividades. Se supone que la capacidad máxima del recurso renovable único es de 10 umidades de recurso por unidad de tiempo.

Tarea	Duración	Recursos	Prec.
1	0	0	0
2	4	9	1
3	2	3	1
4	2	6	1
5	2	4	1
6	3	8	3
7	2	7	4
8	3	2	5,7
9	4	1	6,8
10	0	0	2,9

Cuadro 4.2: Proyecto ejemplo tomado de [WL77]

La red del proyecto, en notación AEN, es la que presenta la figura 4.1 de la página tab:redejemplo.

4.5.1. Construcción de secuencias

Una secuencia o solución S_n es un vector de $n+1$ fechas s_i de inicio o de final f_i de las *n+1* acividades del proyecto, incluyendo las actividades ficticias inicial, 0, y final, *n*. Se denomina longitud o durción de la secuencia, *T(S)*, al plazo de compleción del proyecto si se sigue esa secuencia, es decir, a la fecha de compleción de la actividad final. Una *secuencia factible* es aquella que respeta las restricciones de tiempo, precedencia y recursos. Dada una secuencia S, se denominan actividades *activas* en un momento t al conjunto *A(t)* de actividades que están ejecutandose en ese momento de seguirse esa secuencia.

Para la construcción de secuencias se utilizan los denominados *esquemas de generación se secuencias* (SGS)[Bal03]. Los esquemas de generación de secuencias se agrupan básicamente en dos categorías, *serie* y *paralelo*. Las primeras parten de una secuencia vacía y van iterando *incorporando actividades*. Las segundas, partiendo también de una secuencia vacía realizan una *iteración temporal*.

Algoritmo 1 *Esquema de Generación de Secuencias en Serie*

a) *Se incorpora la actividad ficticia inicial*

b) *Para* $i = 1$ *hasta* $n - 1$

1) *Determinar* E, D, *y* F
2) *Seleccionar* j *una actividad elegible*
3) *Determinar la fecha más temprana de inicio de la actividad* j *compatible con las precedencias*
4) *Determinar la fecha más temprana de inicio de la actividad* j *compatible con la disponibilidad de recursos*
5) *Incorporar a* S *dicha actividad con fecha de finalización a partir de la fecha más temprana compatible con 2.3 y 2.4*

c) *Incorporar a la secuencia la actividad ficticia final con duración 0. La fecha de inicio (y final) es la duración de la secuencia*

El algoritmo general de secuenciación en serie se realiza en tantas etapas como actividades hay que secuenciar. En cada etapa se diferencian dos conjuntos; S, actividades ya secuenciadas y E, conjunto de actividades elegibles, es decir actividades que aun no han sido secuenciadas y cuyas predecesoras ya lo han sido. Igualmente se determina en cada momento t, D, la disponibilidad de recursos, es decir, la cantidad de recursos que no está siendo usada por las actividades activas en ese momento. Por último se determina F, el vector de los tiempos de finalización de las actividades comprendidas en S.

Las diferentes implementaciones de este algoritmo se diferencian por el criterio de selección de la siguiente actividad elegible (paso 2.2).

Ejemplo 2 *Utilizando una de las reglas de prioridad más simples, la holgura mínima, se puede ejemplificar la generación de una secuancia serie. Previamente es necesario determinar las holguras, para lo que se aplica el algoritmo clásico del CPM. Las actividades del camino crítico son precisamente las de holgura 0. Como se ha indicado el algoritmo procede de la forma siguiente:*

- *Se incorpora la actividad ficticia inicial,*$S = \{0\}$
- *Para* $i = 1$ *hasta 9*

Tarea	d_i	ES_i	EF_i	LS_i	LF_i	$SLACK_i$
1	0	0	0	0	0	0
2	4	0	4	7	11	7
3	2	0	2	2	4	2
4	2	0	2	0	2	0
5	2	0	2	2	4	2
6	3	2	5	4	7	2
7	2	2	4	2	4	0
8	3	4	7	4	7	0
9	4	7	11	7	11	0
10	11	11	11	11	11	0

Cuadro 4.3: Fechas y holguras del ejemplo

- $i = 1, S = \{1\}$
  - $E = \{2, 3, 4, 5\}, D = 0, F = \{0\}$
  - mín $[SLACK_j] = 0 \Rightarrow j = 4$
  - $ES_4 = 0$
  - $D = 0, ES_4 = 0$
  - $s_4 = 0, f_4 = s_4 + d_4 = 2$
  - $S = \{1, 4\}$
- $i = 2$
  - $E = \{2, 3, 5, 7\}, D = 4, F = \{0, 2\}$
  - mín $[SLACK_j] = 0 \Rightarrow j = 7$
  - $ES_7 = 2$
  - $D = 4, ES_7 = 2$
  - $s_7 = 2, f_7 = s_7 + d_7 = 4$
  - $S = \{1, 4, 7\}$
- $i = 3$
- $\cdots$
- $\cdots$

- $\boldsymbol{S}$=$\{1, 4, 7, 8, 9, 5, 3, 6, 2\} i = 9$
- $s_{10} = 11, f_{10} = 11, T = 11$

Por su parte el algoritmo de secuenciación en paralelo se basa en iterar en el tiempo. A cada paso de interación le corresponde un tiempo

t_g. En cada momento hay una serie de actividades ya completadas, C, otras activas, A, y otras elegibles definidas como en el caso anterior. El algoritmo procede de la forma siguiente:

Algoritmo 2 *Esquema de Generación de Secuencias en Paralelo*

a) *Inicializar secuenciando la actividad ficticia inicial*

b) *Mientras queden actividades sin completar y sin estar activadas*

1) *Incrementar la iteración un paso*
2) *Fijar el tiempo t_g en la primera fecha de finalización de las actividades activas*
3) *Determinar* C, A, D *y* E
4) *Mientras existan actividades elegibles*
   - a′ *Seleccionar una actividad elegible*
   - b′ *Fijar su fecha de inicio en el tiempo t_g*
   - c′ *Determinar su fecha de finalización*
   - d′ *Actualizar* C, A, D *y* E

c) *Incorporar a la secuencia la actividad ficticia final con duración 0. La fecha de inicio (y final) es la duración de la secuencia*

Una vez más, aquí las diferentes implementaciones dependen del criterio elegido para el paso 2.4.1.

Ejemplo 3 *El algoritmo de generación de secuencias paralelo es semjante pero va avanzando por intervalos de tiempo desde $t = 0$ hasta $t = 11$, pasando por $t = 4$ y $t = 7$.*

Cualquiera de los dos procedimientos permite obtener secuencias óptimas en el problema no restringido en los recursos. Sin embargo no es así en el caso general.

Teniendo en cuanta la limitación de recursos, las secuencias obtenidas se denominan *secuencias activas* o *justificadas a la izquierda* si ninguna de las actividades que comprende puede iniciarse antes de la fecha secuenciada sin provocar un retraso en otra actividad. *Justificada a la derecha*, en cambio, es el caso en el que el retraso de cualquiera provoque un retraso en alguna de las demás. Si se permite la suspensión

temporal de una actividad activa, una secuencia *sin retraso* es auquella que incluso en ese caso da lugar a un retraso si se adelanta. [Kol96] demuestra que el algoritmo serie construye secuencias activas (justificadas a la izquierda) mientras que el paralelo construye secuencias sin retraso. En ambos casos los algoritmos permiten generar todas las secuencias de ambos tipos.

El conjunto de todas las secuencias sin retraso en un subconjunto de las secuencias activas por lo que a priori el algoritmo paralelo construye un espacio de búsqueda para la mejora posterior de una cardinalidad menor. El problema radica en que se demuestra que puede dejar fuera la(s) solución(es) óptima(s) para un problema con función objetivo regular, cosa que no ocurre con el algoritmo serie. De ahí que hayan aparecido diversos algoritmos mixtos, serei-paralelo, que buscan aprovechar las ventajas de ambos métodos [KP01] [KH06].

4.5.2. Reglas de prioridad

Tanto en el algoritmo serie como en el paralelo se requieren reglas para seleccionar las actividades elegibles que se incorporan a la secuencia. El modo más habitual de hacerlo es mediante las *reglas de prioridad*.

Una regla de prioridad se caracteriza por los siguientes elementos:

- Una función real para asignar un valor a cada actividad, el *valor* de la prioridad
- Un criterio de *extremos*, máximo o mínimo
- Un procedimiento para el *desempate* entre dos actividades con igual valor de la prioridad, que habitualmente es simplemente el valor de la etiqueta numérica asignado a cada actividad

Las reglas, a su vez, pueden ser:

- *Locales o globales*, según se calculen con valores exclusivamente asociados a la actividad en consideración o empleen valores asociados a otras actividades
- *Estáticas o dinámicas*, según se mantengan constantes o varíen en el tiempo de ejecución del algoritmo

[h] Regla	Extremo	Valor de la prioridad
Activity number (AN)	min	j
Earliest finish time (EFT)	min	EFT_j
Earliest start time (EST)	min	EST_j
Greatest rank positional weight (GRPW)	max	$dj + (\sum d_i \forall i_j)$
Greatest resource demand (GRD)	max	$dj(\sum r_{jk})$
Longest processing time (LPT)	max	d_j
Latest finish time (LFT)	min	LFT_j
Latest start and finish time (LSTLFT)	min	$LST_j + LFT_j$
Latest start time (LST)	min	LST_j
Minimum free slack (MINFREE)	min	$(EST_i \forall i \in IS_j) - EFT_j$
Minimum total slack (MINSLK)	min	$LST_j - EST_j$
Number of immediate successors (NIS)	max	IS_j
Remaining work (RWK)	max	$d_j + (\sum mind_i \forall i \in AS_j)$
Shortest processing time (SPT)	min	d_j

Cuadro 4.4: Algunas de las reglas de prioridad más conocidas

- Referidas a la *topología* de la red del proyecto, a los *tiempos* o a las *recursos*
- Por último, algunas son aplicables a los dos tipos de algortimos, serie y paralelo, y otras son adecuadas sólo para alguno de ellos.

Entre las reglas más corrientes se encuentran las que se presentan en la tabla siguiente, extraída de [LTB06].

4.5.3. La representación de las soluciones

Con el fin de poder aplicar los procedimientos heurísticos basados en reglas de prioridad así como los métodos más avanzados que se exponen más adelante, se hace necesario contar con una forma de representar las soluciones del problema RCPSP.

En la literatura [Bal03] se proponen diversos modos de representación, cada uno de los cuales se adapta en mejor o pero grado a las diferentes mtodologías de generación y mejora existentes. Básicamente son las siguientes:

Lista de actividades Una lista de actividades es una permutación de las actividades $\lambda = (j_1, j_2, ..., j_n)$ que es compatible con las prioridades. En general una secuencia activa puede admitir más de una representación en términos de lista de actividades.

Vector de prioridades Un vector de prioridades o *random key* es un vector que asigna a cada actividad un valor real correspondiente a la prioridad asignada por la regla correspondiente. Los algoritmos de secuenciación serievan eligiendo actividades de acuerdo con esos valores (máximo o mínimo, según el caso). Cada secuencia admite infinitas representaciones en términos de vector de prioridades.

Vector de reglas de prioridad Un vector de reglas de prioridad aplica a cada actividad a ser secuenciada una regla de prioridad distinta. Este modelo se adopta cuando se verifica que según las características del proyecto no existe una regla de prioridad que domine claramente a las demás. Sin embargo esta representación puede no incluir a todas las secuencias posibles y, por tanto, excluir las óptimas.

Vector de traslación Un vector de traslación, *shift vector*, asigna a cada actividad un retraso a su fecha más temprana de incio respecto de la de finalización de la última predecesora. El problema de esta representación radica en que en algunos casos proporciona soluciones no factibles en términos de recursos. Por otra parte sí recogerá a todas las soluciones y, dentro de ellas, a las óptimas.

Vector de tiempos de inicio En este caso la representación de la solución es la solución misma. Por tanto existe una representación unívoca de todas las soluciones.

Esquema de secuencias Un esquema de secuencias contiene cuatro conjuntos disjuntos de pares de actividades. Las *conjunciones* son los pares de actividades en los que la primera debe finalizar para que comience la segunda. Las *disyunciones* son aquellos pares de actividades que no pueden coincidir en el tiempo. Las *relaciones paralelas* son aquellas en las que las acitivdades deben coincidir al menos en un periodo de tiempo y las *relaciones flexibles* todos aquellos pares que no están sometidos a ninguna restricción. Esta representación exhaustiva del proyecto puede originar un problema NP-duro simplemente para determinar si existe una secuencia

factible dentro de un esquema de secuencias.

Ejemplo 4 *La solución presentada en el ejemplo anterior,*

$$S = \{1, 4, 7, 8, 9, 5, 3, 6, 2, 10\}$$

está en la forma de lista de actividades.
La solución en forma de vector de prioridades o random key *es, en este caso con la regla MINSLK simplemente un vector cuyos elementos son las holguras de cada una de las actividades ordenadas lexicográficamente:*

$$\{0, 7, 2, 0, 2, 2, 0, 0, 0, 0\}$$

El vector de tiempos de inicio *de esta solución está formado por las fechas programadas, es decir, es la propia solución:*

$$\{0, 7, 2, 0, 2, 4, 2, 4, 7, 11\}$$

En este caso, la limitación de recursos hace que la única solución óptima sea aquella en la que las actividades están ajustadas a la derecha, es decir, que empiezan en su fecha más tardía, por eso el vector de tiempos de inicio coincide con la columna (LF_i) del cuadro 4.3.

La representación de la solución óptima en forma de vector de traslación *o* shift vector *es la misma que la* random key *puesto que es un vector que desplaza a la derecha las fechas de inicio respecto de la más temprana posible y como se ha dicho aquí la solución óptima está ajustada a la derecha. Por tanto el vector es:*

$$\{0, 7, 2, 0, 2, 2, 0, 0, 0, 0\}$$

Nótese que esta es la única solución óptima al problema.

Por último, el esquema de secuencias *contiene cuatro conjuntos disjuntos de pares de actividades. Las* conjunciones *son los pares de actividades en los que la primera debe finalizar para que comience la segunda. En nuestro caso se trata de todas las combinaciones que:*

- *están obligadas por las relaciones de precedencia, bien directamente, cómo (1,2), (1,3), (1,4),(1,5), (2,10), (3,6), (4,7), (5,8), (6,9), (7,8), (8,9) y (9,10)*
- *o a través de otras relaciones como (1,7), (1,10), etc.*

Las disyunciones *son aquellos pares de actividades que no pueden coincidir en el tiempo porque están limitadas porque la suma de los recursos necesarios son superiores a 10, como por ejemplo (7,6) o(4,2).*

Las relaciones paralelas *son aquellas en las que las actividades deben coincidir al menos en un periodo de tiempo, algo que en este problema no se produce.*

Por último las relaciones flexibles *todos aquellos pares que no están sometidos a ninguna restricción. En nuestro caso tendríamos precisamente los pares que constituyen la solución óptima.*

4.5.4. Métodos heurísticos y metaheurísticos para la solución del problema RCPSP

Dadas las características de complejidad del problema, el PCPSP es objeto de constante investigación en la aplicación de todo tipo de métodos y técnicas para su resolución. En la literatura existen amplias revisiones de los métodos aplicados tales como [KP01] [KH06] [HRD98] [DH02]. Nos limitaremos en este punto a comentar los métodos más empleados y poner algunos ejemplos.

Métodos multi-paso.

Un mètodo de mejora de las soluciones obtenidas por la generación de secuencias empleando reglas de prioridad es el de someter al grafo del proyecto a una secuenciación múltiple que a su vez puede producrise de dos formas:

Secuenciación hacia adelante y hacia atrás Una vez generada la secuencia en el modo descrito anteriormente, bien serie o paralelo, con una determinada regla de prioridad, se secuencia el grafo

inverso a partir de una cota superior de la duración, justificando hacia la derecha y, desplzando luego la solución hacia el momento t=0 ya que, generalmente, la secuenciación hacia atrás dejará una holgura.

Secuenciación bidireccional Se secuencian la actividad inicial a 0 y la final a una cota superior del problema. En cada paso de la iteración se determina si secuenciar hacia adelante o hacia atrás y la actividad que debe secuenciarse aplicando una regla de prioridad.

Muestreo aleatorio.

En lugar de aplicar una regla de prioridad estricta, la actividad a secuenciar se obtiene mediante muestreo de las actividades elegtibles, a su vez, en varias fromas posibles:

Muestreo aleatorio puro Se extrae aleatoriamente una actividad a secuenciar entre todas las posibles.

Muestreo sesgado La probabilidad de que una actividad sea seleccionada de acuerdo con una probabildiad que es proporcional al valor de la prioridad en una regla elegida de entre las habitualmente empleadas

Muestreo sesgado con penalización La probabildiad se basa no en el valor de la prioridad de cada actividad sino en el coste de oportunidad - *regret* - de no seleccionarla.

Heurísticos basados en metaheurísticas clásicas.

Entre las metaheurísticas clásicas que se exponen en la literatura se encuentran varios procedimientos conocidos como:

Búsqueda local Los métodos de búsqueda local evalúan la función objetivo en un entorno definido en el espacio de las soluciones en relación a una primera solución factible. Se "desplazan" a cualquier otro punto de ese entorno en el que la función objetivo mejore. Ese desplazamiento puede determinarse de varias formas: bien a la mejor solución posible después de realizar una exploración completa del entorno (*best fit strategy*) o bien a la primera que

se encuentre que mejore al valor actual de acuerdo con un orden de búsqueda predeterminado (*first fit strategy*). En ambos casos el desplazamiento se detiene cuando no existe posibilidad de mejorar.

Recocido simulado Esta técnica se denomina así por analogía con el tratamiento térmico al que se refiere su nombre. Básicamente es una mejora de la estrategia *first fit* con la variante de que admite la posibilidad de desplazarse a una solución peor que la actual con una probabilidad dada. Esa probabilidad se va reduciendo conforem el algoritmo va ejecutándose, por lo que se le denomina *temperatura* prosiguiendo con la analogía. El permtir que se opte por soluciones que no mejoran el objetivo pero que abren nuevos entornos a la evaluación persigue evitar que el algoritmo quede atrapado en un óptimo local. El descenso de la temperatura persigue acotar la recursión.

Búsqueda tabú Esta técnica es a su vez una mejora de la estrategia *best fit*. Elije la mejor solución en un entorno de la de partida aunque aquella sea peor que esta primera. Con el fin de no entrar en ciclos mantiene una lista de las soluciones ya visitadas que pasan a estar prohibidas en los próximos desplazamientos. Esta lista tabú es la que da el nombre al procedimiento. La lista tabú es limitada por lo que una solución puede "salir" de ella cuando hayan pasado tantas iteraciones como se establezca para determinar la longitud de la lista. Además de esta memoria a corto plazo se mantiene también una memoria de plazo largo en el que se conservan soluciones indicativas de regiones "atractivas" ya exploradas y de regiones inexploradas que permiten diversificar la búsqueda. Un procedimiento denominado reencadenamiento de trayectorias permite orientar los desplazamientos hacia esas regiones.

Algoritmos genéticos Esta técnica, al contrario que las técnicas descritas anteriormente donde se itera de solución en solución, opera con una población de soluciones. A partir de una población inicial se cruzan los distintos pobladores y se someten a mutaciones análogamente a los procesos de la evolución biológica. Una función de adaptación - *fitness* - evalúa a cada elemento y determina los supervivientes de la siguiente generación que vuelven a recombinarse y mutar. La función de adaptación se basa, obviamente, en la función objetivo del problema. Los algoritmos genéticos pueden

ser muy diversos según la forma de codificación de las soluciones - genes - y los procedimientos de combinación y mutación que se adopten.

Búsqueda dispersa La búsqueda dispersa - *scatter search* - genera una colección de soluciones diversas que almacena en un conjunto de soluciones de prueba. Posteriormente con técnicas de mejora local genera una o varias mejoras de esas soluciones iniciales. A continuación extrae de entre ellas un subconjunto reducido de las mejores soluciones que se denomina conjunto de referencia. Particiona el conjunto de referencia en varios subconjuntos, normalmente de dos elementos, para generar soluciones combinadas. De manera análoga a los algoritmos genéticos combina dichas soluciones para obtener un nuevo conjunto de soluciones de prueba.

Colonias de hormigas El método de la colonia de hormigas se basa en modelos provenientes de la vida artificial. Una generación de hormigas artificiales explora el espacio de las soluciones siguiendo criterios probabilísticos y, eventualmente, heurísticos específicos del problema. Las aristas que conducen a las mejores soluciones son marcadas por *feronomas* de manera que las sucesivas generaciones de hormigas retoman la búsqueda en las regiones más prometedoras exploradas por las generaciones precedentes.

Otras metaheurísticas descritas en la literatura.

La literatura sobre métodos de solución del 'problema RCPSP es extensísima. Además de las ya citadas metaheurísticas se presetan aplicaciones no convencionales basadas en la combinación de varias de las ya descritas y de otras modalidades tales como las técnicas de enjambre - *swarm* - , la relajación lagrangiana, el heurístico electromagnético y, por supuesto, heurísticos basados en búsqueda limitada a partir de los métodos exactos de *branch and bound* y cortes.

4.5.5. Resultados computacionales

Una amplia revisión computacional de los diversos métodos aparece en [KH06]. Contrastados con una librería estándar de problemas generados

Referencia	Técnica	Problema
[GMR08]	Algoritmo genético	RCMPSP
[MGR08]		RCPSP
[KJR06]		RCMPSP
[LCM04]		RCPSP
[AM01]		RCPSP
[Har98]		RCPSP
[APH04]		RCPSP
[VBQ08]		RCPSP
[BZ06]	Branch and Bound	RCPSP
[LTN06]		RCPSP
[RRK]	Búsqueda dispersa	RCPSP
[DRLV06]		RCPSP
[PAM04]	Búsqueda local	RCPSP
[Wal08]	Busqueda tabú	DCFPSP
[PHC]		RCPSP
[MWW08]		MRCPSP
[DER98]		RCPSP
[SM07a]	Colonias de hormigas	RCPSP
[TC06]		RCPSP
[LJF04]	Cortes mínimos	RCPSP
[DV06]	Electromagnetismo	RCPSP
[Sho06]	Muestreo aleatorio	RCMPSP
[VBQ05]	Multipaso	RCPSP
[TL03]		RCPSP
[CH08]	Multipaso, muestreo aleatorio sesgado	RCPSP
[GC03]	Primal-dual	RCPSP
[JMR+01]	Recocido simulado	MRCPSP
[Boc96]		RCPSP
[ASA06]		RCPSP
[BL03]		MRCPSP
[XMNU08]	Reglas de prioridad	RCPSP
[LTB06]		MRCPSP
[HK05]		MRCPSP
[JDSR08]	Swarm	RCPSP

Cuadro 4.5: Referencias de metaheurísticos en la literatura para diversos problemas del tipo RCPSP

aleatoriamente, denominada PSPLIB, que se usa como referencia entre los estudiosos del problema, los mejores algoritmos resultan ser los que reunen estas características:

- Generación de secuencias en *serie*
- Representación de las soluciones como *lista de actividades*
- metaheurísticos *mixtos* que combinan algoritmos genéticos u otros mecanismos basados en la población con otros sistemas de mejora (multipaso, autoadaptación, ...)

Para problemas de dimensión más reducida (hasta 1.000 secuencias posibles) el sistema de secuenciación hacia adelante y hacia atrás (*dorward-backward improvement* combinado con el muestreo alaeatorio puro proporciona resultados muy semejantes a los obtenidos con los metaheurísticos clásicos siendo un método mucho más sencillo. De hecho, para el subconjunto de problemas J120 (120 tareas) resulta incluso mejor que los mejores metaheurísticos.

En cuanto a las reglas de prioridad, muy empleadas en la práctica profesional para problemas de dimensión más reducida donde las técnicas metaheurísticas no se emplean más que en casos muy contados, en [BY06] se presenta un estudio en el contexto RCMPSP que arroja los siguientes resultados:

- Para proyectos *poco complejos*, es decir con una relación arcos/nodos relativamente reducida, la regla SASP ofrece los mejores resultados en términos de *retraso medio* de los proyectos que componen la cartera
- Si el *nivel de complejidad es elevado* la mejor regla es TWK-LST siempre hablando desde el punto de vista del *retraso medio*
- Si lo que se pretende es *minimizar el máximo retraso*, que es el punto de vista de los responsables de la cartera de proyectos, frente al punto de vista anterior que correspondería a los jefes de los proyectos individuales, la mejor regla resulta ser la MINWCS

Las reglas citadas, que corresponden al entorno multiproyecto y por tanto son más complejas que las antes expuestas, referidas al entorno de un proyecto único, se aplican siempre con esquemas de generación de secuencias *serie* a *lista de actividades* y su contenido es el siguiente:

SASP *shortest activity from the shortest project*; se selecciona de entre las actividades elegibles la actividad i del proyecto l que tiene el menor valor de $CP_l + d_{il}$. En caso de empate se elige la de número más bajo.

TWK-LST *total work contain + latest early start*; se selecciona la tarea con más contenido de trabajo pendiente, desempatando si es necesario con la que tiene la fecha más tardía de inicio menor.

MINWCS *minimum worst case slack*; se selecciona la tarea que cumple la siguiente relación:

$$\text{mín}\left(LS_i - \text{máx}\left[E_{ij}\,|(i,j) \in AP_t\right]\right) \quad (4.27)$$

donde E_{ij} es el tiempo más temprano que puede empezar la actividad j si la actividad i ha empezado en t y AP_t es el conjunto de todos los pares de actividades que puenden empezar en t y no están programadas. Esta regla, en ausencia de restricción de recursos equivale a la regla de mínima holgura, MINSLK.

4.6. Conclusiones y propuestas

El problema operacional tal y como se ha planteado es un caso del problema señalado en el apartado 4.3.3, *un problema de programación con restricción de recursos y con ventanas de tiempo*. En [NSZ03] se presentan los resultados computacionales paras diferentes tipos de problemas y función objetivo.

La función objetivo más adecuada, en principio, parece la de la minimización del tiempo total de terminación del proyecto. Al menos para una programación base inicial. Eventualmente cabe la posibilidad de fijar la minimización de la duración de una determinada cadena de actividades, en lugar de la del camino crítico, pero operacionalmente esas dos funciones objetivo son muy semejantes.

EL empleo de una función objetivo no regular, como el equilibrado de recursos o la minimización de flujos monetarios descontados no corresponden al problema al nivel que aquí se trata. En el primer caso, porque el desarrollo de *software* se realiza bajo la presión de los plazos. Es difícil, a nivel operacional, encontrarse con recursos ociosos si han

sido asignados a un proyecto en ejecución, salvo que sean muy especializados. En el segundo caso, sencillamente no corresponde a ese nivel de decisión. Así pues, se propone como *función objetivo* la minimización del tiempo total de terminación del proyecto.

En el caso de la minimización del tiempo total, proyectos pequeños (con menos de 100 actividades), pueden resolverse por diversos métodos *heurísticos y metaheurísticos*. Aplicar el *branch and bound* truncado y algoritmos *genéticos* dan el mejor resultado, con una desviación media del óptimo del entorno del 6 al 8 %. La aplicación de la *búsqueda tabú* resulta más apropiada para problemas de mayor tamaño. La *aplicación conjunta de varias reglas de prioridad* ofrece resultados apenas peores que los dos primeros procedimientos (desviación de la media del 10 %).

Las reglas que mejor resultado generan son:

- MINSLK - holgura mínima
- LST - menor plazo de inicio más tardío admisible
- RSM - *resource scheduling method*, que programa la actividad que al secuenciarse inducirá el menor retraso a las demás que quedan por secuenciar

La forma de representación más adecuada de las soluciones es la *lista de actividades* y la generación de secuencias, a partir del esquema *serie*. Las reglas de prioridad se calculan de forma *dinámica*, es decir, se determinan de nuevo cada vez que se secuencia una nueva actividad.

La aplicación de las técnicas *multiproyecto* al problema operacional no se considera adecuada teniendo en cuenta la estrategia de descomposición adoptada. En el nivel operacional, los equipos de cada proyecto han sido definidos desde la planificación táctica de capacidad. En condiciones normales, un director de proyecto tiene como variable de decisión la de asignar los componentes de su equipo a las distintas tareas de su proyecto.

Sólo en casos excepcionales podrá acudirse a recursos inicialmente asignados a otro proyecto (u otro proyecto podrá "quitarle" recursos) pero esa decisión debe mantenerse en el nivel táctico. Desde el punto de vista de la dirección de cada proyecto individual, se trata de un problema de *incertidumbre*. Este es el objeto del siguiente capítulo.

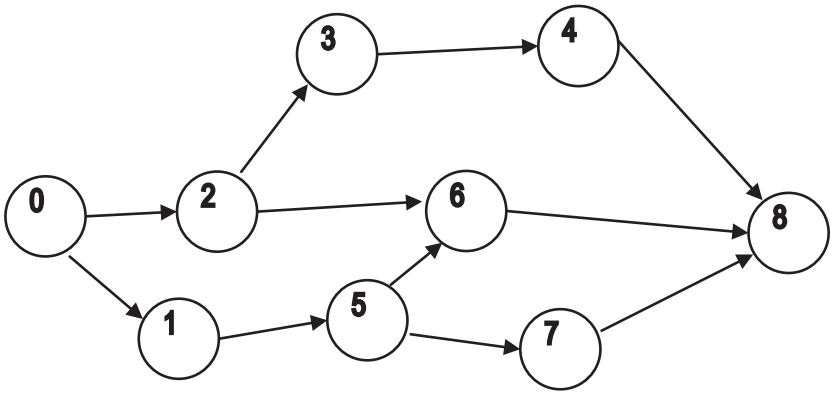

Figura 4.1: Red del ejemplo

Capítulo 5

El problema del riesgo y la incertidumbre en la gestión de proyectos con restricción de recursos

5.1. Objetivo y contenidos

Este capítulo tiene como objetivo describir las principales aproximaciones el tratamiento del riesgo y la incertidumbre en la gestión de proyectos. Los proyectos *software* se desarrollan en unas condiciones de incertidumbre mucho más fuertes que la mayoría de los proyectos para los que se han diseñado los métodos y modelos expuestos en el capítulo anterior. Las fuentes de incertidumbre son muy diversas. En gran medida provienen del propio entorno de los proyectos. El caso de la "volatilidad de los requisitos" es el más conocido. Pero la propia naturaleza del trabajo intelectual complejo que conlleva el desarrollo del *software* en muchas ocasiones, especialmente cuando se trata de *software* a medida y las interrelaciones entre esta complejidad y la incetidumbre provocada por el entorno obligan a tener en cuenta, además de los riesgos derivados de los compenentes elementales del proyecto (duración de actividades, disponibilidad de recursos, ...), los riesgos derivados de la inetracción de todos estos factores entre si.

En este capítulo se comienza por una descripción general del estado del arte en el tratamiento del riesgo desde la propia disciplina de la gestión de proyectos. Para ello se tratan las soluciones basadas en:

- proteger la programación de los proyectos frente a las variaciones en las condiciones de ejecución; son las denominadas estrategias *proactivas* orientadas a la *robustez*
- facilitar la recuperación de la programación tras una disrupción; se trata de las estrategias *predictivas-reactivas* orientadas a la *flexibilidad*
- disponer de políticas de secuenciación de actividades totalmente *rectivas*, es decir, que depachan (o detienen) las actividades según van apareciendo; una clase muy amplia de problemas relacionados con la organización del flujo de tareas en *problemas de taller* pero también de administración de comunicaciones, multiprocesado, etc.

Puesto que, como se ha indicado, los factores de riesgo pueden radicar en cuestiones más complejas que la propia incertidumbre de los componentes del proyecto, se consideran además una serie de propuestas que se han calificado de *alternativas*. Estas propuestas se remiten a cuestiones relacionadas con la posibilidad de aparición de *ciclos* e *iteraciones* y a la propia arquitectura de los procesos de ejecución de las tareas.

El capítulo concluye proponiendo el enfoque que debe emplearse para los dos problemas en estudio, el táctico y el operacional.

5.2. Las soluciones clásicas al problema del riesgo

5.2.1. Riesgos externos y riesgos internos

Los métodos de solución del problema RCPSP descritos en el capítulo anterior corresponden al caso determinista. Sin embargo la mayoría de los problemas reales son problemas en un entorno de *incertidumbre*. Elmaghraby [Elm05] sostiene que la incertidumbre en el terreno de los

proyectos se arraiga en *dos dominios* diferentes. En primer lugar está el riesgo *externo* a las actividades en casos tales como las condiciones climatológicas, el absentismo laboral o el fallo de la maquinaria en un proyecto de construcción. El segundo dominio es el del riesgo *interno*, el derivado del desconocimiento de todos los extremos concernientes a las actividades a realizar y, en particular, el problema de estimar acertadamente el contenido de trabajo, *el esfuerzo*, que conllevan. Según este autor lo tradicional ha sido prever para el primer tipo de contingencias dando por supuesto que las estimaciones que se hacen son suficientemente exactas. Sin embargo en muchos proyectos, especialmente con aquellos de elevado contenido de trabajo intelectual creador o complejo (tales como los proyectos de I+D, de desarrollo de nuevos productos, o, nuestro caso, el desarrollo de *software*, son los factores internos los que desempeñan el papel dominante.

La manera de afrontar comúnmente este tipo de proyectos intenta estimar por intervalos el esfuerzo requerido por las diferentes tareas (entre x e y hombres-mes, p.e.). Si no se dispone de mejor información se puede adoptar el criterio de que el esfuerzo así estimado es una variable aleatoria que se distribuye uniformemente en ese intervalo.

Con esta incertidumbre por delante, el responsable del proyecto aún tiene que decidir sobre los recursos que deberá destinar al proyecto. *La duración del proyecto acaba siendo pues no la fuente de la incertidumbre sino la consecuencia de las decisiones de asignación.* Este punto de vista determina una primera manera de abordar el proyecto en condiciones de riesgo: *se trata de decidir la asignación de recursos a las tareas que asegura el cumplimiento de los plazos, preferiblemente a coste mínimo.*

5.2.2. Tipos de estrategia para hacer frente al riesgo de los proyectos

El reconocimiento de que la incertidumbre se encuentra en el corazón de la propia planificación ha dado lugar a diferentes líneas de investigación en el terreno de la gestión de proyectos en condiciones de incertidumbre. Básicamente existen dos enfoques [HL04b]:

- aquellos en los que no se cuenta con una programación inicial sino

que las actividades se van secuenciando conforme van "apareciendo"

- aquellos en los que, partiendo de una programación inicial, se plantea una estrategia específica para tener en cuenta la incertidubre. Tradicionalmente en este segundo caso las estratategias han sido de dos tipos: ıtemize
- estrategias *proactivas*
- estrategias *predictivas-reactivas*

Una estrategia *proactiva* implica el desarrollo de un programa base, o inicial, *baseline schedule*, que esté protegido lo más posible frente a las interferencias y disrupciones que puedan producirse durante la ejecución del proyecto. Por el contrario, una estrategia *reactiva* se basa en la revisión, y eventual reprogramación, del proyecto cuando un evento altera la secuencia o duración inicialmente previstas. Una estrategia *predictiva-reactiva* es una estrategia que programa el proyecto previendo la posibilidad de que tenga que ser reprogramada.
En el caso en que no se cuenta con programa base nos encontramos ante *un caso extremo de estrategia reactiva*, pues en la práctica equivale a reprogramar -implícita y parcialmente - el proyecto ante la aparición de cada nueva actividad. Por analogía con los problemas de taller, a esta práctica se le llama una regla o política de despacho, *dispatching rule*.

Un concepto fundamental en el contexto en el que se cuenta con un programa base es el de *secuencia crítica* [WL77] o *cadena crítica* [Gol97] que de algún modo generaliza el concepto de camino crítico - *critical path* - propio de los modelos tradicionales PERT/CPM. Se trata de una secuencia de actividades interrelacionadas, bien por relaciones de precedencia bien por relaciones impuestas por la asignación los recursos, de tal manera que la misma determina la duración del proyecto total. De este modo se habla de la *criticalidad* de una tarea como la probabilidad de que pertenezca a una cadena crítica. La programación óptima de un proyecto restringido por los recursos calculada en condiciones deterministas puede resultar manifiestamente subóptima en la ejecución real a consecuencia de la existencia de este tipo de secuencias críticas. En tal caso se dice que es una solución *poco robusta*.

La *robustez*, también denominada *estabilidad* de una solución es una medida de la insensibilidad del tiempo de comienzo de las actividades a las variaciones de los datos de entrada. El término robustez es empleado en dos contextos: *robustez en el espacio de las soluciones* y *robustez en el espacio de los objetivos*. En el segundo caso se le denomina también *robustez cualitativa*. Una estrategia proactiva persigue construir soluciones robustas.

Un concepto próximo al de robustez es el de *flexibilidad* de las soluciones. Una solución es flexible si puede ser *reparada con facilidad* sin merma de su calidad en términos de los objetivos [gdf02]. Cuando se emplea una estrategia predictiva-reactiva, la flexibilidad de la solución obtenida es la característica más deseable.

En función de lo descrito hasta aquí existen diversas metodologías para enfocar el problema que se describen en los apartados que siguen a continuación.

5.2.3. Enfoques proactivos

El enfoque proactivo se basa en generar una programación base que incorpore un grado de anticipación suficiente para hacer frente a la variabilidad que pueda surgir en la ejecución del proyecto. Es el campo de interés de muchos esfuerzos de investigación tanto en el terreno académico como en el profesional.

La Cadena Crítica

La programación basada en la cadena crítica es la aplicación de la teoría de las restricciones - *theory of constraints*- de Goldratt [Gol97] a la gestión de proyectos es una de las líneas más conocidas. También conocida como *buffer management*, consiste en síntesis en aplicar los siguentes pasos:

- Construir una programación base *agresiva*, es decir, sin conceder holgura ni margen a la duración de las actividades, empleando la media o la mediana de dicha duración
- No tener en cuenta ni hitos intermedios ni fechas-límite

- No programar en modo multitarea, es decir, no permitir que un recurso se dedique a más de una actividad a la vez
- Programar las actividades del camino crítico ajustadas a la derecha, es decir, a partir de la fecha más tardía de inicio
- Minimizar el trabajo en curso, WIP, *work in progress*, desplazando a la izquierda las actividiades si es preciso para eliminar los conflictos sobre recursos
- Identificar la *cadena crítica* como la secuencia de actividades que determina la duración del proyecto
- Colocar al final de la cadena crítica la reserva de tiempo que no fue considerada a la hora de construir la programación, esa reserva de tiempo se denomina el *buffer* del proyecto y se suele fijar en el 50 % de la duración total
- Incluir un *buffer* de alimentación, *feeding buffer*, en el enlace entre actividades no críticas y la cadena crítica, también calculado normalmente como el 50 % de la duración de la secuencia que "desemboca" en la cadena crítica
- Por último, incluir un *buffer* de recursos, *resource buffer*, en cada nodo de la cadena crítica en la que se produzca un cambio en los recursos empleados; este servirá a modo de aviso

Tras esta programación base se construye una *programación de proyecto* basada en el inicio más temprano posible, ajustando a la izquierda, y sin considerar los *buffers*. La implementación del procedimiento consiste en desarrollar la programación de proyecto utilizando la programación base como referencia para tomar medidas correctivas. De abí la denominación de *buffer management*. En [HLD02] se analiza críticamente esta metodología señalando sus ventajas e inconvenientes.

En un *entorno multiproyecto* la metodología que se aplica, basada en los principios generales de la teoría de las restricciones, es la siguiente:

a) Priorizar los proyectos de la organización

b) Programar cada proyecto de acuerdo con el procedimiento antes indicado en función del recurso más restrictivo, el "cuello de botella"

c) Escalonar los proyectos de acuerdo con la prioridades y el recurso más restrictivo

d) Insertar *tambores*, reservas o *buffers* del recurso más restrictivo entre proyectos

De esta forma la gestión, y de manera análoga al caso de un sólo proyecto, se convierte en una gestión de los diferentes *buffers*, de tiempo y de recursos. En la literatura hay diversas discusiones sobre la idoneidad o no de concentrar las holguras mediante esta política de *buffers*. Veáse, por ejemplo, [dVDHL05].

Generación de secuencias robustas

Al margen del procedimiento de la cadena crítica, el más conocido en medios profesionales, existen otros procedimientos que se describen en la literatura que se basan en el concepto de *estabilidad*. Cuando la restricción de recursos no opera rígidamente, es decir, cuando se pude recurrir de forma extraoridnaria a recursos adicionales en caso de una disrupción de la programación base, la estabilidad es el objetivo deseable con el fin de reducir la excepcionalidad.

La medida de la estabilidad puede plantearse de dos maneras. Una de ellas es la *estabilidad promedio*, medida como la esperanza matemática de la suma ponderada de las desviaciones del tiempo de comienzo de las actividades cuando se produce una variación en el plazo de inicio de una de ellas. El problema de minimización que se plantea se pude abordar de varias formas diferentes:

- Como un problema de flujo a coste mínimo [HL04a]
- Como un problema de programación lineal paramétrica [TFC98]

En algunos casos, lo que puede interesar es asegurar la estabilidad en *el peor de los casos*, es decir, asegurar la *calidad* de la programación

mediante una cota superior a la máxima disrupción. Este es el procedimiento que se describe en [gdf02].

En la situación citada, se puede recurrir a recursos de forma extraordinaria y el problema siempre se resuelve resecuenciando a partir de la alteración a la fecha más temprana. Cuando eso no es posible se comprueba la existencia de soluciones factibles una vez producida la disrupción utlizando un modelo análogo al empleado en la programación de la producción *just in time* [LN94].

Generación de secuencias con lógica borrosa

Una tercera aproximacion al problema de generar secuencias robustas es el de modelar el problema como un problema "fuzzy". AUnque aparentemente semejante, la idea que hay detrás de la aplicación de la lógica borrosa no es que las duraciones sean variables aleatorias, sino que la incertidumbre es *epistémica*: no se conoce el contenido de trabajo con precisión suficiente. La *función de pertenencia* de Zadeh se emplea tanto para modelar aquellas variables que no se conocen con precision como aquellas otras que son imprecisas en si mismas, como la productividad, por ejemplo. Los métodos del álgebra "fuzzy" permiten tratar los problemas generalizando muchos de los métodos diseñados para los problemas *crisp*.

En [LO] se presenta un modelo que relaciona el método de la cadena crítica con la lógica borrosa, utilizando esta para determinar el *buffer* de proyecto. En [KYY$^+$05] los autores proponen un algoritmo genético híbrido con un controlador de lógica borrosa para el problema multiproyecto - RCMPSP. En [San00] se presenta un modelo de apoyo a la decisisón - DSS - que emplea la lógica borrosa para asignar recursos, en este caso ingenieros, a tareas con diferentes niveles de complejidad y duración "fuzzy", en función de la habilidad de los recursos y la complejidad de las tareas, también modeladas como variables borrosas.En [zA01b] se propone un modelo parecido.

5.2.4. Enfoques predictivos-reactivos

Los enfoques predictivos-reactivos, al contrario que los anteriores, se basan en la modificación de la programación base como respuesta, reacción, a las disrupciones. Hay diferentes estrategias que van desde la más simple, la *reparación de secuencias*, hasta la *reprogramación completa* del proyecto a la vista de la disrupción.

Reparación de secuencias

La reparación de secuencias en su modalidad más simple se basa exclusivamente en el desplazamiento hacia la derecha de las actividades que se ven afectadas por las disrupciónes, bien por verse afectadas por el recurso que ha fallado o bien porque están condicionadas por las relaciones de precedencia. Es una estrategia reactiva que genera resultados pobres en general.

En [AR00] los autores plantean el problema de reprogramar un proyecto cuando se produce una disrupción en un modelo multimodo a partir del problema de insertar óptimamente una nueva actividad inesperada a partir de un determinado momento del tiempo, de manera que el tiempo total de compleción varíe lo menos posible. Utilizando una representación en forma de grafos de recursos, generan cortes en el grafo que posibilitan seleccionar la mejor inserción empleando un sistema de generación de secuencias en serie - SGS - y una regla MINSLACK. Posteriormente en [AMR03] mejoran la solución con un procedimiento de búsqueda tabú.

Reprogramación

La reprogramación o *regeneración* de secuencias, como se denomina en el ámbito de los problemas de taller, emplea una medida determinista de los objetivos, típicamente el tiempo hasta la compleción, para reconstruir una nueva programación a partir de la situación de hecho en la que se produce la disrupción. En cierto sentido, la reparación de secuencias que se ha comentado en el punto anterior es como un *paso heurístico* en un algoritmo de reprogramación.

Dado que en este caso la programación resultante puede diferir extraordinariamente de la solución inicial, el problema se plantea como uno de *inestabilidad*. En el contexto de una organización multi-proyectos, la estabilidad es necesaria para eludir el problema que en el campo de los problemas de taller se denomina el *nerviosismo de la secuencia*. De ahí que en la práctica sea preferible muchas veces reparar la programación antes que alterarla completamente.

Es posible modelar explícitamente este objetivo de prevenir la inestabilidad buscando una *estrategia de mínima perturbación*. Para ello se puede minimizar bien la diferencia entre las fechas de inicio previstas inicialmente y las reprogramadas o el número de actividades a realizar por diferentes unidades de recursos. Cabe distinguir además dos momentos en la reprogramación que dan lugar a situaciones distintas. En unos casos se trata de reasignar una fecha de inicio a actividades que no han comenzado. En otros hay que reasignar recursos y tareas sobre la marcha, una vez iniciadas las actividades.

Una metodología empleada para enfrentar este problema se basa en la programación por metas - *goal programming* - aplicada al problema MRCPSP. Se asignan metas a clases de actividades agrupadas por prioridad. Una de las metas es minimizar el número de actividades que se modifican. El problema se resuelve aplicando la búsqueda tabú [CDM$^+$02].

Secuenciación contingente

Puede ocurrir que sea la propia dirección del proyecto la que realice cambios en la programación del mismo en el tiempo de su ejecución. En este caso, el problema interesante es presentar los cambios posibles con menor impacto en el resultado final en términs de duración. Una forma de abordarlo es la de generar para cada recurso una *secuencia de grupo*, es decir, un conjunto de tareas total o parcialmente ordenadas vinculadas a ese recurso. A partir de cada secuencia de grupo se pueden construir programciones del proyecto total de modo que la dirección del proyecto no cuenta con un único programa posible sino con varios[BDR02].

Aceleración de actividades - *activity crashing*

Una alternativa para tomar acciones correctivas en la ejecución del proyecto es la de acelerar la misma concentrando recursos en las actividades que van retrasadas. Este problama ha sido ampliamente estudiado en la literatura como el problema del *trade-off* tiempo-coste. En [DH02] está ampliamente tratado. En el contexto de los proyectos de sistemas de información, y en otros, se trata ampliamente las paradojas que su implementación en la práctica pueden derivarse en el conocido problema del *mito del hombre-mes*.

5.2.5. Enfoques totalmente reactivos

Políticas dinámicas de secuenciación

Los enfoques totalmente reactivos son el campo del problema SRCPSP - *stochastic resource constrained project scheduling problem*. No se parte de una programación base, sino que se plantea como un *problema de decisión multietapa* que utiliza *reglas*, *políticas* o *estrategias de secuenciación* para decidir sobre la asignación de recursos a las actividades en el momento del inicio del proyecto y a la conclusión de cada actividad en base al pasado observado y al conocimiento a priori de que se dispone.

El SRCPSP no es más que una clase de la denominada programación estocástica, *stochastic scheduling*. El objetivo comúnmente considerado, la minimización del tiempo de compleción del proyecto para una clase de reglas o estrategias de secuenciación, es un caso particular del problema más general de asignar recursos a tareas con características aleatorias que demandan dichos recursos.

Teóricamente estos problemas pueden formularse en términos de *programación dinámica*, pero la aplicación directa de esta técnica resulta muy limitada en términos de efectividad debido a la denominada *maldición de la dimensionalidad*, termino acuñado por Richard Bellman, el "padre" de la programación dinámica [Bel72]. En la práctica eso hace que los modelos disponibles se hayan desarrollado para problemas muy específicos y que los resultados obtenidos no puedan geenralizarse con facilidad a otros dominios, salvo algunas consideraciones muy generales,

según Niño-Moura [NM01]. Según este autor los modelos más estudiados pueden clasificarse en tres grandes categorías que han evolucionado autónomamente. Estas serían: las aplicaciones para la *secuenciación de tareas de duración estocástica*, los denominados *multi-armed bandit problems* y los *modelos generalizados de la Teoría de Colas*. En cada una de las áreas se observa la aparicicón de resultados en tres etapas:

- Un primer grupo de resultados en los que se definen un conjunto de *reglas de índices de prioridad* sencillas y fáciles de computar que se demuestra que permiten ofrecer resultados óptimos para problemas relativamente simples. La característica principal de esas reglas es la de que su valor para cada tarea depende de la propia tarea y su clase y no del resto.
- Un segundo grupo de resultados se ha alcanzado intentando generalizar esas políticas sencillas a casos más generales introduciendo supuestos técnicos adicionales y verificando la optimalidad en ese nuevo contexto.
- El tercer tipo de resultados se refiere a modelos más complejos para los que la optimización no parece estar al alcance de procedimientos y reglas sencillas. En estos casos los investigadores persiguen diseñar heurísticos que sean relativamente fáciles de implementar y que, generalmente, se basan en las reglas de los modelos más simples. La investigación del grado de (sub)optimalidad de estos heurísticos se realiza o bien de modo analítico o bien, de manera más corriente, mediante simulación.

Modelos para secuenciar lotes de actividades de duración estocástica

En esta clase de modelos, un lote de n tareas de duración aleatoria deben procesarse por un conjunto de m máquinas para optimizar alguna variable relativa a los tiempos.

El modelo más simple es el que secuencia n tareas en una máquina. Cada tarea tiene un duración aleatoria e independiente de las demás. Las políticas admisibles no pueden ser ni *anticipativas*, es decir, no conocen a priori la duración de las tareas pendientes de completar, ni

permiten la *realización parcial*, es decir, una tarea que se inicia debe completarse. Si se incurre en un coste w_i por cada unidad de tiempo que transcurre hasta que la tarea i es completada y C_i es el tiempo en el que dicha tarea está completa, el problema consiste en minimizar la expresión:

$$w_1 E_\Pi [C_1] + w_2 E_\Pi [C_2] + \cdots + w_n E_\Pi [C_n] \tag{5.1}$$

donde Π es el espacio de las políticas admisibles y $E_\Pi [C_i]$ es el valor esperado del tiempo de terminación de la tarea i.

La solución óptima en el caso determinista es la regla denominada *tiempo ponderado de proceso menor*. Esta regla secuencia las tareas por tanto en orden creciente de $w_i p_i$ con p_i, la media de la duración de la tarea i. Se ha demosrtado que esta misma regla se puede generalizar al caso no determinista.

Si se permite interrumpir una tarea para retomarla después, se demuestra igualmente que la mejor política es un índice de prioridad que tiene en cuenta la cantidad de trabajo que cada tarea tiene acumulado previamente. Si con lo que se cuenta es con un grupo de máquinas idénticas en número mayor que 1, podemos plantear el problema de dos formas:

- La minimización del tiempo total de permanencia de las tareas, *flowtime*
- La minimización del plazo de finalización de la última tarea, *makespan*

Para el primer caso, cuando la duración de las tareas se puede *ordenar estocásticamente* la regla que asigna la mayor prioridad a la tarea con menor valor esperado de tiempo de procesado resulta óptima. En el segundo caso, la regla óptima es la que asigna el mayor índice de prioridad a la tarea que tiene mayor valor esperado.

Conforme el modelo se hace menos restrictivo, las reglas simples dejan de ser óptimas y el modelo se vuelve computacionalmente intratable. Sin embargo un interesante resultado muestra que el empleo de las reglas antes citadas es una cota superior del problema con independencia del número de tareas. Esto permite concluir que elevando este número, la distancia entre el óptimo y la cota se reducen produciéndose así un fenómeno de *optimalidad asintótica*.

Modelos de asignación de recursos a tareas. El problema *Multi-armed Bandit*

Los modelos de esta clase se refieren la asignación óptima de esfuerzo a un conjunto de tareas que cambian de estado de manera aleatoria dependiendo si están siendo atendidas o no. El ejemplo clásico del problema *multi-armed bandit* en tiempo discreto es el siguiente:

Hay un conjunto de N proyectos, a los que se debe atender. A cada intervalo de tiempo sólo un proyecto puede ser atendido. Un proyecto n puede estar en cualquiera de un número finito de estados $j \in J_n$. Si el proyecto n es atendido en el momento t se recibe una recompensa R_j que es descontada en el tiempo por un factor $0 < \beta < 1$. Cuando un proyecto es atendido cambia del estado i al estado j con probabilidad p_{ij}. El problema consiste en seleccionar la estrategia que maximiza el valor total descontado en el tiempo de las recompensas.

$$\max_{\pi \in \Pi} E_\pi \left[\sum_{t=0}^{\infty} \beta^t R_{j(t)} \right] \tag{5.2}$$

Donde π es una política del conjunto de todas las políticas posibles, Π.

Este problema se resuelve con el denominado *índice de prioridad de Gittins* que se calcula para cada estado posible de un proyecto. Se selecciona el proyecto con el mayor índice.

Para modelos más complejos el índice de Gittins no asegura el óptimo. En particular, cuando se puden abordar varios proyectos al mismo tiempo y los proyectos no tratados tambiñne pueden cambiar de estado da lugar al problema denominado *restless bandit*. Este problema se ha intentado resolver por programación lineal relajando alguna de las restricciones. Está asintóticamente limitado por el modelo básico, lo que permite utilizar el criterio de Gittins para explorar a partir de una primera solución acotada [NM04].

Modelos generalizados de colas

En los modelos generalizados de colas, las tareas llegan aleatoriamente, a diferencia de las dos clases anteriores donde se conocen de antemano. El modelo más simple es el siguiente:

N clases de tareas diferentes deben ser atendidas por un servidor. Las tareas de la clase $j$llegan según una distribución de Poisson a tasa α_j. El tiempo de procesado promedio es una varable aleatoria de media $\frac{1}{\mu_j}$. El coste por unidad de tiempo que una tarea permanece en el sistema, bien esperando bien procesándose es c_j. El objetivo es encontrar una política *no antcipativa* y *sin detención parcial* que minimice el coste total. Al igual que en el caso anterior, π es una política del conjunto de políticas admisibles, Π. Sea $E_\pi[L_j]$ el número esperado de tareas de la clase j que permanecen en el sistema en *régimen permanente* bajo la política π. La función objetivo será:

$$\min_{\pi \in \Pi} c_1 E_\pi[L_1] + c_2 E_\pi[L_2] + \ldots + c_N E_\pi[L_N] \tag{5.3}$$

Para este modelo tan simple se demuestra que la política óptima es la denominada regla $C\mu$, que otorga la mayor prioridad a la clase que da el máximo valor a $c_j \mu_j$ [NM01].

Esta misma regla se verifica si se puede suspender temporalmente y retomar después el procesado de una tarea, siempre que la duración de la misma siga una distribución de probabilidad exponencial. Para problemas más complejos se han evaluado estas reglas por medio del denominado método de *regiones alcanzables* que pretende caracterizar el espacio de estados en el régimen permanente en términos de la longitud media de las colas.

Para condiciones menos restrictivas (diversos procesadores de características diferentes, vinculación entre tareas) el problema se vuelve intratable analíticamente. A esta familia pertenecen los problemas de secuenciación *on line* en los que se busca una regla simple de despacho de tareas y caracterizar su mayor o menor proximidad al óptimo. Existen varios tipos de problemas diferentes:

- Problemas de secuenciación *on line* **a tiempo**. Se trata de decidir a cada momento de tiempo t que tarea iniciar. Básicamente se diferencian dos casos, el caso de "clarividencia" cuando se conoce la carga de trabajo de las tareas, como ocurre con un servidor *web* de páginas estáticas, o sin ella, como ocurre con un sistema operativo cuando comienza un proceso. En estos casos es evidente que si no es posible supender la ejecución de una tarea para aco-

meter otra que acaba de llegar, no es posible una secuenciación óptima.

- Problemas de secuenciación **a lista**. Se trata de colocar en la "cola" tareas pendientes cada tarea cuando llega. Una vez ubicada en ella, ya no puede moverse. Al igual que en el caso anterior se diferencia entre la posiblidad de suspender o no la ejecución de una tarea.

5.3. Aproximaciones alternativas al problema del riesgo y la incertidumbre en los proyectos

5.3.1. La iteración en las redes de los proyectos, la arquitectura de los procesos y el riesgo en la estructura de descomposición en tareas

Las aproximaciones al problema del riesgo en los proyectos, la *variabilidad*, a la que antes se ha hecho referencia fundamentalmente parten de la idea de que la incertidumbre proviene del desconocimiento del contenido en trabajo de las actividades de los proyectos. Se trata, por lo tanto, de *acotar* las consecuencias de ese desconocimiento en función de las posibilidades que la dirección de proyecto tiene de acotar su propio desconocimiento.

Las estrategias proactivas afirman la posibilidad de hacerlo, bien a través del juego de *buffers* del método de la cadena crítica, bien diseñando programaciones *estables* que absorban con poco coste la demanda de recursos extraordinarios para recuperar una programación que ha sido perturbada externamente. Las estrategias predicitivas-reactivas reprograman, parcial o totalmente el proyecto en respuesta a una de esas perturbaciones. Las estrategias puramente reactivas renuncian a una programación base a priori e intentan, estudiando las características fundamentalmente estocásticas de las eventuales perturbaciones, encontrar reglas de prioridad robustas, que en este caso son más bien reglas de despacho de tareas.

Pero la complejidad de las interrelaciones entre las actividades en los proyectos, especialmente en los dominios en los que la ejecución cualitativa de las actividades afecta a todas sus sucesoras y la propia gestión del proyecto influye sobre esa calidad, exije dar un paso más para abordar el problema del riesgo. La manera más evidente en la que se produce esa complejidad es la que se deriva de la existencia de *iteraciones* en la ejecución del proyecto que *desmienten el carácter acíclico del grafo* con el que se modelan e invalidan, así, gran parte de las soluciones hasta ahora presentadas, salvo, lógicamente las puramente reactivas.

La segunda fuente de distorsión de los modelos de redes covencionales es la existencia de eventos que pueden dar lugar a bifurcaciones en la propia red. Según sean los resultados de una actividad, las siguientes deberán de hacerse de un modo u otro. A veces surgirán actividades nuevas no previstas inicialmente. Evidentemente, un programa incial puede contemplar a priori todos esos posibles cambios en la topología de la red, pero a costa de aumentar extraodinariamente la complejidad de la misma y sin que sea evidente como deben tratarse actividades que a lo mejor deben hacerse o a lo mejor no con los métodos convencionales.

Básicamente se trata de incorporar al flujo natural de las actividades un flujo complementario de *información*. La decisión de "volver atrás" en la red para corregir errores o defectos en una actividad, o la de "optar" por un arco u otro, se produce a partir de información sobre el propio proceso de ejecución del proyecto o sobre cambios sobreveenidos en su entorno. No es por tanto sólo un problema de desconocimiento o aleatoriedad en las variables que rijen la ejecución sino una complejidad adicional que hace que la propia ejecución de las tareas modifique la estructura de descomposición del proyecto en tareas.

Desde el propio campo de la gestión de proyectos y desde otros campos se han hecho contribuciones a este problema que denominaremos el *riesgo* (derivado de la complejidad) *de la estructura de descomposición en tareas* - WBS.

5.3.2. Redes estocásticas. El modelo GERT

El modelo original PERT ya incluía una aproximación muy elemental al problema de la variabilidad en la duración de las actividades carac-

terzándolas por tres estimaciones; "pesimista","optimista" y "normal". El GERT, *graphical evaluation reviewe technique,* es una (de entre muchas otras como el VERT, el GAPS, ...) extensión del modelo que tiene en cuenta la existencia de diferentes estados posibles en un sistema y de las interrrelaciones entre las diversas variables de estado, es decir, la información que los diferentes componentes del sistema intercambian. Para ello emplea una representación denominada *red estocástica*.

Una red estocástica modelada por el GERT es una representación de actividades en arcos -AEA. El grafo dirigido dirigido está formado por nodos lógicos y arcos. Los nodos representan hitos o eventos. Los arcos transmiten información entre un nodo y sus sucesores determinando el modo de ejecución de las operaciones asociadas a partir de el estado del predecesor, sus propiedades y las de los sucesores.

Los nodos cuentan con entrada y salida. Las entradas pueden ser:

- *EXCLUSIVE-OR (XOR).* Sólo una de las ramas de entrada puede realizarse para que el nodo se realice.
- *INCLUSIVE-OR (OR).* Cualquier rama que se realice provoca la realización del nodo. El tiempo para la realización es el menor de las ramas que se realizan y llegan al nodo.
- AND. Todas las ramas deben realizarse para que el nodo se realice. El tiempo para la realización es el mayor de todas las entradas.

Las salidas pueden ser:

- *DETERMINISTAS.* Todas las ramas que salen se realizan.
- *PROBABILÍSTICAS.* Sólo una de las ramas que sale se realizará con una cierta probabilidad $0 \leq p \leq 1$.

La *transmitancia* es la relación que existe entre dos nodos consecutivos., o dicho de otra forma, la información que se transmite de un nodo a otro a través de un arco dirigido. La transmitancia puede tener varias dimensiones, unas son aditivas (tiempo, coste, ...) y otras multiplicativas (probabilidades).

Potencial de aplicación del modelo GERT

El modelo descrito permite una *representación* más sintética y rica del problema de la programación de proyectos que las redes clásicas de AEN y AEA. Aparte de ello, se ha empleado para tratar analíticamente los problemas y para la simulación de los mismos. Para ello se representa un proyecto con distintos modos de ejecución y duración de las actividades aleatoria como una una red estocástica formada únicamente por nodos XOR deterministas.

Una actividad definida y realizada de un modo determinado entre dos eventos es un arco dirigido. La transmitancia asociada a ese arco tiene dos parámetros asociados: la probabilidad de que sea ese el modo elegido de realizar la actividad y la duración o el coste (que a su vez es una variable aleatoria).

Tratamiento analítico Por la teoría de grafos de flujo se sabe reducir topológicamente estructuras complejas a estructuras simples: *serie*, *paralelo* y *bucles* de primer orden. Para poder hacer la reducción es necesario que las transmitancias sean multiplicativas y no aditivas, por lo que los factores como tiempo y coste deben sustituirse por otros multiplicativos. Para ello se emplea la *función generatirz de momentos*, partiendo de la base de que las diferentes duraciones (y costes) asociadas a las actividades son variables aleatorias independientes. En tal caso la función generatriz de momentos de la suma de dichas variables es el producto de las funciones generatrices de cada una de ellas.

De este modo, en un GERT que contenga únicamente nodos con entrada XOR atribuyendo a cada actividad (arco) el producto de su probabilidad por la función generatriz de momentos de su duración (coste) podemos resolver analíticamente la red obteniendo la función generatriz de momentos del nodo final y, por tanto, caracterizar la probabilidad de completar el proyecto y calcular la fecha de terminación (o el coste) del mismo como una variable aleatoria.

Simulación Cuando las estructura incluyen nodos AND e INCLUSIVE-OR, la solución analítica no es posible a priori y se requieren procedimientos con un elevado coste computacional. Cuando las redes son

complejas, aunque todos los nodos sean EXCLUSIVE-OR, la reducción topológica también es excesivamente gravosa. Por ello se han desarrollado métodos basados en la simulación de redes estocásticas para el tratamiento de estos problemas.

Aplicaciones detectadas en la literatura

La mayor parte de las referencias recientes consultadas emplean estos modelos a efectos de representación y simulación como las que se recogen en el cuadro 5.1, sin que se pretendan explotar sus posibilidades analíticas. Entre otros motivos porque otros formalismos, como el de la Redes de Petri que se expone a continuación, son más potentes tanto analítica como computacionalmente.

Referencia	**Descripción**
[Kur06]	Simulación de Monte Carlo con disyunciones
[Gav04]	GERT con lógica "fuzzy"
[GGGL03]	Disyunciones con el problema "knapsack"
[A.P03]	Transformación y simplificación de la red
[KN02]	Simulación de Monte Carlo
[Nag02]	GERT y algoritmos genéticos

Cuadro 5.1: Algunas aplicaciones de los modelos GERT en el dominio de la gestión de proyectos

5.3.3. Las Redes de Petri

Las Redes de Petri fueron introducidas por C.A. Petri in 1962. Su éxito se debe básicamente a la simplicidad de su mecanismo básico, si bien, la representación de grandes sistemas es costosa. Básicamente es un modelo de representación de un sistema caracterizado por un conjunto

de estados y sus regles de transición asociadas. Numerosos autores han extendido el modelo básico introduciendo diversos refinamientos.

Una red de Petri es formalmente un tipo particular de grafo dirigido asociado a un estado inicial denominado el marcado inicial, M_0. Se trata de un grafo dirigido y con pesos en el que hay dos tipos de nodos, llamados *lugares* y *transiciones*. Los arcos discurren entre nodos de dos tipos diferentes. Lo lugares representan típicamente acciones o condiciones y las transiciones, eventos. Las fichas, *tokens*, residen en los lugares y representan el estado lógico verdadero de la condición de una acción asociada al lugar correspondiente. Un lugar que contiene al menos una ficha se denomina lugar *marcado*.

Un *marcado* es un conjunto que contiene en cada una de sus posiciones el número de fichas de cada lugar. Las fichas se mueven por la red como consecuencia del disparo de las transiciones. Una transicicón se activa cuando toso los lugares que conducen a ella están marcados con, al menos, una ficha. Si una transición activa se dispara, se retira una marca de todos los lugares predecesores y se añade una a todos los lugares sucesores de esa transición. Un estado del sistema , un marcado, M' es *alcanzable* desde el estado M si existe una secuencia válida de transiciones de M' a M.

Una red de Petri *coloreada* asocia conjuntos de valores a las fichas, de modo que las transiciones reevaluan el "color" de las fichas. Una red de Petri *temporizada* tiene tiempos asociados a los lugares y/o a las transiciones. En la redes de Petri temporizadas, las transiciones para activarse requieren haber agotado su tiempo o que lo haya sido en los lugares predecesores. Una red de Petri *estocástica* atrbiuye una función distribución de probablidad al tiempo de disparo de las transiciones. Las redes de Petri estocásticas coloreadas se han empleado para el modelado de los problemas RCPSP, normalmente asignando lugares al inicio y final de una actividad y transiciones al inicio y final de las mismas. Los colores se emplean para representar el tipo de recursos y su disponibilidad (aunque en otros casos esto se representa por lugares).

Potencial de aplicación de las redes de Petri

Las redes de Petri, en particular en las versiones de alto nivel (HLPN) comentadas: temporizadas, coloreadas, estocásticas, ... son un formalismo de gran potencia para la descripción de un sistema. Como en el caso anterior una aplicación inmediata es la representación de los problemas. Pero posee también un potencial importante para el análisis y para la simulación.

Tratamiento analítico El principal tratamiento analítico es el *análisis del espacio de estados* en particular en términos de verificar la *alcanzabilidad* de determinadas regiones. Sin embargo, para cualquier problema mínimamente relevante el espacio de estados alcanza dimensiones enormes. Para resolver el problema se han desarrollado herramientas muy potentes para reducir este espacio por agregación o condensación de subespacios, análisis de invariantes de lugar, etc. Los algoritmos de exploración del espacio de estados son semejantes a los algoritmos *branch and bound* ya comentados.

Simulación El formalismo de las redes de Petri y su implementación hace que estas sean *ejecutables* desde su especificación, de ahí que se empleen naturalmente para la simulación. El Departamento de Computación de la Universidad de Aarhus en Dinamarca ha desarrollado un paquete denominado *Design/CPN* que puede descargarse de la url `http://www.daimi.aau.dk/CPnets/`.

Aplicaciones detectadas en la literatura

Las redes de Petri han venido a sustituir a todas las representaciones en forma de grafos mejorados previamente existentes y hay abundantes referencias en la literatura respecto a su empleo como herramientas para el tratamiento de problemas de programación de proyectos. Algunas de las consultadas son las que aparecen en el cuadro 5.2.

Referencia	Descripción
[HS08]	Desarrollo de productos
[HBH+06]	Aseguramiento de calidad basado en el valor
[KHY06]	Predicción y reparación de secuencias en entorno multiproyecto
[KWDK06]	Gestión de carteras multiproyecto basada en análisis de costes ABC
[KC06]	Desarrollo de productos
[dSP05]	Desarrollo de software basado en componentes
[ZnRF04]	Evaluación y control de riesgos plazo/coste
[RKC01]	MRCPSP combinado con algoritmo genético
[JKCR00]	Equilibrado de recursos
[Kum98]	Modelo básico de asignación de recursos
[EKR95]	Modelado de procesos work-flow

Cuadro 5.2: Algunas aplicaciones de las Redes de Petri en el dominio de la gestión de proyectos

5.3.4. Las *DSM*, una propuesta desde el dominio del desrrollo de producto

Las matrices de estructura de diseño, *design structure matrix*(ces), *DSM*, son un formalismo para representar el grafo dirigido de un proyecto que se emplea en el dominio del desarrollo de productos. Se trata de una matriz cuadrada con elementos binarios con m filas y columnas y n elementos no nulos, representando m tareas y n arcos indicativos de relaciones de precedencia.

La distribución de la matriz es la siguiente: los procesos o actividades se disponen como encabezamiento de las columnas y etiquetas de las filas. Si existe un arco del nodo (actividad) i al nodo (actividad) j, el elemento (j, i) vale 1. Los elementos de la diagonal principal de la matriz no tienen ningún significado por lo que se dejan normalmente en blanco.

La principal ventaja de este modo de representación sobre el grafo convencional es su compacidad y su capacidad de proporcionar una visión sistemática de las relaciones entre procesos o tareas más fácil de leer con independencia de su tamaño. El orden de presentación de las filas (y las columnas) es el de la secuencia de las actividades, por lo que los elementos de una fila cualquiera indican las tareas que deben haber terminado antes de iniciar la tarea que representa esa fila. De la misma forma, leyendo las columnas, cada fila indica las tareas que reciben información de la corresponiente a la columna en cuestión.

Los elementos no nulos (o *marcas*) que se encuentran por debajo de la diagonal principal indican información "hacia adelante". Los elementos que están por encima de la diagonal, por el contrario, representan procesos de *realimentación* según las cuales una tarea puede depender de sus sucesoras. Esas dependencias "hacia atrás" son la principal causa de iteraciones en proyectos complejos. En el caso de los proyectos de desarrollo *software* representan, por ejemplo, el clásico trabajo de corrección de errores en la codificación.

Particiones y desacoplamiento

La arquitectura de la descomposición en tareas - WBS - de un proyecto puede, hasta cierto punto, definirse por la dirección del mismo. Evitar al máximo las iteraciones equivale a diseñar una estructura en la que el número de elementos situados por encima de la diagonal sea mínimo. En proyectos complejos es poco probable que se puedan anular dichos elementos, pero minimizarlos y desplazarlos lo más próximo posible a la diagonal permite llegar a una estructura de proyecto más fiable y predecible [Bro02]. El proceso para conseguirlo se llama *particionar* la matriz.

Desacoplar la matriz es el proceso de identificación de aquellos elementos que, si se eliminaran, convertirían a la matriz en triangular. Identifcar estos elementos equivale a identificar los supuestos que dan lugar a las iteraciones en el proyecto, es decir, las tareas que están acopladas.

En[YB03] se exponen procedimientos algorítmicos para particionar y desacoplar las matrices.

DSM numéricas

En la notación binaria, las DSM contienen un único atributo relativo a cada par de actividades, si existe o no una relación entre el resultado de una y el de la otra. Las DSM *numéricas* pueden contener muchos atributos, lo cual enriquece la información sobre el proyecto. Así, por ejemplo, si la tarea B depende de información sobre la tarea A, pero esa información es predecible o tiene un impacto escaso el elemento puede ser eliminado. Con el fin de recoger esta mayor capacidad informativa, se proponen las matrices DSM numéricas en las que se incluyen nuevos atributos como:

Indicadores de Nivel: Reflejan el orden en el que las marcas "hacia atrás" pueden eliminarse en función de las posibilidades de estimar "a priori" la información que se genera en la actividad posterior.

Indicadores de Importancia: Una escala sencilla, entre 1 y 3, para indicar la importancia de la dependencia de la actividad sucesora en la actividad precedente.

Probabilidad de repetición: Una estimación de la probabilidad de que la actividad sucesora provoque una repetición de la antecesora; en este modelo, empleado entre otros en [Cho05], los elementos situados sobre la diagonal representan la probabilidad de una iteración hacia atrás mientras que los situados por debajo indican la probabilidad de una segunda iteración. Se han elaborado algoritmos de particionado que minimizan la duración esperada del proyecto.

Simulación de DSM

Una de las más extendidas aplicaciones de este modelo es la simulación para obtener un perfil duración/coste de un proyecto de desarrollo de producto. La varianza en la duración y el coste son atribuibles en gran medida a las iteraciones. Dichas iteraciones pueden o no ocurrir dependiendo del comportamiento de una serie de variables que el modelo simula como variables estocásticas cuya probabilidad de ocurrencia depende de la existencia de determinados paquetes de información que activan un proceso de reelaboración (*rework*. Un proceso de reelaboración puede, a su vez, desencadenar otros sucesivos afectando así al proyecto en su totalidad.

Este modelo se ha utilizado en solitario o en combinación con otras técnicas de las ya reseñadas para elaborar secuenciaciones robustas modificando la arquitectura de la descomposición en tareas de los proyectos. Así ocurre en [CE05], [ZY04] y [MYB07].

5.3.5. La analogía cuántica

Una original alternativa al estudio del problema del reisgo proviene de una*analogía* basada en la *mecánica cuántica* [FL07].

Se supone que la implementación de una actividad es análoga a un paquete de ondas que se propaga en dirección a un hito temporal.

Cuando las actividades son coherentes, el hito se alcanza. Cuando una actividad se retrasa la probabilidad de la ocurrencia del hito en un determinado momento queda afectada, como ocurre con la probabilidad de un sistema cuántico de partículas cuando se producen fenómenos de interferencia.

Esta aproximación se basa en en la estructura del proyecto y de los hitos más que en la información detallada sobre cada una de las actividades. Las acciones externas o las decisiones de la dirección pueden afectar a la realización de los hitos y esto se representa por medio de dos parámetros internos, cada uno de los cuales toma un valor específico en cada uno de los hitos. La idea básica es que la actividad humana no puede definirse con precisión absoluta sino que más bien, cierto principio de incertidumbre gobierna las actividades y la productividad del trabajo. A partir de esto las tareas del proyecto e hitos son modelados mediante *funciones de onda* con su correspondiente *amplitud de probabilidad* ψ_i. La amplitud de probabilidad de un hito se representa como la superposición de las amplitudes de probabilidad de las actividades que *interfieren* en dicho hito:

$$\Psi_H = \psi_1 + \psi_2 + \psi_3 + \dots \tag{5.4}$$

Donde 1, 2, 3, ... son las actividades que concurren en el hito. La amplitud de probabilidad de cada tarea se define de manera cuasi-clásica siguiendo a Feynman [FSLS65] como:

$$\psi \approx \left(\sqrt{\mathrm{K}}\right)^{-1} \exp\left(\int \mathrm{K} dx\right) \tag{5.5}$$

El vector fundamental de una tarea se define como $\mathrm{K}_0 = 2\pi/D$, donde D es la duración de la tarea. En L, $\mathrm{K}_L = \epsilon \mathrm{K}_0$ donde $\epsilon = L/nD$. Si $L < D$, se toma $n = 1$. De lo contrario se toma $n = 1, 2, \dots$ para que $\epsilon \approx 1$. De este modo la contribución de cada una de las funciones de onda de las activades al hito es máxima en éste.

La probabilidad del hito es $P = |\Psi|^2$. Si el hito no es *perturbado*, es decir, si las actividades no se retrasan, las amplitdes de probabilidad de las mismas interfieren en fase creando un máximo en el patrón de interferencia. La amplitud $\left(\sqrt{\mathrm{K}}\right)^{-1}$ mide la mayor importancia de las activdades de mayor duración.

En la proximidad de un hito, la superposición de los campos de difracción de las actividades individuales componen un patrón de difracción definido entre los mínimos más próximos. Para interpretar este patrón como una densidad de probabilidad debe ser normalizada a 1.

Si una tarea se retrasa en ΔD días, su fase ϕ cambia en $\Delta\phi = \kappa_0 \Delta D$, cambiando la contribución de la actividad a la probabilidad del hito. La coherencia de la interferencia es parcialmente destruida y la probabilidad del hito desciende. El modelo recoge también el efecto de un desfase mayor para las tareas que dependen de otras tareas. Las actividades programadas para acabar inmediatamente antes de los hitos tienen un impacto mayor en la probabilidad de estos que otras tareas más alejadas. En cadenas de tareas mutuamente dependientes, la perturbación de la primera se propaga a las siguientes. Las tareas más alejadas contribuyen menos que las más próximas, etc.

Analizando la programación del proyecto, el modelo calcula dos parámetros, la incertidumbre de hito, *milestone date uncertainty*-MDU, y el periodo de recuperación, PR. MDU depende de la distribución de las duraciones de las tareas y las holguras, decrece para tareas cortas.El PR, como indica su nombre, no es más que el plazo necesario para corregir los posibles retrasos que puedan producirse antes del hito. Ambos parámetros *dependen sólo* de la estructura de descomposición en tareas y proporcionan la tolerancia, en términos de unidades de tiempo, de la duración de las actividades así como la capacidad de recuperación del proyecto. Si se dispone de información adicional sobre el "clima" del proyecto, se pueden escalar dichos parámetros para reflejar situaciones de mayor o menor riesgo.

Al contrario que en el caso del camino crítico en el que la demora en una de las actividades situadas sobre ese camino cmabia la probabildiad de los hitos del 100 % al 0 %, este modelo refleja que cualquier variación en cualquier tarea afectará en alguna medida al proyecto. Matemáticamente eso se debe al desplzamiento de fase de la función de onda de las actividades, pero en la práctica lo que refleja es que cualquier retraso en alguna tarea afecta a los hitos al reducir la flexibilidad de la programación.Presenta, pues, una representación continua puesto que los hitos son eventos con una distribución continua en el tiempo y no eventos puntuales. Este modelo está implementado en una aplicación comercial denominada $Quantum^{TM}$ de la que puede obtenerse información y

algún ejemplo en `www.ibico-cor.com`.

5.4. Conclusión y propuestas

En el capítulo 2 se propuso el marco de clasificación de los diferentes entornos *multiproyecto* que aparecen en la figura 2.1. Las circunstancias que afectaban al desarrollo se caracterizaban en términos de *variabilidad* y *dependencia*, de manera que se proponían cuantro alternativa, según las citadas magnitudes fueran *alta* o *baja* cada una de ellas.

Tras lo expuesto en este capítulo es preciso afinar un poco más en esa clasificación con el fin de identificar las mejores opciones para el problema en estudio. Para ello hay que tener en cuenta, en primer lugar, la cuestión de la *dependencia*. La descomposición del problema en los niveles táctico y operacional tiene como objetivo principal *aislar* la cuestión de la dependencia en el nivel táctico. De esta forma, el nivel operacional actúa en un contexto de *independencia relativa*.

Desde el punto de vista de la *variabilidad* en cambio la situación es la inversa. En el plano operacional la variabilidad es mayor, debido a las razones siguientes:

- La WBS tien una granularidad más fina, el detalle de tareas es mayor, lo que redunda en una multiplicación de los factores de incertidumbre
- Los eventos externos que pueden provocar disrupciones son más, por la misma razón
- Las decisiones tácticas pueden actuar a modo de "eventos" externos cuando se reasignen recursos entre unos y otros proyectos

En estas condiciones, en [HL04b] se recomienda el empleo de estrategias *predictivo-reactivas* como las propuestas en 5.2.4 o de *reglas de despacho* en el sentido expuesto en 5.2.5 en el caso de que la variabilidad sea extrema. Dentro de estas estrategias, parece que la opción que más se acomoda al caso en estudio es la denominada *secuenciación contingente*, es decir, la que le permite al director del proyecto evaluar diferentes alternativas a la hora de reprogramar.

En el nivel táctico, con muy alta dependencia y una variabilidad menor, los mismos autores recomiendan el empleo de *planes estables* y *robustos*. En este capítulo se han visto tres aproximaciones, la del método de la Cadena Crítica, otra basada en modelos de flujo y programación lineal y una tercera basada en la lógica borrosa.

El método de la Cadena Crítica es uno de los más "populares". Sin embargo también tiene sus detractores como se ha indicado. Desde el punto de vitsa que aquí interesa, la aplicación del método de la Cadena Crítica al entorno multiproyecto se basa en secuenciar *alrededor del recurso más limitante*. En la producción de *software* es difícil que se pueda identificar un recurso *más limitante*. Existen equipos y personal con diferentes perfiles, formación y capacidades, pero todos son sustituibles, con mayor o menor rendimiento. La única limitación puede venir de la cantidad de recursos y el problema lo que se plantea es precisamente decidir los recursos adicionales que hay que adquirir.

La segunda alternativa, la basada en modelos de programación lineal y algoritmos de flujo se ha elaborado con miras a entornos de taller y producción. Se basan en la adquisicón de recursos extraordinarios, cuando el problema está en decidir la asignación de actividades a los recursos existentes o la ampliación de los mismos.

Aparentemente, la tercera metodología detectada, la basada en la lógica borrosa, es la más interesante, además de haberse desarrollado específicamente para el problema de asignación de equipos a proyectos *software*. Como se ha indicado en el apartado 5.2.3, se trata además de sistemas de apoyo a la decisión (DSS) que evaluan soluciones basadas en reglas de prioridad, algo que las hace más accesibles a las personas responsables de la gestión de los proyectos.

Queda por valorar el potencial de las denominadas aproximaciones *alternativas*. De los casos comentados, la más prometedora parece la basada en la DSM (*Design Structure Matrix*) a efectos de analizar y validar la estructura de la descomposición en tareas de los proyectos *software*. Se trata de una herramienta complementaria que pude ser útil a la hora de la planificación inicial y de optar por un determinado modelo de ciclo de vida.

En ste capítulo se ha ido viendo que las aproximaciones más prometedoras consisten en *herramientas y modelos de apoyo a la decisión*: a

la hora de diseñar las arquitecturas de las WBS, a la hora de secuenciar las tareas de un proyecto y a la hora de asignar recursos dentro de una cartera de proyectos. Todo esto apunta a la oportunidad de la *simulación*.

En la Ingeniería del *Software* existe un cuerpo importante de modelos y propuestas al respecto, que es lo que se analiza en el siguiente capítulo.

Capítulo 6

Simulación del proceso *software*

6.1. Objetivo y contenidos

Este capítulo tiene como objetivo enmarcar las diferentes opciones utilizadas en la modelización para la simulación del proceso *software* de cara a su empleo en la gestión de proyectos de desarrollo en el entorno multiproyecto.

Se caracterizan los modelos a partir de referencias clásicas en la bibliografía. Se repasan algunos de los modelos más significativos. Se presta especial atención a tres aspectos:

- Modelos que combinan la simulación continua con la discreta
- Modelos que tienen en cuenta las acciones encaminadas a la mejora de procesos como una actividad más en el proyecto
- Modelos que implementan en algún modo los métodos y técnicas de la gestión de proyectos tratados en los capítulos anteriores

El capítulo concluye proponiendo criterios, formalismos y metodologías para el trabajo futuro.

6.2. Las razones para la simulación del proceso *software*.

En un artículo publicado en 1999, como conclusión del primer taller PROSIM, Keller, Madachy, Raffo RefWorks:134 proponen una visión panorámica del trabajo realizado en la simulación de procesos *software* basada en tres preguntas *¿Con qué objetivo modelar? ¿Qué modelar? y ¿Cómo modelar?*

6.2.1. Objetivos de la simulación: ¿Para qué modelar?

Entre los objetivos se señalan los siguientes:

Gestión estratégica de los procesos *software*. Los modelos de simulación permiten, según los autores, recoger las consecuencias que la forma general de organizar el proceso *software* tiene en la evolución estratégica de organización desarroladora, en términos de creación de valor, posicionamiento de mercado, etc. Así, las prácticas de deslocalizar, subcontratar, desarrollar en base a componentes o mediante aplicaciones "ad hoc", etc. pueden evaluarse en términos de su impacto en el posicionamiento general de la empresa.

Planificación de los procesos. La simulación posibilita también la planificación proporcionando estimaciones de carga de trabajo y esfuerzo, utilización de recursos, analizar riesgos, etc. La planificación no se limita a simular en el momento previo al inicio de un desarrollo sino que puede ayudar al seguimiento y evaluación, así como a eventuales cambios en la planificación.

Gestión operativa de los procesos. Con un nivel de granularidad más fino, los modelos de simulación pueden stabién ervir para la gestión operativa. Aquí los modelos de simulación por el método de MonteCarlo pueden servir para controlar desviaciones, estimar tasas de errores y necesidades de corrección, etc.

Mejora de los procesos y adopción de tecnología. Las decisiones sobre modificar los procesos o adoptar determinadas herramientas u otros elementos pueden ensayarse sobre modelos de simulación antes de ponerse en práctica, midiendo las consecuencias inmediatas pero también las consecuencias de segundo orden que pueden derviarse, sin incurrir en los costes de una decisión errónea.

Comprensión de los procesos. La construcción de un modelo de simulación y el análisis de escenarios y resultados implica un esfuerzo por desentrañar los mecanismos que actúan en la práctica del proceso *software* del cual se puede obtener un mayor grado de comprensión de la realidad, descubriendo por contraste con los datos que la misma arroja los aspectos que permanecen ocultos.

Entrenamiento y aprendizaje. Por último, los modelos de simulación son un instrumento potencialmente utilizable en el aprendizaje y entrenamiento en un área de conocimiento donde la experimentación puede resultar prohibitiva.

Todos estos aspectos se refuerzan si se tiene en cuenta que los modelos analíticos que se emplean para estudiar los procesos presentan el inconveniente de que, en general, resultan demasiado abstractos y obvian las características específicas de cada organización de desarrollo, que no deja de sr un sistema sociotécnico complejo donde a vec es los datos numéricos no reflejan más que muy parcialmente la realidad, no se pueden aplicar técnicas estadísticas y a veces no hay manera de entender las desviaciones entre lo predicho por el modelo analítico y lo que ocurre en la práctica, ya que el modelo analítico no ofrece una buena guía para ello.

6.2.2. Alcance de la Simulación: ¿Qué modelar?

Desde el punto de vista del alcance, los modelos de simulacón pueden abarcar desde una parte del ciclo de vida de un producto concreto hasta la evolución de la organización a largo plazo, pasando por un proyecto, varios proyectos o la evolución en el tiempo de una línea o líneas de producto a través de sus sucesivas versiones. Desde el punto de vista de los resultados perseguidos, un modelo de simulación puede informar

sobre esfuerzo empleado, costes, recursos asignados o tasas de error hasta variables más globales relativas al valor conseguido, la evolución de las plantillas de personal o el nivel de calidad global. De manera muy particular, los modelos de simulación pueden informar sobre el riesgo en los diferentes niveles expuestos en relación con el alcance del modelo; desde el de no cumplir un hito hasta la exposición global al riesgo de la empresa en el mercado.

Para ello los modelos de simulación utilizan diferentes niveles de abstracción de los procesos modelados. Este dependerá del interés que guía el modelo. Así se pueden identificar tareas y recursos clave, o enfatizar el flujo de objetos a lo largo del proceso, las interdependencias entre tareas, las relaciones y bucles de realimentación, etc.

Para obtener resultados relevantes es igualmente necesario proporcionar los parámteros de entrada adecuados, que pueden ser medidas de tamaño, productividad, ... o parámteros de los procesos organizativos como el ciclo de y los retrasos en la decisión de contratar, los intervalos de medida y (re)programación, etc.

6.2.3. Metodologías de simulación: ¿Cómo modelar?

Las metodologías de simulación empeladas son bastante diversas aunque predominan la simulación continua (Dinámica de Sistemas) y la simulación de eventos discretos. Ambas metodologías se prestan específicamente a representar y estudiar facetas diferentes del proceso de producción de *software*. La primera permite analizar la estructura de interrelaciones complejas entre las diferentes variables de estado del sistema en estudio, especialmente cuando estas relaciones son fuertemente no lineales y los métodos analíticos resultan poco adecuados. Una organización dedicada al desarrollo de productos *software* es un sistema sociotécnico complejo,en los que las interacciones humanas, los flujos de comunicación y la presencia de objetivos diferentes y, hasta contradictorios, requieren este tipo de representación.

La metodología orientada a eventos discretos posibilita el estudio del ciclo de vida de entidades individuales diferenciadas y su circulación a

través de las diferentes componentes del sistema. La producción de *software* está plagada de este tipo de entidades: desde el propio producto, que no es siplemente un "bloque de código" más o menos grande sino una unidad con su funcionalidad propia y diferente de la de otros productos, hasta la multiplicidad de entregables y otro tipo de artefactos que se generan en la producción de *software*. Los propios recursos que se emplean en la producción no son simplemente acumulables; el recurso principal son personas, cada una con sus diferentes capacidades, conocimientos y habilidades. Si bien el tratamiento del ciclo de vida de objetos individuales en un sistema es objeto de estudio desde hace mucho tiempo, aquí lo mismo o más que en el análisis de relaciones estrcuturales en los sistemas continuos, los métodos analíticos no son suficientes para tratar suficientemente el problema.

En [ZL04] se cita el dato siguiente: revisando los *proceedings* de los talleres PROSIM entre 1998 y 2006, el 54 % de los modelos presentados son de Dinámica de Sistemas (DS) y el 27 % restante, simulación de eventos discretos (SED). A gran distancia en el número de casos que se recogen en la literatura aparece un segundo grupo de metodologías. Estas lo conforman las redes de Petri y la simulación basada en agentes.

Las redes de Petri (y otros modelos anteriores basados en redes estocásticas o en otro tipo de representación de estados) tienen una primera aplicación a la simulación estocástica tipo Montecarlo que aparentemente no se diferencia gran cosa de la simulación de eventos discretos, simplemente con un grado de formalización mayor. Sin embargo esta formalización tiene otras potencialidades que han motivado su empleo espcífico, tanto en el campo del tratamiento analítico como su posibilidad de *instanciamiento* del propio modelo [dSP05].

La metodología basada en agentes es la más novedosa de las citadas porque permite capturar el carácter distribuido de los procesos y simular procesos de negociación entre partes y reglas de decisión implementadas de forma local y no globalmente embebidas en el sistema.

En los siguientes apartados se comentan con más detalle algunas de las aplicaciones más interesantes de estas meotdologías. Están clasificadas en cuatro grupos:

a) Aplicaciones más o menos basadas en el modelo de Abdel Hamid

y Madnick que aportan aproximaciones nueva, bien por el área de interés que focalizan, bien por integrar varias metodologías

b) Aplicaciones orientadas específicamente al entorno multiproyecto

c) Aplicaciones que tienen en cuenta la mejora de los procesos *software*

d) Aplicaciones que incluyen alguno de los métodos de la gestión de proyectos revisados en los capítulos anteriores

Además de las referencias indicadas, se han revisado las que aparecen en la tabla siguiente:

Referencia	**Descripción**
[LCWB08]	Diseño iterativo
[NZBS07]	Desviaciones de coste en líneas de producto *software*
[PAER07]	Estabilidad en el lanzamiento de versiones
[LM04]	Cadena Crítica y DS
[RHB03]	Participación y compromiso en equipos de desarrollo
[ARRRR02]	Reglas de gestión a partir de la simulación
[BWT02]	Six Sigma
[LKR02]	Complejidad en la evolución del *software*
[WH02]	Desarrollo global de *software*
[KLRW01]	Complejidad en la evolución del *software*
[RRT01]	Diseño colaborativo
[SG01]	Diseño colaborativo
[HH00]	Desarrollo de grandes sistemas
[MT00]	Estudios de caso
[PL00b]	Construcción de modelos
[PL00a]	Impacto de la volatilidad de los requisitos
[RKP+99]	Métodos empíricos y DS
[Abd93a]	Reutilización de componentes

Cuadro 6.1: Referencias de Dinámica de Sistemas no comentadas en el texto

Referencia	Descripción
[KG07]	Mejora de procesos
[Pad05]	Programación dinámica y simulación
[Pad04b]	Simulación de colas
[Pad02a]	Comparación de políticas de secuenciación de tareas
[Pad02b]	Programación dinámica

Cuadro 6.2: Referencias de simulación de eventos discretos no comentadas en el texto

Referencia	Descripción
[HBH+06]	Redes de Petri
[KC06]	Redes de Petri
[dSP05]	Redes de Petri - constructivo
[EKR95]	Work-Flow (analogía)
[XOZ+07]	Agentes
[SCFR06]	Agentes - código abierto
[RRT+03]	Agentes
[XMQ+04]	Agentes adaptativos
[NC03]	Agentes distribuidos
[CDSC05]	DSM

Cuadro 6.3: Referencias de otras metodologías de simulación

6.3. Modelos de simulación del proceso *software*

En la tesis de Mercedes Ruiz Carreira [Car03] se recoge una relación de trabajos de simulación del proceso *software* entre los años 1991 y 2000. En el origen de todos ellos está el trabajo de Abdel-Hamid y Madnick [AM91] que supuso un exhaustivo intento de modelar el proceso de inspección formal con un grado de detalle muy elevado. Reúne las actividades relacionadas con la gestión de recursos humanos, la producción, la planificación y el control en un modelo de Dinámica de Sistemas que se ha convertido en una referencia en este campo de investigación.A partir de ahí en la misma fuente se recogen referencias a una serie de trabajos que han ido enfocando diferentes perspectivas y problemas dentro del marco general antes citado.

Con posterioridad a la publicación de este trabajo, la producción no ha cesado, enriqueciéndose el campo de la producción de modelos y acometiéndose esfuerzos de modelado híbrido dentro del campo de la simulación e integrándolo con métodos y herramientas de otros campos, contemplandose:

- El empleo simultáneo de varios paradigmas de modelado (continuo y discreto, fundamentalmente)
- La integración entre la simulación y la aplicación de métodos y modelos de optimización del campo de la Investigación Operativa y de la Inteligencia Artificial
- La integración de la simulación dentro de herramientas de ayuda a la decisión en tiempo de ejecución básándose en la metodología de agentes
- La integración de la simulación en modelos generales de representación y explotación del conocimiento

Desde el punto de vista académico y profesional la referencia fundamental es la serie de talleres PROSIM (*Workshop of Software Process Simulation and Modeling*) iniciada en 1998 y fusionada a partir de 2006 con el taller internacional SPW (*Software Process Workshop*), que se

celebra en el marco de las conferencias ICSE (*International Conference on Software Engineering*) organizadas por la SCM y el IEEE.

En lo que sigue de este apartado se exponen algunos de los trabajos más interesantes revisados, remitiendo para una relación más exhaustiva a la ya citada tesis.

6.3.1. La integración de la simulación con otras áreas de la ingeniería del *software*, el modelo *IMMoS*

El modelo *IMMoS* [PR02] combina el modelado a tres niveles, con Dinámica de Sistemas modelos de procesos y el empleo de modelos cuantitativos basados en la medida. La motivación básica es la "falta de procedimientos bien definidos y repetibles para generar o empelar información producida en la práctica real que sea adecuada para la construcción de modelos de DS", de ahí el nombre de la metodología, *integrated measurement, modeling and simulation.*

Para ello se fija cuatro objetivos y se dota de un componente apropiado para ello:

a) Guiar el proceso de construcción del modelo. El primer componente proporciona un modelo de ciclo de vida para el modelo de DS además de modelos de los roles, los productos y el proceso.

b) Dar soporte a la definición de objetivos de modelado. para ello proporciona una taxonomía de objetivos que persiguen capturar la definición del problema en las etapas tempranas del modelado.

c) Facilitar la integración de los modelos de DS con los modelos estáticos de procesos *software* existentes. El componente correspondiente traza un mapa entre los modelos estáticos y las facetas del modelo de DS.

d) Integrar las dos formas de modelado con una metodología de medición basada en la GQM. Propone un enfoque GQM "dinámico".

Según las conclusiones de los autores, el resultado alcanzado requiere un esfuerzo de modularización para hacer posible la reutilización del modelo sin un esfuerzo excesivo de readaptación a cada caso. Por otra parte, se señalan objetivos en relación con el análisis coste-benefcio del empleo de estos modelos frente a otros métodos basados en la construcción de experimentos.

6.3.2. Simulación para la fiabilidad; el modelo de Rus, Colofello, y Lakey

Este modelo [RCL99] plantea la simulación como un sistema de apoyo a la decisión, DSS: Enfoca el problema específico de la fiabilidad del *software* y su calidad, evaluando el efecto de diferentes políticas de calidad y dando apoyo a la gestión y planificación de los proyectos softare desde esta perspectiva.

Se plantea preguntas como:

- ¿Qué tiempo y esfuerzo se precisa para completar el diseño preliminar?
- Cuántos defectos se generarán en esa fase?
- ¿Cuánto aumenta el esfuerzo si se emplean prácticas de prevención de dfectos?

En este modelo se introduce la *simulación de eventos discretos* como consecuencia de la necesidad detectada de considerar los atributos individuales de las entidades. Así, por ejemplo, los objetos de diseño y de código difieren entre si en tamaño, complejidad y modularidad. La duración de las actividades y el tiempo para generar y producir dichos objetos puede variar. El modelo resultante es un *modelo híbrido* en el que aparecen bucles de realimentación entre los factores continuos del proceso (mano de obra, programación y presión del plazo, sobre caraga de comunicación, ...) y los factores discretos del producto (tamaño, calidad y complejidad).

6.3.3. ¿Los procesos son contínuos o discretos? El modelo híbrido de Donzelli y Iazeolla.

Este modelo [DI01] aborda el modelado proceso *software* a tres niveles, como un sistema continuo aplicando dinámica de sistemas, como un sistema guiado por eventos, es decir de forma discreta, y por último tratandolo como un problema analítico.

A partir de un modelo de proceso *en cascada* el modelo se desenvuelve en *dos niveles de abstracción*, uno superior en el que el proceso se modela como una cola de objetos que van desplazandose entre estaciones como objetos discretos. El nivel inferior modela la dinámica interna de cada uno de esos movimientos como procesos continuos o a partir de magnitudes promedio, empleando los modelos analíticos de la teoría de colas.

Este modelo tiene como principal inconveniente el que no se establecen relaciones de realimentación entre ambos niveles sino que los mecanismos de realimentación se mantienen confinados a la explicación de la dinámica de los procesos a la escala más fina. Evidentemente, más en un entrono multiproyecto, las relaciones entre los diferentes niveles, es decir, la capacidad de que un proyecto influya en la ejecución de todos los demás, es una faceta que la simulación debe ser capaz de representar.

6.3.4. La visión inversa: el modelo híbrido de Martin y Raffo

En [RM00] se presenta una aplicación conjunta de simulación discreta y continua. El modelo está implementado en EXTEND y persigue evaluar los cambios potenciales en el proceso, modelando las etapas del proceso como entidades discretas con sus atributos propios lo que permite calcular el tiempo, esfuerzo y tasa de error. Estos datos son pasados a un modelo de DS que representa la evolución continua del entorno del proyecto. A su vez, el entorno proporciona las variables donde evolucionan las entidades discretas.

Cada entidad computa su propio tiempo de evolución pero sólamente en intervalos discretos δ definidos para la simulación de eventos. Eso hace

que la evolución de las actividades no pueda calcularse dinámicamente. Por el contrario, las decisiones de asignación de recursos se toman de manera discreta lo cual resulta más realista que la asignación continua del modelo tradicional de Abdel-Hamid, aunque al neviel del entrono del proyecto se mantenga una evolución continua de los factores que afectan a la fuerza de trabajo.

6.3.5. Simulación híbrida para las compañías de *software* global

En [SWR07] Se presenta un trabajo que emplea la simulación híbrida para modelar el problema de la organización de la producción de *software* según la estrategia denominada "seguir al sol", *follow-the-sun*, practicada por las grandes compañías para acelerar los desarrollos. El modelo se asemeja bastante al descrito en el apartado anterior. La principal variación radica en que añade una dimensión más para recoger la existencia de diversas localizaciones geográficas.

Así pues el modelo contiene cuatro niveles:

- A nivel local:
  - un modelo discreto que representa las tareas y los objetos entregables
  - un modelo continuo que representa la capacidad en términos de mano de obra disponbile así como las otras magnitudes relacionadas al entorno de trabajo (aprendizaje, presión de plazso y productividad, etc.)
- A nivel global:
  - un modelo discreto que representa el movimiento de los artefactos entre las diferentes localidades "siguiendo al sol"
  - un modelo continuo que recoge la gestión global de los recursos y de la mano de obra, la planificación y el control sobre la ejecución

Este modelo permite a los autores formular hipótesis sobre la complejidad que emerge del trabajo conjunto de equipos con diferentes culturas, lenguas y actitudes. Estas hipótesis se contrastan con datos empíricos provenientes de estudios realizados en otras áreas de conocimiento viéndose validadas las hipótesis formuladas.

6.3.6. Análisis de sensibilidad para identificar los factores de riesgo en los proyectos

Un grupo de la Arizona State University [HMC01] readpatan el modelo de Abdel-Hamid y el de Tvedt [TC95] para evaluar seis factores de riesgo:

- El crecimiento incontrolado de los requisitos
- La inexactitud en la estimación de los costes
- La presión del plazo
- La falta de compromiso y la baja moral en el equipo de desarrollo
- La inestabilidad de la plantilla de desarrollo
- La falta de compromiso de la Dirección

Para cada una de estas dimensiones, los autores identifican una serie de variables significativas y simulan parametrizando dichas varialbes para obtener una estimación por intervalos de confianza del plazo de realización y del coste total de los proyectos.

Este mismo grupo de investigadores analiza el *comportamiento* de diferentes modelos frente a esas mismas variables. En [HFC+01] aportan una interesante reflexión sobre como las opciones para modelar el comportamiento de determinadas variables y relaciones se manifiesta en los resultadso del análisis de sensibilidad. Aconsejan utilizar este análisis para criticar también las suposiciones sobre las que los modeladores han ido construyendo.

6.3.7. Simulación de la interacción entre grupos humanos en un proyecto *software*

En [SG01] se presenta una aplicación de una simulación basada en DS al proceso de elicitación cooperativa de requisitos dentro de la filosofía del modelo *WinWin* de Boehm. Para ello los autores tienen que trasladar al modelo a simular, además de los formalismos de los procesos objetivos, representaciones de fenómentos sociales y de comportamiento humano.

6.3.8. El modelo dinámico reducido

La acumulación de experiencias en construcción de modelos va decantando una serie de *comportamientos típicos* de los proyectos que aparecen recurrentemente. Para identificar estos comportamientos no es necesario un modelado exhasutivo caso por caso sino que pueden platearse a modo de *hechos estilizados* una serie de pautas básicas y de mecanismos que las explican. A raiz del trabajo de la tesis doctoral de Isabel Ramos [Ram99] se pone en marcha una línea de trabajo que se presenta en [RRT01] donde se propone un *modelo dinámico reducido* (MDR) que persigue:

a) 1. Poder realizar estimaciones en las primeras etapas de un proyecto *software*, etapas en las que aún se dispone de poca información sobre el mismo. Especialmente para:

- Aquellas organizaciones o empresas que no disponen de una base histórica de proyectos, o la información necesaria para definir tanto las características del entorno de desarrollo en el que se trabaja (relativas al proyecto y a la organización o empresa) como las características que permitan definir el grado de madurez adquirido.
- Las organizaciones o empresas que trabajan en entornos de desarrollos diferentes y/o variables.

b) Poder enseñar a nuestros alumnos, o al personal novel en temas de gestión de PDS, cuáles son los lazos o bucles de realimentación básicos que definen el comportamiento de las variables fundamentales de dichos proyectos.

c) Obtener reglas de gestión donde intervenga un número reducido de parámetros.

El MDR se obtiene manteniendo la estructura en subsistemas del Modelo de Abdel-Hamid y Madnick [AM91], y se implementa en el entorno de simulación *Vensim*. La creación de modelos simplificados, a partir de otros ya existentes, se viene realizando en otros campos de conocimiento, fundamentalmente, a partir de los trabajos de simplificación de modelos dinámicos realizados por Robert L. Eberlein [Ebe89]. Estos trabajos están centrados en la reducción de modelos dinámicos de un cierto tamaño, para obtener un modelo más pequeño en los que se eliminan los bucles de realimentación que se consideran que no son esenciales para el comportamiento que se quiere analizar. Según Eberlein, la simplificación de modelos dinámicos grandes y/o complejos no solamente es útil, sino que además es un trabajo que debe de realizarse siempre para conocer los bucles de realimentación básicos. El modelo reduce sustancialmente el tamaño del modelo original identificando cuatro bucles de realimentación básicos que gobiernan la trayectorai global del proyecto:

- En primer lugar está el mecanismo que regula el empleo de recursos, modelado en este caso como *contratación de personal*, en función de las estimaciones de carga de trabajo que van surgiendo
- En segundo lugar se modela el mecanismo a través del cual la *presión del plazo* incrementa la posibilidad de errores en la codificación, de manera que el trabajo se amplifica por la necesidad de corregirlos posteriormente
- En tercer lugar se plantea un bucle de realimentación que modela el impacto de la incorporación de nuevo personal al equipo de desarrollo, incorporación que se traduce en un incremento de la *sobrecarga de comunicación* derivada del

tamaño creciente del equipo, y el *efecto aprendizaje* que reduce la productividad efectiva del conjunto

- El último bucle contemplado refleja el efecto no-lineal de la presión del plazo sobre la *productividad*.

Con este modelo reducido se puede conseguir una visión sintética de algunos de los mecanismos fundamentales que provocan la incertidumbre inherente al ciclo de vida de un proyecto, recogiendo las no-linealidades características de un sistema socio-técnico.

A partir de aquí el modelo se ha utilizado para ganar en la comprensión del efecto de las diferentes *reglas de gestión* sobre la dinámica del proyecto y la capacidad mayor o menor de alcanzar los objetivos del mismo en tiempo y plazo. Para ello se han desarrollado métodos para la inferencia de reglas a partir de algortimos evolutivos y minería de datos que se presentan en [ARRRT01] y [MRGT08].

6.3.9. Integrando simulación discreta y continua con el formalismo DEVS

En [CBK06] se presenta una aplicación del formalismo DEVS, *discrete event system specification*, propuesto por Ziegler ([ZKP00]) que posibilita representar con naturalidad un sistema híbrido al gestionar conjuntamente una simulación continua con un manejo muy flexible de la lógica de eventos y los flujos de información asociados al inicio y finalización de tareas y a la monitorización de las mismas. Este formalismo presenta además la ventaja de que el modelo básico, basado en un sistema muy simple denominado el DEVS *atómico* y en un acoplamiento elemental de dos átomos, es escalable y muy fácilmente modelable con la tecnología orientada a objetos. Esto permite generar una clase de modelos fácilmente parametrizables y reutilizables.

El ejemplo que presentan es una "relectura" del modelo clásico de Abdel Hamid y Madnick en un proceso con ciclo de vida en cascada en el que se pudene diferenciar, y a la vez mantener coherentes, los procesos continuos del entorno de la organización y del

propio desarrollo de las tareas elementales, sujetas al tipo de interacciones que sintetiza el MDR, y los procesos discretos tanto de flujos materiales (entregables, módulos, ...) como de medida, control y decisión.

6.4. Simulación del proceso software en el entorno multiproyecto

El entorno multiproyecto, con sus elevadas dependencias, está naturalmente inclinado a multiplicar los efectos cruzados de los diferentes bucles de realimentación que gobiernna la evolución del estado de la organización de desarrollo. Es significativo, por tanto, que el esfuerzo por la simulación de este tipo de entorno sea relativamente reducido en comparación con el caso de un único proyecto.El propio Abdel Hamid en su artículo [Abd93b] llama la atención sobre lo inapropiados que pueden ser los métodos y modelos de estimación cuando no tienen en cuenta las implicaciones a escala de toda la organización. En los apartados siguientes se presentan los dos modelos más interesantes que se han detectado.

6.4.1. Abdel Hamid: Desmintiendo la Ley de Brooks

En el artículo citado [Abd93b] se presenta un trabajo en el que se modela específicamente la transferencia de prosnal entre varios proyectos que se destán desarrollando en paralelo. La transferencia obedece a una serie de reglas relacionadas con la presión del plazo. Se realizan dos experimentos: uno de aceleración de los plazos y otro en el que se emula la regla MINSLACK para asignar recursos de la organización a los proyectos que compiten por ellos.

El primer experimento corrobora lo señalado sobre la necesidad de incorporar en las estimaciones del coste de acelerar los plazos, los efectos que esa decisión tiene no sólo sobre el coste *directo* del

proyecto afectado sino también el coste de *oportunidad* que no se manifiesta directamente sino a nivel de toda la organización.

El segundo experimento proporciona la oportunidad de relativizar la percepción generalizada sobre la denominada "Ley de Brooks". La adición de nuevos recursos a un proyecto, si bien reduce la productividad promedio del equipo de desarrollo a través de las *sobrecargas de comunicación* y del *efecto aprendizaje*, puede aumentar la productividad total, acelerando así los plazos, siempre que se den las condiciones que Boehm denomina *proyectos orgánicos*, es decir proyectos de un tipo semejante a otros que se desarrollan en la organización.

Conviene repetir que, a pesar de la afirmación del autor, el trabajo posterior en el entorno multiproyecto es bastante escaso.

6.4.2. Powell *et al*: El desarrollo incremental desde la perspectiva multiproyecto

En [PMB99] se presenta un modelo de dinámica de sistemas en el que se modela un ciclo de vida incremental en términos que recuerdan al entorno de proyectos. El ciclo de vida incremental conlleva relaciones de concurrencia entre diferentes paquetes de trabajo que, a otra escala, equivalen a la competencia por recursos de varios proyectos simultáneos, con la complicación adicional que dicha interrelación se establece no sólo por esa competencia por los recursos sino también por la información que fluye de un paquete de trabajo a otro, acoplándolos por tanto también a través de ese mecanismo.

Los autores proponen un modelo sencillo, una vez más basado en una estrucutra simplifcada del modelo de Abdel Hamid, que relaciona recursos, tiempo y esfuerzo con el paquete de trabajo. Esta sencilla estructura se denomina el *tirángulo del proceso*. Este triángulo puede definirse, modularmente, a diferentes niveles jerárquicos: paquetes de trabajo, fases, entregas, proyectos y organización en su conjunto.

Para acoplarlos dieferentes módulos se emplea tanto la asignación de recursos a módulos como el acoplamiento de los trabajos. El

primero es una regla de reparto de recursos entre paquetes de trabajo (o módulos de nivel superior) que recuerda a la RWK tratada en 4.4(página tab:reglas): en base a prioridades y presiones de plazo. El acoplamiento de los trabajos implica tratar el trabajo de *corrección de errores* y las *transiciones*. La corrección de errores puede aparecer dentro de un mismo paquete de trabajo o en otro subsiguiente. Trasladando la corrección de errores a una entrega posterior se *externalizan* estas tareas. Las transiciones se producen cuando una fase madura lo suficiente como para iniciarse la siguiente. Las transiciones pueden producirse de diversas formas: mediante *hitos* o por *unidades* de trabajo que al concluirse se convierten en *inputs* de las siguientes. Esto permite modelar la superposición de fases. El objetivo último del modelo es analizar las estrategias de desarrollo incremental, tanto a nivel previo (planificación) como su implementación y su modificación en tiempo de ejecución.

6.5. Simulación y mejora de procesos *software*

Como se ha indicado más arriba, una de las aplicaciones de la construcción de modelos para la simulación del proceso *software* es la de evaluar los cambios en el propio proceso. La puesta en marcha de programas para la mejora del proceso *software* en las organizaciones de desarrollo se ha convertido, por tanto, en un área de interés natural para la simulación. Un cambio en los procesos debe traducirse en una mejora de la situación de la organización en términos de sus objetivos estratégicos. Pero ese cambio no es ni fácil de implementar ni es evidente la relación netre las medidas concretas y las mejoras deseadas. Ello se debe al carácter complejo e interrelacionado de la propia actividad y del entorno en que se desarrolla.

Existe además una relación inversa entre ambas cuestiones. Un proceso poco maduro y estructurado, con empleo reducido de técnicas de medición y de métodos cuantitativos en general, resulta extremadamente dificil de modelar de manera formal. Los

mecanismos que explican los resultados permanecen ocultos en un conglomerado de prácticas informales y poco transparentes. Por lo tanto el esfuerzo para construir el modelo en si mismo es un esfuerzo que ayuda a diagnosticar las prácticas y áreas susceptibles de mejora.

Estos hechos son conocidos por la comunidad investigadora y así han sido señalados desde hace tiempo [Chr99] por lo que existen varias referencias sobre la interacción entre modelos de simulación que sintetizaremos en dos por referirse una a cada uno de los estándares más generalizados: la familia CMM del SEI (*Software Engineering Institute*) [Pau95] y la norma ISO/IEC 15504([Rou03]).

6.5.1. El modelo de Stallinger para mejora de procesos

En [Sta00] se presenta un esfuerzo por modelar el proceso táctico de mejora del proceso *software* mediante un modelo generalizado de DS que se combina con el estándar de evaluación ISO/IEC TR 15504-5 (en su versión de *Technical Report* de 1998), intentando representar así los principios del paradigma de la calidad total, TQM.

El objetivo es dar soporte a la planificación del proceso de mejora a un nivel táctico caracterizando el impacto de un conjunto de medidas programadas de mejora en la totalidad del proceso. Eso lleva al modelo a identificar, de un lado, el comportamiento de los objetivos estratégicos de la mejora: el tiempo de realización, el coste, la calidad, etc. De otro lado implica identificar los costes del propio proceso de mejora, principalmente a través del coste mismo de los recursos empleados y de las nuevas actividades que aparecen como consecuencia de la formalización creciente de los procesos.

La filosofía es que el modelo sea aplicable tanto a organizaciones con un bajo nivel de madurez como a aquellas que ya están más avanzadas. En el primer caso, el modelo opera a la manera de *un simulador de vuelo* mediante una presentación de escenarios

de mejora en relación con la situación inicial de la organización que ha sido sometida a una evaluación previa. En el segundo caso, en organizaciones con mayor nivel de madurez, en las que las medidas para la efectividad de los procesos ya están implementadas, la aplicación modela explícitamente la implementación de las mismas como una tarea más dentro del proceso.

6.5.2. Marco dinámico integrado para la mejora de los procesos *software*

Frente a esta propuesta en la que la simulación se emplea como herramienta para apoyar el proceso de mejora, en [RRT02] se presenta, bajo el título *un marco dinámico integrado para la mejora de los procesos* software (DIFSPI, *Dynamic Integrated Framework for Software Process Improvement*),una metodología y un entorno de trabajo más "bidireccional" en el sentido de que es el grado de madurez el que a su vez determina las posibilidades de la simulación.. Este trabajo es un resultado de la investigación presentada en [Car03]. Una aproximación semejante se recoge en [RVM99] Uno de los objetivos principales de DIFSPI consiste en facilitar la evolución del nivel de madurez de las organizaciones de desarrollo de acuerdo con el modelo CMM. El propio diseño y construcción del marco y de los modelos dinámicos que lo integran permite, en sí mismo, la definición de métricas que, por un lado son necesarias para la inicialización y validación del modelo y que, por otro lado, sirven para la definición de un programa de recogidas de métricas concretas dentro de la organización. Esta recogida definida y sistemática de métricas persigue no sólo la utilización de modelos dinámicos sino además mejorar el conocimiento real del estado de los procesos dentro de la organización.

Una de las características definitorias del empleo de modelos dinámicos en cualquier área de la ingeniería es que sea cual fuere el modelo, éste requiere satisfacer un determinado nivel de calidad o fiabilidad antes de poder ser utilizado en la práctica. De hecho, la potencia y versatilidad de los sistemas de simulación actuales facilitan en gran medida la construcción de modelos dinámicos que

pueden llegar a ser realmente complejos. Sin embargo, si la organización no dispone de la suficiente información para inicializar y definir los parámetros y funciones que gobiernan la evolución del comportamiento del modelo, éste resultará inútil. El problema mencionado es uno de los inconvenientes más frecuentes con el que podemos encontrarnos cuando se intentan aplicar modelos clásicos de simulación en organizaciones situadas en niveles bajos de madurez, sobre todo en las etapas iniciales de los proyectos.

Los autores afirman que el beneficio obtenido con la aplicación de modelos dinámicos dentro de la organización está directamente relacionado con el conocimiento y la información empírica que la misma tenga de sus procesos y cómo valide sus procesos simulados, por lo que el nivel de madurez de la organización guarda una relación directa con la facilidad de validar los modelos dinámicos y, por tanto, con su fiabilidad.

Tras la definición de los componentes del sistema de métricas, éste puede implementarse en la organización, traduciéndose en la definición y desarrollo de una base de datos histórica. Estos datos de carácter histórico serán los que se utilizarán para la simulación y validación empírica del modelo construido. Una vez demostrada la validez de dicho modelo, los valores ofrecidos por la simulación permiten instanciar una base de datos que permita realizar análisis acerca de los efectos de diferentes acciones o mejoras.

Un aumento de la complejidad de las acciones que se pretenden evaluar, se traduce en un aumento de la complejidad del modelo dinámico que vuelve a diseñar un nuevo programa de recogida de métricas para los nuevos módulos de simulación, cerrándose, con ello, el ciclo. Si el nivel de madurez de la organización es bajo, estos datos permiten la observación de las tendencias gráficas que muestran la evolución del proyecto a partir de determinadas condiciones iniciales (establecidas por los valores de los parámetros de inicialización); cuando el nivel de madurez aumenta, la definición y conocimiento de los procesos de desarrollo también, por lo que los resultados de la simulación se convierten en estimaciones cuantitativas reales que predicen, de acuerdo con la incertidumbre de sus parámetros, los resultados futuros del proyecto. En

segundo lugar, la posibilidad de definir y experimentar diferentes escenarios sin asumir coste o riesgo alguno, es una de las principales cualidades de la simulación de modelos en cualquier área. También la gestión de proyectos *software* puede verse favorecida por esta cualidad. Es posible, por tanto, definir y experimentar diferentes mejoras de procesos, simular el modelo, y analizar los resultados de manera que la mejora o mejoras que se implementen de manera efectiva en el proyecto sean aquellas que arrojaron los mejores resultados durante la simulación. En tercer lugar, los modelos de simulación también se pueden utilizar la estimación de costes del proyecto; costes que pueden estar relacionados con algún sector concreto del mismo, como por ejemplo el coste de la calidad o el de las actividades de revisión, o con el proyecto completo.

Las capacidades para facilitar las predicciones de la evolución del proyecto, para establecer sus costes y para evaluar el resultado de diferentes mejoras del proceso constituyen tres de los factores principales para el progreso en los niveles de madurez del modelo CMM.

6.5.3. ¿Existe una correspondencia entre madurez y tipología de modelos de simulación?

En [ZL04] se plantea una cuestión que supone una persectiva diferente a las ya comentadas en torno a la relación entre la madurez de los procesos y la simulación de los mismos. Los autores sostienen que el grado de madurez determina el tipo de paradigma de simulación aplicable. Así, utilizando el modelo CMMMI por etapas, plantean que el mapa es el que aparece en el cuadro 6.4. Sin negar la relación evidente entre el nivel de frormalización de los procesos y las posibilidades de construir modelos de simulación, este planteamiento denota una excesiva rigidez. De hecho uno de los problemas característicos de la aplicación de las metodologías implícitas en los modelos de evaluación y mejora es el señalado por Day en [Day00], la traslación de la metáfora "ingenieril" a la gestión del proceso *software*, provoca paradójicamente la renuncia a emplear los métodos de la ingeniería. En

Nivel de madurez	Paradigma de Simulación
CM1	Simulación cualitativa
CM2	Simulación semicuantitativa
CM3	Dinámica de Sistemas
CM4	Simulación de Eventos Discretos
CM5	Simulación híbrida

Cuadro 6.4: Correspondencia entre niveles de madurez CMMI y metodologías de simulación según [ZL04]

[MHSZ04], los autores presentan un estudio empírico realizado entre empresas de desarrollo australianas que muestra que no hay correlación entre las prácticas de gestión y la madurez del proceso. Aún más, se postula que los modelos de madurez de los procesos de desarrollo probablemente no sean ni adecuados ni trasladables al dominio de la gestión de proyectos por su orientación a "procesos de flujo de taller" (*industrial-like flowshop processes*. Por otra parte, las referencias a la sobrecarga (*overhead*) provocada por las prácticas requeridas por los modelos de evaluación y madurez son una barrera importante a la adopción de estas por muchas entidades desarrolladoras como se cita en repetidas ocasiones ([ACC+04, SdOC03, Ric01, EB00]).

6.5.4. La simulación de las actividades de gestión y mejora de procesos

Teniendo presente lo anterior, otros autores ([MV06]) proponen la incorporación explícita de las actividades asociadas a los procesos de mejora a la descomposición en tareas, WBS, de los proyectos. De hecho, el modelo CMMI desde sus primeros niveles requiere la existencia de una descomposición en paquetes de trabajo de

todas las actividades. De hecho, enel modelo por etapas es preciso incorporar *todas las tareas*, uncluyendo las de medida del esfuerzo en la WBS desde el nivel 1. Para el nivel 2 se requiere implementar prácticas de planificación de recursos lo mismo que otras actividades "no constructivas" como la gestión de la configuración (*configuration management*.

El ciclo de vida total puede descomponerse en tres fases principales: la previa al proyecto, la ejecución del proyecto y la posterior al proyecto, en las cuales el esfuerzo es estimado inicalmente, medido, controlado y reestimado posteriormente, y al final evaluado y analizado. Aparte de las propias actividades implicadas estas tres fases invluyen también tareas relativas a las actividades "constructivas" y flujos de información entre unas y otras.

Durante la fase previa, el proyecto se programa y se pone a punto. La programación del proyecto debe incluir las actividades propias del desarrollo pero también las subactividades de gestión y control del proyecto. El seguimiento del proyecto durante la ejecución normalemente contempla la recogida de información sobre el esfuerzo invertido en las actividades constructivas pero debe *también* contemplar el esfuerzo de las propias actividades de gestión. Por último, el análisis *post mortem* del proyecto dbe incluir también el análisis de las actividades de gestión.

El modelado explícito de estas tareas tiene dos virtualidades:

- Proporciona mayor precisión en el cálculo del esfuerzo requerido que no es sólo el trabajo directo de análisis y construcción sino que incluye, además de la propia gestión, las tareas anexas a ésta que corresponden a lo que genéricamente se llama la sobrecarga de comunicación (*communication overhead*)

- Arroja un mapa efectivo para la mejora de los procesos al permitir contrastar el coste de la misma con el beneficio esperable

6.6. Simulación e investigación operativa en los proyectos *software*

Los modelos que se presentan a continución son modelos híbridos que emplean técnicas de la *investigación operativa y simulación de forma conjunta*. En los casos que se han revisado aparecen básicamente tres enfoques:

- generar una solución a partir de un modelo de investigación operativa y simular luego las consecuencias de su aplicación, generalmente por medio del método de Montecarlo
- generar el espacio de estados a través del proceso de simulación y luego aplicar un heurístico de la investigación operativa para optimizar el estado del sistema
- modelar el decisor explícitamente como un agente dotado de las reglas heurísticas; en este caso se trata de una herramienta de apoyo a la dirección del proyecto para que evalue las consecuencias de las decisiones adoptadas de forma dinámica

Los heurísticos que se han detectado en la literatura son de diferentes tipos:

- reglas simples de prioridad
- programación dinámica (métodos aproximados)
- algoritmos de búsqueda *greedy* (miopes)
- algoritmos genéticos
- *sequeaky wheel optimizaton*; un algoritmo de búsqueda en el espacio de las estrategias de prioridad

Los casos más interesantes que se han encontrado en la literatura son los que se describen a continuación.

6.6.1. Antoniol *et al*: Teoría de colas para la asignación de personal a las tareas de desarrollo

Los autores [ACLP04] Presentan una aproximación al problema basada en la teoría de colas y la simulación estocástica para el

apoyo a la planificación, gestión y control de la asignación de personal a un proyecto de mantenimiento multifase. Se trata de la adaptación al efecto 2000, *Y2K*, de un gran sistema de información de una entidad financiera desarrollado en COBOL/JCL que requiere la constitución de diferentes equipos trbajando en varios centros geográficamente distribuidos. Se estudia una configuración centralizada desde el punto de vista del cliente así como otras configuraciones más deagregadas. Los métodos de teorías de colas y la simulación esotcástica se emplean para evaluar las políticas de asignación de personal a los equipos y la posibilidad de alcanzar los hitos previstos para la ejecución del proyecto. En otro segundo artículo [APH04] se emplean los *algoritmos genéticos* para generar soluciones para el problema de teoría de colas teniendo en cuenta la necesidad de corregir errores y de reprogramar y abandonar algunas de las tareas inicialmente previstas. La simulación de Montecarlo de las colas junto con los algoritmos genéticos permiten generar secuencias y asignar personal a los equipos. El impacto posible del trabajo de corrección de errores, el abandono de tareas y los errores e incertidumbres en las estimaiones iniciales se caracterizan con funciones de distribución ue se utilizan como entradas para la simulación.

6.6.2. Padberg: Programación dinámica y simulación de reglas de asignación

En varios artículos [Pad02b, Pad04a, Pad04b, Pad03, Pad05] este autor presenta su trabajo en el que se modelan como problemas *markovianos* de decisión en los que se tienen en cuenta, además de la variación estocástica de la duración de las actividades, la posibilidad de vueltas atrás y retrasos. Para ello se generan políticas de secuenciación con un algoritmo aproximado de *programación dinámica*.

Con un conjunto de problemas similares pero que se diferencian en la necesidad mayor o menor de recursos especializados y el grado de integración entre los componentes del producto, se simula la aplicación de la política óptima generada por el algoritmo con las políticas clásicas de listas. Se compureba que cuanto mayor es la

necesidad de recursos especializados y cuanto menor es el grado de integracion entre los componentes, más destaca la solución propuesta sobre las reglas de listas.

6.6.3. Browning, Meire y Yassine: Simulación y algoritmos genéticos en el desarrollo de producto

El modelo que se presenta por estos autores [MYB07] se basa en la DSM ya descrita en el capítulo anterior. La varianza en la duración y el coste en los proyectos de desarrollo de producto son atribuibles según su criterio en gran medida a las iteraciones, algo que es común también en el proceso *software*. Dichas iteraciones pueden o no ocurrir dependiendo del comportamiento de una serie de variables que el modelo simula como variables estocásticas cuya probabilidad de ocurrencia depende de la existencia de determinados paquetes de información que activan un proceso de reelaboración (*rework*. Un proceso de reelaboración puede, a su vez, desencadenar otros sucesivos afectando así al proyecto en su totalidad. Para hacer frente a este problema utilizan el método de Montecarlo para simular políticas de secuenciación robustas generadas por un algoritmo genético.

6.6.4. Özdamar: Simulación de reglas de prioridad en proyectos definidos con el formalismo *fuzzy*

Este trabajo [zA01a] no consiste realmente en un modelo de simulación sino que se trata de una aplicación del MRCPSP a los proyectos *software*. Presenta un heurístico para la programación de las actividades de un modelo de desarrollo en cascada que se modela utilizando la lógica borrosa. la duración de las actividades depende del modo de ejecución seleccionado representado por una función de pertenencia trapezoidal. También se modela la disponibilidad de los recursos diferenciando entre dos catégorías, una de personal más cualificado que comparte varias tareas y - por

tanto - asignable de forma continua, y otra categoría de personal que no realiza multitareas y que se presenta en unidades discretas.

El objetivo del modelo es minimizar el tiempo de realización. Para ello se emplea un algoritmo de generación de secuencias en serie y se ensayan diferentes reglas de prioridad entre las más sencillas empleadas en los problemas de secuenciación. El modelo permite la resecuenciación dinámica para ecoger las variaciones en las estimaciones a lo largo de la ejecución del proyecto.

6.6.5. Joslin: simulación basada en agentes

En este trabajo [JP05] se presenta un modelo basado en agentes que simula de forma dinámica la reasignación de personal a las tareas en un proyecto con diferentes funcionalidades que deben entregarse en un plazo fijo.

Se trata de un sistema de ayuda a la decisión (DSS) basado en un heurístico denominado *Squeaky Wheel* que explora en el espacio de las estrategias en lugar de en el espacio de las soluciones. El simulador puede implementar diferentes reglas de asignación si bien la versión que se presenta en el artículo sólo presenta los primeros resultados para reglas muy sencillas.

La mecánica básica es la de reasignar el personal entre tareas conforme se aproximan los plazos y se detecta el riesgo de no cumplirlos. El modelo contempla el efecto aprendizaje cuando se produce un cambio de tarea y también el efecto de la sobrecarga de comunicación por lo que incorpora no linealidades el tipo de las que se presentan en los modelos clásicos de dinámica de sistemas.Los autores anuncian la futura implementacion de reglas más sofisticadas, como las basadas en algoritmos genéticos.

6.7. Conclusión y propuestas

Si bien el problema específico que se desea modelar en esta investigación no tiene una correspondencia clara en la literatura, sí que

hay en las referencias revisadas propuestas y aproximaciones que deberán considerarse en el trabajo futuro:

Los modelos reducidos como "átomos". El modelo presentado en el apartado 6.3.8 pueden servir para constituir el núcleo básico de la simulación de la evolución de las tareas. Su integración en un marco más amplio obligará a redefinir los bucles de realimentación que se cierran en el propio átomo y cuales traspasan las frontreas de estos.

Los triángulos de procesos. La estructura de integración jerárquica de los *triángulos de procesos* descritos en 6.4.2 permite a la vez modelar la jararquía tareas - fases - proyectos - cartera y representar, a través de los tres primeros niveles un modelo de desarrollo incremental, que puede reducirse al modelo en cascada con facilidad.

La mejora del proceso como tarea del proceso. La incorporación de las actividades de ingeniería y soporte como tareas específicas del proyecto como se propone en 6.5.4 permite modelar explícitamente el *trade-off* entre costes y beneficios de la mejora de procesos.

La orientación a agentes. El modelo presentado en 6.6.5 está orientado a la búsqueda en el espacio de las estrategias de priorización de las actividades. Parece, pues, el más adecuado para simular las reglas heurísticas descritas en el capítulo 5.

El formalismo DEVS. EL formalismo referenciado en 6.3.9 permite la implementación del modelo de forma natural, encapsulada y escalable. Existen diversas herramientas para la edición de modelos y la simulación, basadas en Java y en Python, que son de acceso libre. A pesar de no contar con sofisticadas GUI ni herramientas de presentación de resultados, son transparentes a diferencia de otras herramientas que permiten la simulación híbrida por lo que el modelador conserva el control sobre el modelo en todo momento. La orientación a objeto de dichas herramientas facilitan además la composición de modelos jerárquicos anidados.

De la integración de estos elementos y de la explotación y aprovechamiento de muchos otros detalles y visiones sobre problemas relevantes que aparecen en la literatura examinada, cabe esperar que será posible construir el modelo que se busca.

Aparte de resolver la integración de todos estos elementos, quedan pendientes otras tareas entre las que la más relevante para el inmediato futuro está la de encontrar un marco de *métricas de proyecto* adecuadas para el problema. La aproximación prevista se basa en el denominado sistema de valor alcanzado - *earned value management* (EVM) - que es una práctica estándar en la gestión de proyectos bajo contrato, como es el caso que nos ocupa.

Capítulo 7

Conclusiones

Este trabajo de investigación pretende responder a las siguientes preguntas:

a) ¿Cómo se modela, o se puede modelar, específicamente un entorno de gestión multiproyecto desde el punto de vista normativo más allá de agregar todos los proyectos en un único *metaproyecto*?

b) ¿Cómo se toman las decisiones de asignación de recursos a actividades y proyectos para cumplir plazos y minimizar costes?

c) ¿Cómo se tienen en cuenta los factores de riesgo e incertidumbre en estas propuestas normativas?

d) ¿Qué propuestas hay de modelos de simulación de los procesos software que tengan en cuenta el carácter a la vez continuo y discreto de los mismos?

e) ¿Cómo se pueden implementar las técnicas del punto 3 anterior en esos modelos?

f) ¿En qué medida la mejora de procesos puede reflejarse y evaluarse con esos modelos?

7.1. Modelo del entorno multiproyecto

En el capítulo 2 se ha considerado el problema de la gestión multiproyecto en dos marcos de referencia. El primero clasifica los entornos multiproyecto en términos de *variabilidad* y *dependencia*. El segundo diferencia entre los niveles de decisión *estratégico*, *táctico* y *operacional*.

La conclusión es que organización dedicada al desarrollo de *software* a medida es típicamente una organización por proyectos, por tanto *orientada a reducir la dependencia entre proyectos a nivel operacional asumiéndola en el nivel táctico*. Es decir, en la medida de lo posible se intenta independizar en términos de recursos cada proyecto individual. Esta independencia será siempre relativa, en función del recurso del que se trate. Habrá por tanto recursos que se pueden compatibilizar entre proyectos y recursos exclusivos de cada uno de ellos.

La decisión de aceptar proyectos se ha dejado fuera del ámbito del problema en estudio. Se deriva de criterios estratégicos y de la acción comercial. A los efectos que de este trabajo, los proyectos "aparecen" previamente "marcados" con fechas límite que pueden deberse a condiciones internas o externas.

Los objetivos del estudio entonces se plantean como:

- En el nivel *táctico*; asignar a los proyectos los recursos necesarios para asegurar el cumplimiento de plazos de entrega. Estos recursos deben obtenerse empleando la capacidad productiva de la empresa de la forma más económica posible y dentro de los requisitos de calidad de los proyectos. Se trata de un problema de *planificación táctica de capacidad* que debe considerar, aparte de las restricciones temporales, las asignaciones previas y el coste de los recursos.
- En el nivel *operacional*; programar los proyectos dentro de los márgenes impuestos desde el nivel táctico de forma que se pueda hacer frente a la variabilidad propia de cada uno de ellos. Se trata de un problema de *programación de proyec-*

tos con restricción de recursos en entorno de incertidumbre y con ventanas de tiempo. Según el grado de dependencia mutua estaremos hablando de proyectos *individuales* o *multiproyecto*.

7.2. La asignación de los recursos a los proyectos y la programación de los mismos

Como se he expuesto en el punto anterior, se ha descompuesto el problema en dos, uno táctico y otro operacional.

7.2.1. El problema táctico

En el capítulo 3 se ha estudiado el problema de dimensionar los equipos necesarios para abordar los proyectos de manera que se asegure que se cumplen los plazos. Se trata de un problema *restringido por el tiempo*.

Los proyectos se representan como una serie actividades agregadas de manera que la proporción entre los distintos recursos que emplea cada una de ellas es constante. El modelo admite que los recursos sean compartidos por varios proyectos, algo que a este nivel de análisis se corresponde con la realidad del problema en estudio.

En los proyectos de *software* a medida, existen hitos externos caracterizados por entregables que deben hacerse llegar al cliente. Esto se representa de forma adecuada con *las ventanas de tiempo admisibles*.

El modelo elegido presenta algunos problemas para el caso de estudio:

- El modelo asume una productividad lineal, es decir, que el tiempo de desarrollo es inversamente proporcional al número de personas empleadas en las tareas, algo que no es evidente en el caso del desarrollo de *software*.

- El modelo es estático, en el sentido de que la aparición de un nuevo proyecto obliga a una reprogramación. Sin embargo, el ritmo de aparición de nuevos proyectos hace que no sea computacionalmente excesivamente costosa la reprogramación.

Por tanto, la propuesta es emplear el modelo presentado en el apartado 3.3.1, estudiando el impacto que puede tener la linealidad y explorando el espacio de las soluciones a partir de los algoritmos de generación de secuencias.

La inclusión de un nuevo proyecto generará una nueva solución factible simplemente añadiendo recursos extraordinarios para cumplir los plazos. Este es el punto de partida para una nueva exploración ahora con el nuevo proyecto.

7.2.2. El problema operacional

El problema operacional se ha definido como un problema de *programación de proyectos con restricción de recursos y ventanas de tiempo*.

La aplicación de las técnicas *multiproyecto* al problema operacional no se considera adecuada teniendo en cuenta la estrategia de descomposición adoptada. En el nivel operacional, los equipos de cada proyecto han sido definidos desde la planificación táctica de capacidad. En condiciones normales, un director de proyecto tiene como variable de decisión la de asignar los componentes de su equipo a las distintas tareas de su proyecto.

El objetivo es minimizar la duración total del proyecto individual. Para ello se propone la aplicación de *heurísticas* basadas en *reglas de prioridad dinámicas*, generando secuencias en *serie* y representando las soluciones en la forma *lista de actividades*.

A pesar de lo anterior, desde el punto de vista de la supervisión desde el nivel táctico es preciso contar con una función objetivo más global. Se propone para ello un seguimiento de la máxima desviación de todo el conjunto de proyectos simultáneos.

7.3. El tratmiento del riesgo

En el capítulo 5 se ha estudiado el riesgo desde dos puntos de vista: el más corriente en términos de la gestión de proyectos y algunas aportaciones desde otros dominios entre las que destaca la metodología de la *design structure matrix* -DSM- originada en el campo del desarrollo de productos que permite captar y tratar los ciclos y las iteraciones, facetas que comparten los procesos de desarrollo de producto y los procesos *software*.

Con un análisis más detallado de la incidencia de la variabilidad y la dependencia en los dos niveles, táctico y operacional, se llega a la conclusión de que:

- Al nivel táctico, las estrategias son las de obtener programaciones *robustas*, es decir, capaces de absorber las variaciones sin grandes desviaciones sobre las fechas de terminación programadas inicialmente.Se ha identificado un modelo basado en la lógica *fuzzy* que contribuye a la robustez y que admite una representación dinámica en forma de *reglas de prioridad*.
- Al nivel operacional, la opción más adecuada son las estrategias *predictivas-reactivas* orientadas a la flexibilidad, es decir, a la recuperación de programaciones que se ven alteradas por alguna disrupción. Dentro de estas, la aproximación denominada *secuenciación contingente* parece la más adecuada. Al igual que la anterior, es una solución orientada a la simulación para el apoyo a la decisión.

7.4. Modelos híbridos

En el capítulo 6 se han revisado diversos modelos, continuos, discretos e híbridos, elaborados desde diversos enfoques y puntos de vista para simular el proceso *software*. La simulación híbrida es una tecnología de uso general en muchos dominios por lo que existen muchas soluciones.

En el problema que aquí se está planteando, la cuestión crítica parece ser la de no “perder” el efecto de la realimentación en el

paso de contínuo a discreto y viceversa.

Desde el punto de vista conceptual, el problema es identificar que procesos tienen lugar de forma contínua y qué procesos se producen de forma discreta. Esto está ampliamente documentado en la literatura, Los mecanismos que regulan la productividad, la tasa de errores, el efecto de la presión de los plazos, son típicamente continuos. El ciclo de vida de las tareas y los entrgables, sin embargo se manifiesta de forma discreta. Y desde luego la información y las decisiones son discretas.

Desde el punto de vista de la implementación el problema lo resuelve la utilización de la integración numérica para la simulación continua. Es, por tanto, un problema de intervalos de tiempo. Salvo que estemos en presencia de mecanismos no lineales que den lugar a fenómenos de bifurcación, los métodos de integración numérica pueden reslver el problema. Queda como tarea para el posterior desarrollo de este trabajo investigar analíticamente los modelos que resulten para caracterizar, aunque sea aproximadamente, el espacio de estados e identificar posibles anomalías.

La respuesta a la pregunta de este apartado incluye también la respuesta al siguiente. Se ha identificado un formalismo de modelado, el denominado DEVS - *discrete event system specification*, que permite la implementación del modelo de forma natural, encapsulada y escalable. Las herramientas para la edición de modelos y la simulación son de acceso libre. Existen antecedentes en la literatura de su aplicación a la simulación de procesos *software* con buenos resultados.

7.5. La implementación de las técnicas de la gestión de proyectos en los modelos de simulación

Formalmente, ambos ámbitos de decisión - el táctico y el operacional - se basan en reglas de prioridad dinámicas. La representación de las reglas de prioridad es, de todas las técnicas de solución heurística presentadas, la más sencilla.

El problema es modelar el proceso por el cual los decisores estiman los parámetros con los que calculan las reglas de prioridad. Máxime, teniendo presente que la incertidumbre da lugar a que sus estimaciones no sean exactas.

La representación del decisor siguiendo el modelo de agentes permite implementar ese proceso. Los agentes, de acuerdo con las estrategias que se modelarán, reúnen información aproximada sobre el estado del sistema y formulan reglas de prioridad de las que se desprenden las decisiones de asignación de recursos y de secuenciación de tareas.

Esto se pude modelar a ambos niveles, al téctico y al operacional. El decisor táctico asigna equipos a una cola de proyectos. Eventualmente decide adquirir nuevos recursos, sustituyendose así el mecanismo continuo de los modelos estrictamente continuos por paquetes de contratación que se deciden cuando se establecen los planes tácticos. Los recursos evolucionan de forma continua: tanto en su calidad mediante procesos de aprendizaje como a través de una tasa de abandono "natural".

El decisor táctico también puede decidir reasignar recursos de un proyecto a otro. Para ello tiene unos umbrales a partir de los cuales se considera esta opción.

El decisor operacional gestiona sus recursos y hace frente a las disrupciones en su programación readaptando ésta o reasignando los recursos de que dispone entre tareas para asegurar que se cumplen los plazos. La aproximación es la de la *secuenciación contingente*: "qué ocurre si ...?"

En la literatura hay modelos que permiten representar estos procesos de decisión y sus consecuencias. El heurístico *squeaky wheel* expuesto en 6.6.5 puede explorar el espacio de las estrategias para implementar de forma eficiente la secuenciación contigente.

La integración de ambos niveles y la representación de los procesos de desarrollo incremental, con sus ciclos entre fases se puede implementar la modo que se presenta en 6.4.2. El mecanismo básico que gobierna la evolución al nivel más elemental, previa adaptación, puede ser el de los modelos reducidos expuesto en sec:reducido.

La mejora de procesos y la simulación Las referencias encontradas a la simulación de procesos *software* en relación con la mejora de esos mismos procesos son abundantes. Tanto para el CMM y CMMI (las más) como para la norma ISO-IEC 15504 (menos).

La aproximación al problema que se ha considerado más interesante, por las razones que se exponen en 6.5.4 es la de modelar explícitamente las tareas que la mejora impone. El modelado explícito de estas tareas tiene dos virtualidades:

- Proporciona mayor precisión en el cálculo del esfuerzo requerido que no es sólo el trabajo directo de análisis y construcción sino que incluye, además de la propia gestión, las tareas anexas a ésta que corresponden a lo que genéricamente se llama la sobrecarga de comunicación (*communication overhead*)
- Arroja un mapa efectivo para la mejora de los procesos al permitir contrastar el coste de la misma con el beneficio esperable

Dado que la progresión en la madurez de la organización desarrolladora avanza en el sentido de mejorar la calidad de la gestión, al menos en los tres primeros nieveles del marco de referencia del CMMI, esto se modelará explícitamente en la mejora de los mecanismos de decisión reflejados en el apartado anterior.

7.6. Desarrollos posteriores y tareas pendientes

La principal tarea que aparece de manera más inmediata es la integración de estos elementos en un único modelo, lo cual no parece un trabajo sencillo a primera vista.

En segundo lugar está la de encontrar un marco de *métricas de proyecto* adecuadas para el problema. La aproximación prevista se basa en el denominado sistema de valor alcanzado - *earned value management* (EVM) - que es una práctica estándar en la gestión de proyectos bajo contrato, como es el caso que nos ocupa.

En tercer lugar está la implementación del modelo y su ajuste con la información disponible en el caso de estudio.

que De la integración de estos elementos y de la explotación y aprovechamiento de muchos otros detalles y visiones sobre problemas relevantes que aparecen en la literatura examinada, cabe esperar que será posible construir el modelo que se busca.

Aparte de resolver la integración de todos estos elementos, quedan pendientes otras tareas entre las que la más relevante para el inmediato futuro está la de encontrar un marco de *métricas de proyecto* adecuadas para el problema. La aproximación prevista se basa en el denominado sistema de valor alcanzado - *earned value management* (EVM) - que es una práctica estándar en la gestión de proyectos bajo contrato, como es el caso que nos ocupa.

Apéndice A

Curriculum Vitae

a) Formación académica

- Ingeniero Industrial. Universidad de de Sevilla, 1981-82.
- Periodo de Docencia. tercer Ciclo. Programa de Doctorado de Ingeniería de Organización. Universidad de Sevilla, 2001-02.

b) Experiencia Profesional

- 1981-1984. Profesor Ayudante de Clase Prácticas. ETS de Ingenieros Industriales, Universidad de Sevilla.
- 1984-1992. Ingeniero Consultor. Heredia Consultores Andalucía SA
- 1992-1993. Ingeniero de Proyecto. Guadaltel SL
- 1994-1996. Responsable de Área. Izquierda Unida - Convocatoria por Andalucía
- 1996- . Profesor Asociado. Departamento de Organización Industrial y Gestión de Empresas. Universidad de Sevilla.

c) Investigación

- Miembro del grupo de investigación Ingeniería de Organización (TEP 127) Universidad de Sevilla
- Colaborador en el proyecto "Taxonomía de modelos para la medición y evaluación de procesos software" (TIN2004-06689-C03-03)

- Colaborador de AICIA
- Colaborador del Insituto Andaluz de Tecnología (IAT)

d) Otras actividades relevantes

- 1998-2002. Miembro del Consejo de Administración de El Monte. Caja de Ahorros de Sevilla y Huelva
- 1996-2000. Miembro del Consejo de Administración de la RTVA
- 2000- . Director de la Fundación de Investigaciones Marxistas

Bibliografía

[Abd93a] Tarek K. AbdelHamid. Modeling the dynamics of software reuse: An integrating system dynamics perspective. In *Proceedings of the Sixth Workshop on Institutionalizing Software Reuse*, 1993. annote: incomplete.

[Abd93b] Tarek K. AbdelHamid. A multiproject perspective of single-project dynamics. *Journal of Systems and Software*, 22(3):151–165, sep 1993.

[ACC+04] Lerina Aversano, Gerardo Canfora, Giovanni Capasso, Giuseppe A. Di Lucca, and Corrado A. Visaggio. Introducing quality system in small and medium enterprises: An experience report. *Springer-Verlag Berlin Heidelberg*, 2004.

[ACLP04] G. Antoniol, A. Cimitile, G. A. Di Lucca, and M. Di Penta. Assessing staffing needs for a software maintenance project through queuing simulation. *IEEE Transactions on Software Engineering*, 30(1):43–58, 2004.

[AIG03] S. Anavi-Isakow and B. Golany. Managing multi-project environments through constant work-in-process. *International Journal of Project Management*, 21(1):9–18, 1 2003.

[AM91] Tarek K. AbdelHamid and S. E. Madnick. *Software Project Dynamics An Integrated Approach*. Prentice Hall, Englewood Cliffs NJ, 1991.

[AM01] J. Alcaraz and C. Maroto. A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operations Research*, 102(1-4):83–109, 2001.

[AMNS95] Paul S. Adler, Avi Mandelbaum, Viên Nguyen, and Elizabeth Schwerer. From project to process management: An empirically-based framework for analyzing product development time. *Management Science*, 41(3):458, 1995.

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows : theory, algorithms, and applications*. Prentice Hall, Englewood Cliffs, N.J, 1993.

[AMR03] Christian Artigues, Philippe Michelon, and Stephane Reusser. Insertion techniques for static and dynamic resource-constrained project scheduling. *European Journal of Operational Research*, 149(2):249–267, 9/1 2003.

[A.P03] Shibanov A.P. Finding the distribution density of the time taken to fulfill the gert network on the basis of equivalent simplifying transformations. *Automation and Remote Control*, 64:279–287(9), 2003.

[AP06] Elodie Adida and Georgia Perakis. A robust optimization approach to dynamic pricing and inventory control with no backorders. *Mathematical Programming*, 107(1):97–129, 06 2006.

[APH04] G. Antoniol, M. Di Penta, and M. Harman. A robust search-based approach to project management in the presence of abandonment, rework, error and uncertainty. In *Proceedings - 10th International Symposium on Software Metrics, METRICS 2004*, pages 172–183, 14 September 2004 through 16 September 2004 2004.

[AR00] Christian Artigues and François Roubellat. A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and

multiple modes. *European Journal of Operational Research*, 127(2):297–316, 12/1 2000.

[ARRRR02] Jesús S. Aguilar-Ruiz, José C. Riquelme, Daniel Rodríguez, and Isabel Ramos. Generation of management rules through system dynamics and evolutionary computation. *Springer-Verlag Berlin Heidelberg*, pages 615–628, 2002.

[ARRRT01] Jesús S. Aguilar-Ruiz, Isabel Ramos, José C. Riquelme, and Miguel Toro. An evolutionary approach to estimating software development projects. *Information and Software Technology*, 43(14):875–882, 12/15 2001.

[ASA06] Babak Abbasi, Shahram Shadrokh, and Jamal Arkat. Bi-objective resource-constrained project scheduling with robustness and makespan criteria. *Applied Mathematics and Computation*, 180(1):146–152, 9/1 2006.

[AT08] Osman Alp and Tarkan Tan. Tactical capacity management under capacity flexibility. *IIE Transactions*, 40(3):221, 2008.

[Bal03] Francisco Ballestín. Nuevos métodos de resolución del probelma de secuenciación de proyectos con recursos limitados, 2003.

[BD07] Jeroen Beliën and Erik Demeulemeester. On the trade-off between staff-decomposed and activity-decomposed column generation for a staff scheduling problem. *Annals of Operations Research*, 155(1):143–166, 11/22 2007.

[BDM+99] Peter Brucker, Andreas Drexl, Rolf Möhring, Klaus Neumann, and Erwin Pesch. Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41, 1/1 1999.

[BdOBW08] Milton Barreto, Marcio de O. Barros, and Claudia M. L. Werner. Staffing a software project:

A constraint satisfaction and optimization-based approach. *Computers & Operations Research*, In Press, Corrected Proof:2153, 2008.

[BDR02] C. Briand, E. Despontin, and François Roubellat. Scheduling with time lags and preferences: a heuristic. In *8th Workshop on Project Management and Scheduling.*, Valencia., 2002.

[Bel72] Richard E. Bellman. *Dynamic Programming*. Princeton University Press, 6 edition, 1972.

[BJN+98] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[BL03] K. Bouleimen and H. Lecocq. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. *European Journal of Operational Research*, 149(2):268–281, 9/1 2003.

[Boc96] F. F. Boctor. Resource-constrained project scheduling by simulated annealing. *International Journal of Production Research*, 34(8):2335–2351, 1996.

[Boe98] R. De Boer. Resource-constrained multi-project management, 1998.

[Bro02] Tyson R. Browning. Process integration using the design structure matrix. *Systems Engineering*, 5(3):180, 2002.

[BS99] R. De Boer and J. M. J. Schutten. Multi-project rough-cut capacity planning, 1999.

[BT96] Walid Belassi and Oya Iemeli Tukel. A new framework for determining critical success/failure factors in projects. *International Journal of Project Management*, 14(3):141–151, 6 1996.

[BTN00] Aharon Ben-Tal and Arkadi Nemirovski. Robust solutions of linear programming problems contami-

nated with uncertain data. *Mathematical Programming*, 88(3):411, 2000.

[BWT02] Marcio Oliveira Barros, Claudia Maria Lima Werner, and Guilherme Horta Travassos. A system dynamics metamodel for software process modeling. *Software Process: Improvement and Practice*, 7(3-4):161–172, 2002.

[BY06] Tyson R. Browning and Ali A. Yassine. Resource-constrained multi-project scheduling: Priority rule performance revisited, 2006.

[BZ06] Z. A. Banaszak and M. B. Zaremba. Project-driven planning and scheduling support for virtual manufacturing. *Journal of Intelligent Manufacturing*, 17(6):641–651, 2006.

[Car03] Mercedes Ruiz Carreira. Modelado y simulación para la mejora de los procesos software, 2003. Memoria de la Tesis Doctoral para optar al grado de Doctora en Informática por la Universidad de Sevilla presentada por Mercedes Ruiz Carreira ; Directores, Isabel Ramos Román, Miguel Toro Bonilla.; Tesis - Universidad de Sevilla. - Ejemplar no publicado.

[CB07] Marjorie J. Cooper and Charlene Spoede Budd. Tying the pieces together: A normative framework for integrating sales and project operations. *Industrial Marketing Management*, 36(2):173–182, 2 2007.

[CBK06] KeungSik Choi, DooHwan Bae, and TagGon Kim. An approach to a hybrid software process simulation using the devs formalism. *Software Process: Improvement and Practice*, 11(4):373–383, 2006.

[CDM+02] Kevin M. Calhoun, Richard F. Deckro, James T. Moore, James W. Chrissis, and John C. Van Hove. Planning and re-planning in project and production scheduling. *Omega*, 30(3):155–170, 6 2002.

[CDSC05] Rodrigo Cern, Juan C. Dueñas, Enrique Serrano, and Rafael Capilla. A meta-model for requirements engineering in system family context for software process improvement using cmmi. In Frank Bomarius and Seija Komi-Sirviø, editors, *PROFES; Product Focused Software Process Improvement, 6th International Conference, PROFES 2005, Oulu, Finland, June 13-15, 2005, Proceedings; Lecture Notes in Computer Science*, volume 3547, pages 173–188. Springer, 2005.

[CE05] Soo-Haeng Cho and Steven D. Eppinger. A simulation-based process model for managing complex design projects. *IEEE Transactions on Engineering Management*, 52(3):316, 2005.

[CH08] Haadi Chtourou and Mohamed Haouari. A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling. *Computers & Industrial Engineering*, In Press, Corrected Proof:154, 2008.

[Cho05] Soo-Haeng Cho. A simulation-based process model for managing complex design projects. *IEEE Transactions on Engineering Management*, 52(3):316–328, 2005.

[Chr99] Alan M. Christie. Simulation in support of cmm-based process improvement. *Journal of Systems and Software*, 46(2-3):107–112, 4/15 1999.

[CLT02] Zhi-Long Chen, Shanling Li, and Devanath Tirupati. A scenario-based stochastic programming approach for technology and capacity planning. *Computers & Operations Research*, 29(7):781–806, 6 2002.

[Cor03] Standish Corporation. Chaos chronicles version 3.0. Technical report, The Standish Group, 2003.

[CR06] A. Ceselli and G. Righini. A branch-and-price algorithm for the multilevel generalized assignment problem. *Operations Research*, 54(6):1172–1184, 2006.

[CRL04] Jaein Choi, Matthew J. Realff, and Jay H. Lee. Dynamic programming in a heuristically confined state space: a stochastic resource-constrained project scheduling application. *Computers Chemical Engineering*, 28(6-7):1039–1058, 6/15 2004.

[Day00] Julian Day. Software development as organizational conversation: analogy as a systems intervention. *Systems Research and Behavioral Science*, 17(4):349–358, 2000.

[DER98] B. Dodin, A. A. Elimam, and E. Rolland. Tabu search in audit scheduling. *European Journal of Operational Research*, 106(2-3):373–392, 1998.

[DF00] José Javier Dolado and Luis Fernández. *Medición para la gestión en la Ingeniería del software*. RaMa, Madrid, 2000.

[DH02] Erik L. Demeulemeester and Willy Herroelen. *Project scheduling*, volume 49. Kluwer Academic Publishers, Boston, 2002.

[DI01] Paolo Donzelli and Giuseppe Iazeolla. Hybrid simulation modelling of the software process. *Journal of Systems and Software*, 59(3):227–235, 12/15 2001.

[Don02] A. J. M. Donaldson. A case narrative of the project problems with the Denver Airport Baggage Handling System (DABHS). Technical Report Technical Report TR 2002-01, Software Forensics Centre, 2002.

[DRLV06] D. Debels, B. De Reyck, R. Leus, and M. Vanhoucke. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. *European Journal of Operational Research*, 169(2):638–653, 2006.

[dSP05] Leandro Dias da Silva and Angelo Perkusich. Composition of software artifacts modelled using colored petri nets. *Science of Computer Programming*, 56(1-2):171–189, 4 2005.

[DV06] D. Debels and M. Vanhoucke. The electromagnetism meta-heuristic applied to the resource-constrained project scheduling problem. Technical report, 2006.

[dVDHL05] Stijn Van de Vonder, Erik Demeulemeester, Willy Herroelen, and Roel Leus. The use of buffers in project management: The trade-off between stability and makespan. *International Journal of Production Economics*, 97(2):227–240, 8/18 2005.

[EB00] Khaled El Emam and Andreas Birk. Validating the iso/iec 15504 measures of software development process capability. *Journal of Systems and Software*, 51(2):119–149, 4/15 2000.

[Ebe89] R. L. Eberlein. Simplification and understanding of. models. *System Dynamics Review*, 5(1):59–68, 1989.

[EKR95] Clarence Ellis, Karim Keddara, and Grzegorz Rozenberg. Dynamic change within workflow systems, 1995.

[Elm70] Salah Eldin Elmaghraby. *Some network models in management science*, volume 29. Springer-Verlag, Berlin etc., 1970.

[Elm05] Salah E. Elmaghraby. On the fallacy of averages in project risk management. *European Journal of Operational Research*, 165(2):307–313, 9/1 2005.

[FCMA08] H. Fei, C. Chu, N. Meskens, and A. Artiba. Solving surgical cases assignment problem by a branch-and-price approach. *International Journal of Production Economics*, 112(1):96–108, 3 2008.

[FL07] Ilya M. Fishman and Raymond E. Levitt. The virtual design team and quantum tm: Comparison of project organization models. Technical report, 2007.

[FMK01] S. Fujii, H. Morita, and T. Kanawa. Resource constrained planning of multiple projects with separable

activities. *JSME International Journal, Series C: Mechanical Systems, Machine Elements and Manufacturing*, 44(1):261–266, 2001.

[Fre01] Michael L. Fredley. A de-composition approach for the multi-modal, resource-constraint, multi-project scheduling problem with generalized precedences and expediting resources, 2001.

[FS00] Sydney Finkelstein and Shade H. Sanford. Learning from corporate mistakes:: The rise and fall of iridium. *Organizational Dynamics*, 29(2):138–148, 11 2000.

[FSLS65] Richard Phillips Feynman, Matthew Sands, Robert B. Leighton, and Matthew Sands. *The Feynman lectures on physics*. Reading, Mass, 1965. III, Quantum mechanics / Richard P. Feynman, Robert B. Leighton, Matthew Sands.; v. : il. ; 28 cm; Indices.; Quantum mechanics.

[Gar01] Roland Gareis. The project-oriented society: A new creation to ensure international competitiveness. In *IPMA International Symposium and NORDNET 2001*, 2001.

[Gav04] M. H. K. Gavareshki. New fuzzy gert method for research projects scheduling, 2004.

[GC03] Alain Guinet and Sondes Chaabane. Operating theatre planning. *International Journal of Production Economics*, 85(1):69–81, 7/11 2003.

[gdf02] GOTHA groupe de flexibilité. Flexilibilité et robustesse en ordonnancement. *Bulletin de la ROADEF*, 8:10–12, 2002.

[GGGL03] Dimitri Golenko-Ginzburg, Aharon Gonik, and Zohar Laslo. Resource constrained scheduling simulation model for alternative stochastic network projects. *Mathematics and Computers in Simulation*, 63(2):105–117, 6/10 2003.

[GMR08] J. F. Gonçalves, J. J. M. Mendes, and M. G. C. Resende. A genetic algorithm for the resource constrained multi-project scheduling problem. *European Journal of Operational Research*, In Press, Corrected Proof, 2008.

[Gol97] Eliyahu M. Goldratt. *Critical chain*. North River Press, Great Barrington, MA, 1997.

[GS05] Noud Gademann and Marco Schutten. Linear-programming-based heuristics for project capacity planning. *IIE Transactions*, 37(2):153–165, 02 2005.

[Hal80] Peter Hall. *Great Planning Disasters*. Weidenfeld and Nicolson, London, 1980.

[Han01] E. W. Hans. Resource loading by branch-and-price techniques, 2001.

[Har98] S. Hartmann. A competitive genetic algorithm for resource-constrained project scheduling. *Naval Research Logistics*, 45(7):733–750, 1998.

[HBH+06] L. Huang, Barry Boehm, H. Hu, J. Ge, J. Lu, and C. Qian. Applying the value/petri process to erp software development in china. In *28th International Conference on Software Engineering 2006, ICSE '06*, volume 2006, pages 502–511, 20 May 2006 through 28 May 2006 2006.

[HD98] S. Hartmann and A. Drexl. Project scheduling with multiple modes: A comparison of exact algorithms. *Networks*, 32(4):283–297, 1998.

[Her05] Willy Herroelen. Project scheduling-theory and practice. *Production and Operations Management*, 14(4):413, Winter 2005.

[HFC+01] Dan X. Houston, Susan Ferreira, James S. Collofello, Douglas C. Montgomery, Gerald T. Mackulak, and Dan L. Shunk. Behavioral characterization: finding and using the influential factors in software process simulation models. *The Journal of Systems*

and Software, 59(3):259–270, dec 2001. CODEN: JSSODM.

[HGdVZ02a] E. W. Hans, A. J. R. M. Gademann, S. L. Van de Velde, and W. H. M. Zijm. Resource loading by branch-and-price techniques: models and algorithms, 2002.

[HGdVZ02b] E. W. Hans, A. J. R. M. Gademann, S. L. Van de Velde, and W. H. M. Zijm. Rough-cut capacity planning by branch-and-price techniques, 2002.

[HH00] Peter Henderson and Yvonne Howard. Simulating a process strategy for large scale software development using systems dynamics. *Software Process: Improvement and Practice*, 5(2-3):121–131, 2000.

[HHLW07] E. W. Hans, Willy Herroelen, R. Leus, and G. Wullink. A hierarchical approach to multi-project planning under uncertainty. *Omega*, 35(5):563–577, 10 2007.

[HK05] C. C Hsu and D. S. Kim. A new heuristic for the multi-mode resource investment problem. *Journal of the Operational Research Society*, 56(4):406–413, 2005.

[HL89] Steven T. Hackman and Robert C. Leachman. Aggregate model of project-oriented production. *IEEE Transactions on Systems, Man and Cybernetics*, 19(2):220–231, 1989.

[HL04a] Willy Herroelen and Roel Leus. The construction of stable project baseline schedules. *European Journal of Operational Research*, 156(3):550–565, 8/1 2004.

[HL04b] Willy Herroelen and Roel Leus. Robust and reactive project scheduling: a review and classification of procedures. *International Journal of Production Research*, 42(8):1599–1620, 04/15 2004.

[HLD02] Willy Herroelen, Roel Leus, and Erik Demeulemeester. Critical chain project scheduling: Do not oversimplify. *Project Management Journal*, 33(4):48, 12 2002.

[HMC01] Dan X. Houston, Gerald T. Mackulak, and James S. Collofello. Stochastic simulation of risk factor potential effects for software development risk management. *Journal of Systems and Software*, 59(3):247–257, 12/15 2001.

[HRD98] Willy Herroelen, B. De Reyck, and E. Demeulemeester. Resource-constrained project scheduling: A survey of recent developments. *Computers and Operations Research*, 25(4):279–302, 1998.

[HS96] S. Hartmann and A. Sprecher. A note on "hierarchical models for multi-project planning and scheduling". *European Journal of Operational Research*, 94(2):377–383, 1996.

[HS08] Seungchul Ha and Hyo-Won Suh. A timed colored petri nets modeling for dynamic workflow in product development process. *Computers in Industry*, 59(2-3):193–209, 3 2008.

[Ins04] Project Management Institute. *Guía de los fundamentos de la Dirección de proyectos [Project Management Institute]*. Project Management Institute, Newtown Square, Pa., 2004.

[JDSR08] B. Jarboui, N. Damak, P. Siarry, and A. Rebai. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. *Applied Mathematics and Computation*, 195(1):299–308, 1/15 2008.

[JKCR00] V. A. Jeetendra, O. V. Krishnaiah Chetty, and J. Prashanth Reddy. Petri nets for project management and resource levelling. *The International Journal of Advanced Manufacturing Technology*, 16(7):516–520, 06/21 2000.

[JMR+01] J. Josefowska, M. Mika, R. Rozycki, G. Waligora, and J. Weglarz. Simulated annealing for multi-mode resource-constrained project scheduling. *Annals of Operations Research*, 102(1-4):137–155, 2001.

[JP05] David Joslin and William Poole. Agent-based simulation for software project planning. In *Winter Simulation Conference*, pages 1059–1066, 2005.

[KC06] S. Kumanan and O. V. Krishnaiah Chetty. Estimating product development cycle time using petri nets. *The International Journal of Advanced Manufacturing Technology*, 28(1):215–220, 02/13 2006.

[KG07] Konstantinos G. Kouskouras and Andreas C. Georgiou. A discrete event simulation model in the case of managing a software project. *European Journal of Operational Research*, 181(1):374–389, 8/16 2007.

[KH06] Rainer Kolisch and Sönjnke Hartmann. Experimental investigation of heuristics for resource-constrained project scheduling: An update. *European Journal of Operational Research*, 174(1):23–37, 10/1 2006.

[KHY06] H. P Kao, B. Hsieh, and Y. Yeh. A petri-net based approach for scheduling and rescheduling resource-constrained multiple projects. *Journal of the Chinese Institute of Industrial Engineers*, 23(6):468–477, 2006.

[KJR06] S. Kumanan, G. Jegan Jose, and K. Raja. Multi-project scheduling using an heuristic and a genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, 31(3):360–366, 11/15 2006.

[KL05] Soulaymane Kachani and Jerome Langella. A robust optimization approach to capital rationing and capital budgeting. *Engineering Economist*, 50(3):195–229, 2005.

[KLRW01] G. Kahen, M. M. Lehman, J. F. Ramil, and P. Wernick. System dynamics modelling of software evolution processes for policy investigation: Approach and example. *Journal of Systems and Software*, 59(3):271–281, 12/15 2001.

[KN02] Kenzo Kurihara and Nobuyuki Nishiuchi. Efficient monte carlo simulation method of gert-type network for project management. *Computers & Industrial Engineering*, 42(2-4):521–531, 4/11 2002.

[Kol96] Rainer Kolisch. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research*, 90(2):320–333, 1996.

[Kol00] Rainer Kolisch. *Make-to-order assembly management*. Springer, New York, 2000.

[KP01] Rainer Kolisch and R. Padman. An integrated survey of deterministic project scheduling. *Omega*, 29(3):249–272, 6 2001.

[Kum98] A. Kumar. Use of petri nets for resource allocation in projects, 1998.

[Kur06] K. Kurihara. Branching probabilities planning of stochastic network for project duration planning, 2006.

[KWDK06] Hsing-Pei Kao, Brian Wang, James Dong, and Kuo-Cheng Ku. An event-driven approach with makespan/cost tradeoff analysis for project portfolio scheduling. *Computers in Industry*, 57(5):379–397, 06 2006.

[KYY+05] K. Kim, Y. Yun, J. Yoon, M. Gen, and G. Yamazaki. Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling. *Computers in Industry*, 56(2):143–160, 2005.

[Lar60] Juan Larrañeta. *Programación y control de proyectos unitarios*. Universidad de Sevilla, E.T.S. de Ingenieros Industriales, Sevilla, 1992 1960.

[Lar87] Juan Larrañeta. *Programación lineal y grafos*, volume 1. Universidad de Sevilla, Sevilla, 1987.

[LCM04] R. Liao, Q. X Chen, and N. Mao. Genetic algorithm for resource-constrained project scheduling.

Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS, 10(7), 2004.

[LCWB08] Jun Lin, Kah Hin Chai, Yoke San Wong, and Aarnout C. Brombacher. A dynamic model for managing overlapped iterative product development. *European Journal of Operational Research*, 185(1):378–392, 2/16 2008.

[LJF04] Xiaoxia Lin, Stacy L. Janak, and Christodoulos A. Floudas. A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers & Chemical Engineering*, 28(6-7):1069–1085, 6/15 2004.

[LK02] Christoph H. Loch and Stylianos Kavadias. Dynamic portfolio selection of npd programs using marginal returns. *Management Science*, 48(10):1227–1241, October 1 2002.

[LKR02] M. M. Lehman, G. Kahen, and J. F. Ramil. Behavioural modelling of long-lived evolution processes–some issues and an example. *Journal of Software Maintenance and Evolution: Research and Practice*, 14:335–351, 2002.

[LM04] Bengee Lee and James Miller. Multi-project management in software engineering using simulation modelling. *Software Quality Journal*, 12:59–82, 2004.

[LN94] E. V. Levner and A. S. Nemirovsky. A network flow algorithm for just-in-time project scheduling. *European Journal of Operational Research*, 79(2):167–175, 12/8 1994.

[LO] Luong Duc Long and Ario Ohsato. Fuzzy critical chain method for project scheduling under resource constraints and uncertainty. *International Journal of Project Management*, In Press, Corrected Proof.

[LOL88] Juan Larrañeta, Luis Onieva, and Sebastián Lozano. *Métodos modernos de gestión de la producción*, volume 122. Alianza, Madrid, 1988.

[LTB06] Antonio Lova, Pilar Tormos, and Federico Barber. Multi-mode resource constrained project scheduling: Scheduling schemes, priority rules and mode selection rules. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 30:69–86, 2006.

[LTN06] H. C. Lau, R. Thangarajoo, and K. M. Ng. A hybrid MIP/heuristic model for experience based driver assignment. In *8th IEEE International Conference of Tools with Artificial Intelligence*, pages 407–415, 2006.

[LWT01] S. X Liu, M-G Wang, and J. F Tang. Optimization algorithms for solving resource-constrained project scheduling problem: A review. *Kongzhi yu Juece/Control and Decision*, 16(SUPPL.):647–651, 2001.

[MBCDH06] Harvey Maylor, Tim Brady, Terry Cooke-Davies, and Damian Hodgson. From projectification to programmification. *International Journal of Project Management*, 24(8):663–674, 11 2006.

[ME78] Joseph J. Moder and Salah Eldin Elmaghraby. *Handbook of operations research.* Van Nostrand Reinhold, New York etc., 1978.

[Mer06] Jack R. Meredith. *Project management : a managerial approach.* Wiley, New York ; Chichester, 2006.

[MGR08] J. J. M. Mendes, J. F. Gonçalves, and M. G. C. Resende. A random key based genetic algorithm for the resource constrained project scheduling problem. *Computers & Operations Research*, In Press, Corrected Proof:2153, 2008.

[MHSZ04] T. McBride, B. Henderson-Sellers, and D. Zowghi. Project management capability levels: an empirical study. In B. Henderson-Sellers, editor, *11th Asia-Pacific Software Engineering Conference*, pages 56–63, 2004. ID: 1.

[MRGT08] María Moreno, Isabel Ramos, Francisco J. García, and Miguel Toro. An association rule mining method for estimating the impact of project management policies on software quality, development time and effort. *Expert Systems with Applications*, 34(1):522–529, 1 2008.

[MS00] Rolf H. Mohring and Frederik Stork. Linear preselective policies for stochastic project scheduling. *Mathematical Methods of Operations Research*, 52(3):501, 2000.

[MSSU03] Rolf H. Mohring, Andreas S. Schulz, Frederik Stork, and Marc Uetz. Solving project scheduling problems by minimum cut computations. *Management Science*, 49(3):330–350, March 1 2003.

[MT00] Ray Madachy and Denton Tarbet. Case studies in software process modeling with system dynamics. *Software Process: Improvement and Practice*, 5:133–146, 2000.

[MV06] Jürgen Münch and Matias Vierimaa, editors. *Product-Focused Software Process Improvement, 7th International Conference, PROFES 2006, Amsterdam, The Netherlands, June 12-14, 2006, Proceedings*, volume 4034 of *Lecture Notes in Computer Science*. Springer, 2006.

[MWW08] Marek Mika, Grzegorz Waligóra, and Jan Woglarz. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. *European Journal of Operational Research*, 187(3):1238–1250, 6/16 2008.

[MYB07] Christoph Meier, Ali A. Yassine, and Tyson R. Browning. Design process sequencing with competent genetic algorithms. *Transactions of the ASME*, 129:566, 2007.

[Nag02] M.Ñagai. Project duration planning method based on the combination use of genetic algorithm and monte carlo simulation, 2002.

[NC03] Rita Nienaber and Elsabe Cloete. A software agent framework for the support of software project management. *Proceedings of SAICSIT*, pages 16–23, 2003.

[NM01] José Niño-Mora. *Encyclopedia of optimization*, volume V, chapter Stochastic scheduling, pages 367–372. Kluwer Academic Publishers, 2001.

[NM04] José Niño-Mora. Restless bandit marginal productivity indices ii: Multiproject case and scheduling a multiclass make-to-order/-stock m/g/1 queue. Technical report, Universidad Carlos III, Departamento de Estadística y Econometría, Feb 2004.

[NS98] K.Ñeumann and C. Schwindt. A capacitated hierarchical approach to make-to-order production. *Journal Europeen des Systemes Automatises*, 32(4):397–413, 1998.

[NSZ03] Klaus Neumann, C. Schwindt, and J. Zimmermann. *Project scheduling with time windows and scarce resources*. Springer Verlag, Heidelberg, 2003.

[NZBS07] M.Ñonaka, L. Zhu, M. A. Babar, and M. Staples. Project cost overrun simulation in software product line development. In *8th International Conference on Product-Focused Software Process Improvement, PROFES 2007*, volume 4589 LNCS, pages 330–344, 2 July 2007 through 4 July 2007 2007.

[Pad02a] F. Padberg. Using process simulation to compare scheduling strategies for software projects, 2002.

[Pad02b] Frank Padberg. A discrete simulation model for assessing software project scheduling policies. *Software Process: Improvement and Practice*, 7(3-4):127–139, 2002.

[Pad03] F. Padberg. A software process scheduling simulator, 2003.

[Pad04a] F. Padberg. Computing optimal scheduling policies for software projects, 2004.

[Pad04b] F. Padberg. Linking software process modeling with markov decision theory, 2004.

[Pad05] F. Padberg. On the potential of process simulation in software project schedule optimization, 2005.

[PAER07] Dietmar Pfahl, Ahmed Al-Emran, and GÃÂ$\frac{1}{4}$nther Ruhe. A system dynamics simulation model for analyzing the stability of software release plans. *Software Process: Improvement and Practice*, 12(5):475–490, 2007.

[PAM04] Mireille Palpant, Christian Artigues, and Philippe Michelon. Lssper: Solving the resource-constrained project scheduling problem with large neighbourhood search. *Annals of Operations Research*, 131(1):237–257, 10/08 2004.

[Pau95] Mark C. Paulk. The evolation of the sei's capacity maturity model for software. *Software Process: Improvement and Practice*, 1(1):3–15, 1995.

[PHC] Nai-Hsin Pan, Po-Wen Hsaio, and Kuei-Yen Chen. A study of project scheduling optimization using tabu search algorithm. *Engineering Applications of Artificial Intelligence*, In Press, Corrected Proof.

[PL00a] D. Pfahl and K. Lebsanft. Using simulation to analyse the impact of software requirement volatility on project performance. *Information and Software Technology*, 42(14):1001–1008, 11/15 2000.

[PL00b] Dietmar Pfahl and Karl Lebsanft. Knowledge acquisition and process guidance for building system dynamics simulation models : An experience report from software industry. *International Journal of Software Engineering and Knowledge Engineering*, 10(4):487–510, 2000.

[PM90] Jeffrey K. Pinto and Samuel J. Mantel. The causes of project failure. *IEEE Transactions on Engineering Management*, 37(4):269–276, 1990.

[PMB99] Antony Powell, Keith Mander, and Duncan Brown. Strategies for lifecycle concurrency and iteration a

system dynamics approach. *Journal of Systems and Software*, 46(2-3):151–161, 4/15 1999.

[PP90] Jeffrey K. Pinto and John E. Prescott. Planning and tactical factors in the project implementation process. *Journal of Management Studies*, 27(3):305–327, 05 1990.

[PR02] Dietmar Pfahl and Günther Ruhe. Immos: a methodology for integrated measurement, modelling and simulation. *Software Process: Improvement and Practice*, 7(3-4):189–210, 2002.

[PS06] Georgia Perakis and Anshul Sood. Competitive multi-period pricing for perishable products: A robust optimization approach. *Mathematical Programming*, 107(1):295–335, 06 2006.

[PWW69] A. Alan B. Pritsker, Lawrence J. Watters, and Philip M. Wolfe. Multiproject scheduling with limited resources: a zero-one programming approach. *Management Science*, 16(1):93–108, 09 1969.

[Ram99] Isabel Ramos. Un nuevo enfoque en la gestion de proyectosde desarrollo de software, 1999.

[RCL99] Ioana Rus, James Collofello, and Peter Lakey. Software process simulation for reliability management. *The Journal of Systems and Software*, 46(2–3):173–182, apr 1999. annote: incomplete.

[RHB03] I. Rus, M. Halling, and S. Biffl. Supporting decision-making in software engineering with process simulation and empirical studies. *International Journal of Software Engineering and Knowledge Engineering*, 13(5):531–545, 2003.

[Ric01] Ita Richardson. Software process matrix: A small company spi model. *Software Process: Improvement and Practice*, 6:157–165, 2001.

[RKC01] J. Prashant Reddy, S. Kumanan, and O. V. Krishnaiah Chetty. Application of petri nets and a genetic algorithm to multi-mode multi-resource

constrained project scheduling. *The International Journal of Advanced Manufacturing Technology*, 17(4):305–314, 01/05 2001.

[RKP+99] D. Raffo, T. Kaltio, D. Partridge, K. Phalp, and Juan FernÃÂ¡ndez-Ramil. Empirical studies applied to software process models. *Empirical Software Engineering*, 4(4):353–369, 1999.

[RM00] David Raffo and Robert H. Martin. A model of the software development process using both continous and discrete models. *Software Process Improvement and Practice*, 5:147–157, 2000.

[Rou03] T. P. Rout. Iso/iec 15504 - evolution to an international standard. *Software Process Improvement and Practice*, 8(1):27–40, 2003. Cited By (since 1996): 2.

[RRK] M. Ranjbar, B. De Reyck, and F. Kianfar. A hybrid scatter search for the discrete time/resource tradeoff problem in project scheduling. *European Journal of Operational Research*, In Press, Corrected Proof.

[RRT01] Mercedes Ruiz, Isabel Ramos, and Miguel Toro. A simplified model of software project dynamics. *Journal of Systems and Software*, 59(3):299–309, 12/15 2001.

[RRT02] M. Ruiz, I. Ramos, and Miguel Toro. A dynamic integrated framework for software process improvement. *Software Quality Journal*, 10(2):181–194, 2002.

[RRT+03] Mercedes Ruiz, Isabel Ramos, Miguel Toro, Nuno David, Nuno David, Jaime Simão Sichman, and Helder Coelho. An integrated framework for simulation-based software process improvement; towards an emergence-driven software process for agent-based simulation. *Software Process: Improvement and Practice*, 9(2):81–93, mar 12 2004; 2003.

[RVM99] David M. Raffo, Joseph V. Vandeville, and Robert H. Martin. Software process simulation to

achieve higher cmm levels. *The Journal of Systems and Software*, 46(2–3):163–172, apr 1999. annote: incomplete.

[San00] U. Z. Sanal. A decision support system for fuzzy scheduling of software projects, 2000.

[Sca01] Walt Scacchi. *Process Models in Software Engineering*. Encyclopedia of Software Engineering. John Wiley and Sons, New York, 2001.

[SCFR06] Neil Smith, Andrea Capiluppi, and Juan Fernández-Ramil. Agent-based simulation of open source evolution. *Software Process: Improvement and Practice*, 11(4):423–434, 2006.

[SD07] Aaron J. Shenhar and Dov Dvir. Project management research–the challenge and opportunity. *Project Management Journal*, 38(2):93–99, 06 2007.

[SdOC03] Miguel A. Serrano, Carlos Montes de Oca, and Karina Cedillo. An experience on using the team software process for implementing the capability maturity model for software in a small organization. In *QSIC*, page 327. IEEE Computer Society, 2003.

[SG01] Friedrich Stallinger and Paul Grünbacher. System dynamics modelling and simulation of collaborative requirements engineering. *Journal of Systems and Software*, 59(3):311–321, 12/15 2001.

[Sho06] Y. Y Shou. Composite random sampling algorithm for scheduling concurrent projects. *Zhejiang Daxue Xuebao (Gongxue Ban)/Journal of Zhejiang University (Engineering Science)*, 40(2):344–347, 2006.

[SM07a] B. Skoaud and B. Marcineczyk. Ant colony optimization in project management. *Computer Assisted Mechanics and Engineering Sciences*, 14(4):745–752, 2007.

[SM07b] Susan A. Slotnick and Thomas E. Morton. Order acceptance with weighted tardiness. *Computers & Operations Research*, 34(10):3029–3042, 10 2007.

[SS03] C. S. Sung and S. H. Song. Branch-and-price algorithm for a combined problem of virtual path establishment and traffic packet routing in a layered communication network. *Journal of the Operational Research Society*, 54(1):72–82, 2003.

[Sta00] Friedrich Stallinger. Software process simulation to support ISO/IEC 15504 based software process improvement. *Software Process: Improvement and Practice*, 5(2-3):197–209, 2000.

[SV93] M. Grazia Speranza and Carlo Vercellis. Hierarchical models for multi-project planning and scheduling. *European Journal of Operational Research*, 64(2):312–325, 1/22 1993.

[SW06] E. F. Silva and R. K. Wood. Solving a class of stochastic mixed-integer programs with branch and price. *Mathematical Programming*, 108(2-3):395–418, 2006.

[SWR07] Siri-On Setamanit, Wayne Wakeland, and David M. Raffo. Using simulation to evaluate global software development task allocation strategies. *Software Process: Improvement and Practice*, 12(5):491–503, 2007.

[Tav99a] L. V. Tavares. *Advanced models for project management*. Kluwer Academic Publishers, Boston ; London, 1999.

[Tav99b] L. Valadares Tavares. *Advanced models for project management*, volume 16. Kluwer Academic Publishers, Boston, MA, 1999.

[Tav02] L. V. Tavares. A review of the contribution of operational research to project management. *European Journal of Operational Research*, 136(1):1–18, 1/1 2002.

[TC95] John D. Tvedt and James S. Collofello. Evaluating the effectiveness of process improvements on software development cycle time via system dynamics modeling. In *COMPSAC*, pages 318–325, 1995.

[TC06] Lin-Yu Tseng and Shih-Chieh Chen. A hybrid metaheuristic for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 175(2):707–721, 12/1 2006.

[TFC98] L. Valadares Tavares, J. A. Antunes Ferreira, and J. Silva Coelho. On the optimal management of project risk. *European Journal of Operational Research*, 107(2):451–469, 6/1 1998.

[TGR05] K. Taaffe, J. Geunes, and H. Edwin Romeijn. Capacity acquisition and stochastic customer demand assignment in a network of facilities. Technical report, Department of Industrial and Systems Engineering,University of Florida, 2005.

[TL03] P. Tormos and A. Lova. An efficient multi-pass heuristic for project scheduling with constrained resources. *International Journal of Production Research*, 41(5):1071–1086, 2003.

[Uet01] Marc Uetz. Algorithms for deterministic and stochastic scheduling, 2001.

[VBQ05] Vicente Valls, Francisco Ballestín, and Sacramento Quintanilla. Justification and rcpsp: A technique that pays. *European Journal of Operational Research*, 165(2):375–386, 9/1 2005.

[VBQ08] V. Valls, F. Ballestín, and S. Quintanilla. A hybrid genetic algorithm for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 185(2):495–508, 2008.

[Wal08] Grzegorz Waligóra. Discrete and continuous project scheduling with discounted cash flows: a tabu search approach. *Computers & Operations Research*, 35(7):2141–2153, 7 2008.

[WH02] Paul Wernick and Tracy Hall. Simulating global software evolution processes by combining simple models: An initial study. *Software Process Improvement and Practice*, 7:113–126, 2002.

[WKR04] Linda Wallace, Mark Keil, and Arun Rai. Understanding software project risk: a cluster analysis. *Information & Management*, 42(1):115–125, 12 2004.

[WL77] Jerome D. Wiest and Ferdinand K. Levy. *A management guide to PERT CPM : with GERT PDMD CPM and other networks*. Prentice-Hall, Englewood Cliffs, N.J, 1977.

[WS07] Stephen Jonathan Whitty and Mark Frederick Schulz. The impact of puritan ideology on aspects of project management. *International Journal of Project Management*, 25(1):10–20, 1 2007.

[XMNU08] Ningxiong Xu, Sally A. McKee, Linda K. Nozick, and Ruke Ufomata. Augmenting priority rule heuristics with justification and rollout to solve the resource-constrained project scheduling problem. *Computers & Operations Research*, In Press, Corrected Proof:2153, 2008.

[XMQ+04] Zhao XP, Li MS, Wang Q, Chan K, and Leung H. An agent-based self-adaptive software process model. *Journal of Software*, 15(3):348–359, 2004.

[XOZ+07] Junchao Xiao, Leon J. Osterweil, Lei Zhang, Alexander Wise, and Qing Wang. Applying little-jil to describe process-agent knowledge and support project planning in softpm. *Software Process: Improvement and Practice*, 12(5):437–448, 2007.

[YB03] Ali A. Yassine and Dan Braha. Complex concurrent engineering and the design structure matrix method. *Concurrent Engineering*, 11(3):165, 2003.

[YS93] Kum-Khiong Yang and Chee-Chuong Sum. A comparison of resource allocation and activity scheduling rules in a dynamic multi-project environment. *Journal of Operations Management*, 11(2):207–218, 6 1993.

[YS97] Kum-Khiong Yang and Chee-Chuong Sum. An evaluation of due date, resource allocation, project re-

lease, and activity scheduling rules in a multiproject environment. *European Journal of Operational Research*, 103(1):139–154, 11/16 1997.

[zA01a] L. Özdamar and Ebru Alanya. Uncertainty modelling in software development projects (with case study). *Annals of Operations Research*, 102(1-4):157–178, 2001.

[zA01b] Linet Özdamar and Ebru Alanya. Uncertainty modelling in software development projects (with case study). *Annals of Operations Research*, 102:157–178, 2001.

[ZKP00] B. P. Zeigler, T. G. Kim, and H. Praehofer. *Theory and practice of modeling and simulation.* Academic Press, New York, 2000.

[ZL04] Hong Zhang and Heng Li. Simulation-based optimization for dynamic resource allocation. *Automation in Construction*, 13(3):409–420, 5 2004.

[ZnRF04] Ascensión Zafra, Miguel Ángel Ridao, and Eduardo Fernández. An algorithm for optimal scheduling and risk assessment of projects. Technical report, Departamento de Ingenieria de Sistemas y Automática and Universidad de Sevilla. Grupo de Investigación Automática y Robótica Industrial, 2004.

[ZY04] M. Zhuang and A. A. Yassine. Task scheduling of parallel development projects using genetic algorithms. In *Proceedings of the ASME Design Engineering Technical Conference*, volume 1, pages 215–223, 2004.