Universitat Autònoma
de Barcelona

Planificación de la producción en sistemas poco complejos.

(Algoritmo de planificación de BOLD APS)

Memoria del proyecto

de Ingeniería Técnica en

Informática de Gestión

realizado por

Guayasén Osorio Castañeda

y dirigido por

Xavier Verge Mestre

Escola Universitària d'Informàtica

Sabadell, Junio de 2009

El abajo firmante , Xavier Verge Mestre, profesor de l'Escola Universitària d'Informàtica de la UAB

CERTIFICA:

Que el trabajo al que corresponde la presente memoria ha estado realizado bajo su dirección por Guayasén Osorio Castañeda.
Y para que conste firma la presente.
Sabadell, Junio de 2009

Firmado: Xavier Verge Mestre

El abajo firmante, Enrique Font Sierra,
de la empresa Global Planning Solutions,

CERTIFICA:

Que el trabajo al que corresponde la presente memoria ha sido realizado bajo su supervisión por Guayasén Osorio Castañeda

Y para que conste firma la presente.

Sabadell, Junio de 2009

- -

Firmado: Enrique Font Sierra

Agradecimientos:

A Don Francesc López Ramos por enseñarme diferentes formas de abordar el problema de la planificación.

A Don Jaume Golobardes por su amena explicación sobre la teoría de las limitaciones.

A Aurelio Joaniquet Maristany por enseñarme herramientas que hacen más llevadero el trabajo y procedimientos muy útiles a la hora de programar.

A Enrique Font Sierra, por su colaboración a lo largo de todo el proyecto y por tantos ratos ante la pizarra dibujando lotes.

A Gustau Santos Casademont, por molestarse en leer, corregir y comentar esta memoria al mismo tiempo que cementaba en mí conceptos básicos y no tan básicos. Sin su ayuda este trabajo sería mucho peor.

A Raul Azanza por ayudarme con mis primeros pasos en BOLD APS.

A algunos del San Fernando y a quienes empezaron conmigo el duro camino. Sergio (por todo y por cargar conmigo desde hace tanto), Javi (por todo y porque sus clases de S.Digitales llegaron muy lejos) y Chus (por todo y especialmente por la práctica de objetos). A quienes lo terminan conmigo, por sufrirme y ayudarme con cosas que me son imposibles. Gracias Pedro, como no por la Wiki y por hacer que estos agradecimientos sean más largos. Gracias Angel por ser tranquilo, constante y llevarte bien con gente tan diferente. Gracias Sebastián por muchas horas de estudio productivas y por hacer que no tenga amigos normales. Gracias Andrés por ser serio y aguantarnos, sobre todo a mí. Gracias Paola por hacerme pensar cada día que no hago lo suficiente.

A Arístides por parecer una cosa, ser las dos y crear un ambiente adecuado para la escritura de la memoria.

A Laura por su especial paciencia y por lograr que conserve apariencia y comportamientos humanos.

A mis hermanos por lograr que mi casa esté donde sea. A Las Tres Viejas por el fundamento, los refranes y la ingenuidad. A mis padres por el fantástico aeropuerto y por participar activamente en la construcción de este documento.

A Techi y Ricardo por alimentar y cultivar mi interés por lo vivo.

Gracias a todos.

Resum del projecte

BOLD APS es un software diseñado para solventar el problema de la planificación de la producción. Como tal, cuenta con un algoritmo cuya función, es la de tomar las decisiones oportunas, con el fin de obtener una buena programación de tareas. Dicho algoritmo se encuentra en continuo desarrollo, evolucionando conforme lo hacen los problemas a los que ha de enfrentarse, o según avanza la investigación sobre la forma de solventarlos.

Este proyecto pretende ser un paso más, en el proceso de mejora y se encuentra dividido en dos fases.

La principal consiste en el diseño e implementación de una nueva sección dentro del algoritmo de planificación de la producción que utiliza la empresa Global Planning Solutions como parte de su software. El objetivo de esta nueva porción de código es procurar ofrecer alguna mejora en la calidad de las soluciones que proporciona actualmente el algoritmo ya desarrollado.

Se pretende llevar a cabo una ampliación localizada sobre lo que ya existe, de forma que se eliminen o reduzcan algunas de las principales carencias con que cuenta.

La mayoría de los cambios propuestos se encuentran centralizados en un nuevo módulo llamado AlgoReordenaLotes, que funciona como un bypass en un punto estratégico del algoritmo y que permite efectuar modificaciones en la solución final de manera limpia, sin tener que alterar de forma significativa el flujo de código existente con anterioridad. Aunque por supuesto también ha sido necesario crear algunos métodos accesorios en diferentes puntos.

La fase secundaria del proyecto ha consistido en una labor de depuración, limpieza y ordenación del código. El algoritmo que se utiliza fue diseñado hace bastante tiempo y ha sufrido diversas modificaciones y ampliaciones durante los últimos 10 años, lo que ha provocado la existencia de porciones obsoletas o desorganizadas.

Aprovechando que para la ejecución del proyecto era necesario familiarizarse con la tecnología y metodología usada hasta el momento, se ha ido limpiando y segmentando el código de cara a facilitar su comprensión y posterior modificación.

Índice

Prefacio

Yo me llamo Guayasén, mi nombre es indígena de Canarias. Allí, antes de la conquista, cada Isla era propiedad de todos sus habitantes, y los Jefes y los Viejos de las tribus se reunían una vez al año para volver a repartir la Isla, a cada tribu según hubiera crecido en gente, en ganados, o en enfermedades. Todo se tenía en cuenta.

La Isla no podía pasar hambre, y en las Grandes Cuevas de los Acantilados se guardaban de un año para el otro los alimentos que duran, granos, miel, sal, mantecas, carnes secas, grandes quesos o tortas hechas amasando nuestro santo gofio con miel y con higos secos aplastados. Al llegar el mes de la reunión, el beñesmén, se gastaba colectivamente lo suficiente para poder almacenar más, y así mantener siempre las reservas frescas.

Se hacían los cómputos del año (según palabras de los cronistas) para hacer las particiones y saber qué y cuantos bienes debían ser almacenados, gastados o trasladados. Había que tener en cuenta muchas cosas, si se pensaba en un próximo invierno seco, si había muerto tanta gente, si las cabras habían parido bien…

En las Grandes Cuevas vivían los "Viejos que Saben", aislados en los riscos a mitad de camino entre el mar y las nubes. Eran servidos y atendidos por los chicos soñadores de cada tribu, que, con el paso del tiempo se hacían viejos y seguía la cadena, de mano a mano.

"Los Jóvenes que Sueñan" oían a los "Viejos que Saben" sus extraños refranes cantados..."Viento en marzo, más juagarzo", "Hierba chica huele bien, cierra miel cierra miel", "agua hierve allá en el mar, baja sal, baja sal"… estos refranes, como una calculadora poética, encerraban las cuentas del año sumando y restando acontecimientos, pequeños detalles, que la experiencia de los Viejos había ido acumulando y pasando "de mano a mano". No había números complejos, no había almanaques universales ni centros meteorológicos, pero la Isla nunca pasó hambre, porque los cantos y los cuentos de Los Viejos contenían todas las operaciones algorítmicas necesarias para mantener la vida en pie, y los niños y los jóvenes soñadores, sin saberlo, aprendían, con el lento transcurso de sus propias vidas en las Cuevas de los Acantilados.

De ahí este proyecto, porque el mundo de mi nombre sigue existiendo, y una industria, una empresa, cualquier casa, es otra vez una isla que tienen sus propias necesidades y sus propias producciones, y tiene que mantener cada cosa en su sitio cambiándolas continuamente de lugar. Un clavo está en una caja, ahora en la mano de un operario y ahora en el eje de una pata de una silla, y ahora la caja llena de sillas va en un camión hasta un almacén, cuantas cajas caben en cuantos camiones para llenar cuantos almacenes, cuantos barcos nos harán falta llenos de clavos, cuantos vendedores para convertir las sillas en grano, en gofio para guardar y que la Isla-empresa siga viva en todos sus aspectos. Ahora yo escucho a los viejos que me dan la mano y avanzamos un paso más.

Capítulo 1: : Introducción

El problema que trataremos a lo largo de este proyecto es el denominado “Planificación de la producción”. A grandes rasgos consiste en determinar las actividades que han de realizar unos recursos para conseguir unos objetivos fijados de antemano. Resolver este tipo de cuestiones no es algo moderno, sino que lleva ligado a la gestión empresarial desde la revolución industrial. Aunque evidentemente, el tipo y complejidad de los problemas ha aumentado enormemente con el paso del tiempo. Lo que es innegable es que es un tema con historia.

Actualmente, los recursos que hemos de planificar suelen ser máquinas que forman parte de fábricas y que pueden realizar diversas actividades como por ejemplo, una fresadora (Recurso 1), que puede utilizarse tanto para fabricar el molde de un pistón (actividad 1) como para realizar los agujeros necesarios donde irán los tornillos de una carcasa de motor (actividad 2). Normalmente la transición de una actividad a otra conlleva un tiempo de preparación, que puede ser una limpieza, intercambio de piezas, un ajuste electrónico, etc. A esto se le llama tiempo de cambio y es otro tipo de actividad de entre las que puede realizar un determinado recurso. También puede, por supuesto permanecer detenido, aunque en general nuestra intención es mantener los recursos ocupados, ya que de otra forma podríamos estar ante algún tipo de problema como falta de demanda, mala organización en el plan de trabajo o incluso una mala distribución del tipo de recursos que tenemos a nuestra disposición.

Los objetivos que se han de conseguir con estos recursos están formados por las demandas de producción que pueden ser tanto de productos finales (aquello que solicita directamente el cliente) o productos intermedios (materiales que somos capaces de producir y que intervienen en la fabricación de los productos finales).

Actualmente una empresa que desee mantener su competitividad en el mercado, necesita contar con la ayuda de un sistema de planificación automatizado que ofrezca la posibilidad de adaptarse a los continuos cambios que suceden en el seno de su entorno productivo. Actuar de otra manera ha quedado completamente en el pasado o asociado a empresas de pequeño tamaño y sin intención de expansión.

Las empresas que aun no cuentan con software especializado en el tema, abordan su

problema de planificación de la producción de diferentes maneras, aunque siempre con una forma empírica de actuar, basada en el conocimiento de los responsables de producción, lo cual conlleva ciertos inconvenientes que expondremos.

Habitualmente, en PIMES, nos encontramos casos en los que se practica una planificación manual con ayuda de algún tipo de herramienta no orientada a resolver este tipo de problemas como Microsoft Excel. Esta situación suele darse con mayor frecuencia en empresas de mediano o pequeño tamaño, donde la complejidad de las decisiones a tomar en cuanto a qué, donde y cuando producir resulta abordable para los ojos expertos de la figura del encargado de producción. Aún en estos casos, el llegar a una solución adecuada suele ser fruto de un duro trabajo sujeto como cualquier proceso manual a múltiples errores. Al no contar con un entorno de simulación, muchas veces se tratan de efectuar acciones del tipo “¿Qué pasa si…?”, lo cual evidentemente, puede llevar a perder muchísimo tiempo. Además uno de los problemas más graves que se originan al confiar en estos procedimientos, es que la totalidad del conocimiento necesario para llevar a cabo la planificación de la producción, suele residir en una sola persona, con lo que ante cualquier eventualidad o simplemente durante la época de sus vacaciones, se produce una gran desorientación por parte del personal debida a la gran dificultad de llevar a cabo la programación de trabajo, más aún si durante la ausencia del encargado de producción se modifican las demandas previstas, pues la capacidad de reacción de la persona que queda al cargo suele ser limitada dada la falta de costumbre en este tipo de tareas.

Para resolver este tipo de cuestiones algunas empresas optan por desarrollos a medida, lo que tiene como ventaja una adecuación perfecta al modelo de fabricación en cuestión. Aunque suelen ser proyectos demasiado costosos y desaprovechan todo el potencial de desarrollo existente actualmente en el mercado.

Tampoco resulta extraño encontrar entornos en los que se confía en un ERP, con la falsa idea de que este tipo de programas nos aportan una solución al problema de la planificación, cuando lo usual suele ser que nos ofrezcan unos datos aproximados acerca de qué cantidades de productos hemos de fabricar para poder servir los pedidos, pero sin un detalle concreto, pues el modelo utilizado suele no ser lo suficientemente completo como para proporcionar con seguridad planes factibles.

Otra opción es tratar de utilizar alguna de las herramientas disponibles, que tratan en la medida de lo posible de ser adaptables a diferentes modelos de fabricación; esta

flexibilidad resulta un arma de doble filo, pues el software por regla general ha de encontrar un equilibrio entre ella y otros dos factores decisivos, eficiencia y eficacia. Es decir, que cuanto más sencillo resulte plasmar la realidad de una fábrica en un modelo informático o realizar sutiles cambios en él una vez implantado, menos cerca del óptimo se encuentran las soluciones propuestas. Por supuesto hay que comentar también que la relación entre esa cercanía y tiempo de cálculo es exponencial, por ello se debe tener claro qué tipo de problema se quiere abordar, antes de escoger la herramienta que más convenga. Por ejemplo, si se desea conocer la secuencia de producción óptima en una fábrica donde la demanda es siempre muy similar (cercana a constante), es posible que interese invertir el tiempo necesario en conseguir una buena solución, pues se empleará de forma continua sin importar si se ha tardado un mes entero de cálculo o apenas unos minutos para llegar a ella. Sin embargo, cuando las demandas son variables, o las capacidades y/o disponibilidades de los recursos inciertas, se necesita obtener soluciones casi instantáneas para poder reaccionar a tiempo y de forma efectiva antes los eventuales cambios.

Otra cuestión importante a tener en cuenta es el hecho de que cada día surgen nuevas problemáticas asociadas a los diferentes entornos productivos, con lo que es necesario y fundamental contar con una herramienta que asegure su capacidad de actualización, tanto para esto como para poder contar con los últimos avances en investigación y, de esta manera poder siempre trabajar con las mejores soluciones posibles en un tiempo adecuado. No suele ser razonable adquirir un producto que sea estático, pues con total seguridad dejará de ser útil tras la implantación, al observarse que no es capaz de tener en cuenta determinadas cuestiones que fueron pasadas por alto en un primer momento.

Los programas especializados en planificación, ofrecen como gran ventaja a sus clientes frente a desarrollos a medida, el hecho de que evolucionan de forma continua, por lo que seguramente sea posible contar con actualizaciones según avance la investigación en el campo o se encuentren nuevos problemas hasta entonces intratados; además al ser un software con un mercado más amplio los costes por empresa son menores.

BOLD APS es un software de este tipo y el presente proyecto está íntimamente relacionado con el proceso de mejora continua en el módulo del algoritmo de planificación, es decir, el conjunto de reglas y procedimientos que se aplican para llegar a la solución propuesta.

Es importantísimo comprender la importancia y necesidad de actualización en la forma de pensamiento y toma de decisiones, que luego intentamos plasmar en diferentes algoritmos de planificación, pues esto mismo lleva sucediendo de forma natural desde los primeros, escondidos en refranes, hasta el día de hoy donde nos servimos de complejos sistemas informáticos.

Objetivo del proyecto

Este trabajo consiste en el diseño e implementación de las medidas necesarias para obtener una mejora en las soluciones propuestas actualmente por el algoritmo, manteniendo la totalidad de la funcionalidad existente previamente y procurando mantener el rango de tiempos de cálculo con el que suele trabajar (del orden de un máximo 2-3 minutos en casos normales y como límite máximo aceptable de 5 minutos para casos extremos).

Es necesaria también una cierta reestructuración del código existente dentro del módulo del algoritmo, pues a causa de la enorme complejidad y de la forma en que se ha ido trabajando sobre él, ha empezado a resultar inmanejable para alguien ajeno al grupo de trabajo que lo diseñó en su origen. Por ello se tratará de encapsular algo más el código, tratando de evitar clases o métodos excesivamente extensos. También sería de gran ayuda eliminar partes del código que se encuentran actualmente obsoletas. El código fuente del algoritmo consta de 44650 líneas de código, entre comentarios, declaraciones, líneas en blanco, etc. Todas estas líneas se encuentran divididas en 23 ficheros, con lo que de media, cada fichero es de unas 2000. Este es un tamaño bastante por encima del que se podría considerar manejable. El objetivo sería evitar ficheros de más de mil líneas.

Actualmente **BOLD APS** toma sus decisiones en cuanto a la solución propuesta en base a criterios de urgencia y prioridades. Básicamente lo que se trata de hacer es destinar las actividades de los diferentes recursos a hacer lo necesario para cubrir las demandas que nos resultan más complicadas de atender (las más urgentes) y en caso de existir empate en cuanto a urgencia, se optará por aquella producción que tenga mayor prioridad (a nivel de cliente, de recurso en el que se fabrica, de tipo de producto…. etc).

Causa directa de esta forma de proceder es la práctica total ineficacia en resolver casos puntuales de demandas con la misma urgencia, tipos de producto, clientes, recursos de

destino,… similares. El hecho de no poder distinguir la urgencia de dos demandas, nos hace considerarlas iguales por lo que (a causa del complejo proceder del algoritmo de planificación) se acaba buscando minimizar el tiempo no productivo de los recursos, es decir las *operaciones de cambio.* Esto nos puede llevar a descuidar valores interesantes a tener en cuenta como el tiempo que permanecen ocupadas las máquinas. Suele ser bueno no tener recursos detenidos a la espera de materia prima o productos intermedios.

El actual proyecto se centrará en mejorar la forma de actuar ante casos del tipo descrito.

Contenido de la memoria

Después de la introducción la memoria se estructura en los siguientes capítulos:

En el capítulo 2, el estudio de viabilidad, se evalúa este trabajo y su planificación desde el punto de vista de la gestión de proyectos.

El capítulo 3 está constituido por una serie de definiciones básicas para entender el resto del documento. Tras él, se encuentra la exposición del caso a resolver (capítulo 4), junto con la solución teórica del problema y un análisis que puede ayudar a comprender la naturaleza del problema.

El siguiente capítulo, el quinto, contiene una descripción del modelo de datos que utiliza BOLD APS y las estructuras que necesita para poder ejecutar el proceso de planificación, que se describe también en esta sección.

Seguidamente, se detalla la solución propuesta por BOLD APS como fruto de la aplicación de las reglas comentadas anteriormente.

El capítulo seis está formado, por una descripción de los cambios que se han realizado en el código existente durante el transcurso de este proyecto. Tanto los referentes a re-estructuración de código, como a la nueva funcionalidad.

Por último tenemos las conclusiones finales y la bibliografía.

Capítulo 2: Estudio de viabilidad

Este capítulo tiene como objetivo determinar la viabilidad del presente trabajo, estudiando aspectos esenciales que ha de cumplir un proyecto final de Ingeniería Técnica en Informática de Gestión. Se pretende averiguar si es razonable alcanzar los objetivos en un plazo adecuado, para ello se llevará a cabo una planificación del desarrollo y un pequeño estudio de costes. También se comentarán aspectos a analizar previos a comenzar el trabajo y que posiblemente, ayuden a prevenir posibles errores y determinar los puntos conflictivos a lo largo de la vida del proyecto.

Propuesta

Cualquier software del género que sea, para mantener su competitividad en el mercado necesita de un desarrollo continuo. La mejora a diferentes niveles es imprescindible y puesto que BOLD APS es una herramienta de planificación quizás uno de los aspectos fundamentales es la calidad de las soluciones que propone. Conseguir una mejora en ellas, sin ver afectada de forma considerable el tiempo de respuesta y sin perder las funcionalidades existentes, es una buena forma de evolucionar este programa.

Se pretende llegar a mejores soluciones en casos generales, pero nos centraremos en analizar la bondad de no buscar minimizar las *operaciones de cambio,* tratando de trabajar con *lotes* de producción de menor tamaño, lo cual obligará a (por ejemplo) limpiar las máquinas entre la fabricación de un producto y otro más veces, ocupando los recursos con procesos no productivos. Las ideas a aplicar al algoritmo están inspiradas en la teoría *DBR*, de la que se hablará más adelante, y aunque pueda parecer contraproducente en muchos casos se consiguen beneficios.

Durante el transcurso de este proyecto se procurará explicar los fundamentos de esta idea a medida que se intenta trasladarla e incorporar al algoritmo de planificación de BOLD APS.

Evaluación de riesgos inicial

Al tratarse de un desarrollo con un alto contenido teórico en un mundo prácticamente desconocido para el proyectista, el de la planificación industrial, es muy posible que los plazos previstos se puedan ver afectados e incluso que la conclusión satisfactoria del proyecto se vea comprometida. Además el código fuente existente y con el que se ha de tratar, puede resultar abrumador para alguien que no lo ha visto evolucionar desde sus inicios, por lo que el avance, sobre todo durante los primeros meses puede llegar a ser muy lento. El adaptarse a una estructura de datos compleja y desconocida lleva mucho tiempo, hasta conseguir trabajar con la fluidez necesaria para ir realizando cambios en el algoritmo con dinamismo.

Un factor de riesgo importante a considerar es también que el nuevo módulo desarrollado consuma demasiado tiempo de ejecución, convirtiendo su utilización en inviable totalmente para los casos a los que pretendemos enfrentarnos.

Oportunidades del proyecto

La conclusión satisfactoria del proyecto daría como beneficios directos la mejora de las soluciones aportadas, consiguiendo así ser más competitivos en el sector.

- Los clientes actuales y futuros, al disponer de la posibilidad de activar la nueva etapa de análisis de secuencias de producción podrán programar sus tareas de forma más eficiente, con el consiguiente beneficio, tanto por el aumento de producción como por el mayor éxito al entregar pedidos en fecha prevista.
- Aumentar el conocimiento acerca del funcionamiento del núcleo de BOLD APS aumenta las posibilidades de mejora futura y por tanto su propia vida.
- Mediante la etapa de re-estructuración del código se conseguirá facilitar el desarrollo de nuevas modificaciones. También evita posibles errores el hecho de eliminar código inválido y que hace tiempo que no se utiliza, pues en algunos casos se desconoce si actualmente sería correcto su funcionamiento.
- Al existir clientes que conocen, confían y ponen en práctica las ideas propuestas

por E. Goldratt [*] sobre DBR con gran entusiasmo, resulta un punto a favor de BOLD APS el hecho de contar con un módulo inspirado en ellas.

Análisis de costes

Los costes de este proyecto se pueden dividir en dos secciones, costes de personal y costes de amortización del entorno de desarrollo.

Costes de personal:

Hemos planificado un total de 507horas hombre que podemos dividir de la siguiente de la forma:

Concepto	Cantidad	Coste/unidad	Total
Horas de trabajo del proyectista	406 h	20 €/h	8.120 €
Horas de trabajo tutor de proyecto en la empresa	76 h	50 €/h	3.800 €
Horas de trabajo tutor de proyecto en la universidad	9 h	0 €/h	0
Consultor externo (Empresa Torrent i Dedeu)	16 h	0 €/h	0
TOTAL	507		11.920 €

[*] **E.Goldratt. La meta.** Habla acerca del concepto *cuello de botella* como factor limitante de la producción de cualquier fábrica o empresa.

Costes asociados al entorno de desarrollo:

Concepto	Coste total	Amortización mensual	Total
Ordenador Dell Inspiron 6000	1200 €	Amortizamos en 3 años. 33 € / mes	33* 6 meses de duración del proyecto = 200 €
Borland Builder 6.0	400 €	Amortizamos en 2 años que ya han transcurrido. 0 € / mes	0 €
Understand c++ (trial version)	0 €	0 €/mes	0 €
Microsoft Office	700 €	Amortizamos en 2 años. 30 € / mes	30*6 = 180 €
Edit Plus (freeware)	0 €	0 €/mes	0
Tortoise SVN (freeware)	0 €	0 € /mes	0
SQL Server 2005 Express	0 €	0 € /mes	0
Enterprise Architec 7.1 (trial version)	0 €	0 €/mes	0
BOLD WorkPlanner (propiedad de la empresa desarrolladora del proyecto)	0 €	0 €/mes	0
TOTAL			380 €

Organización del proyecto

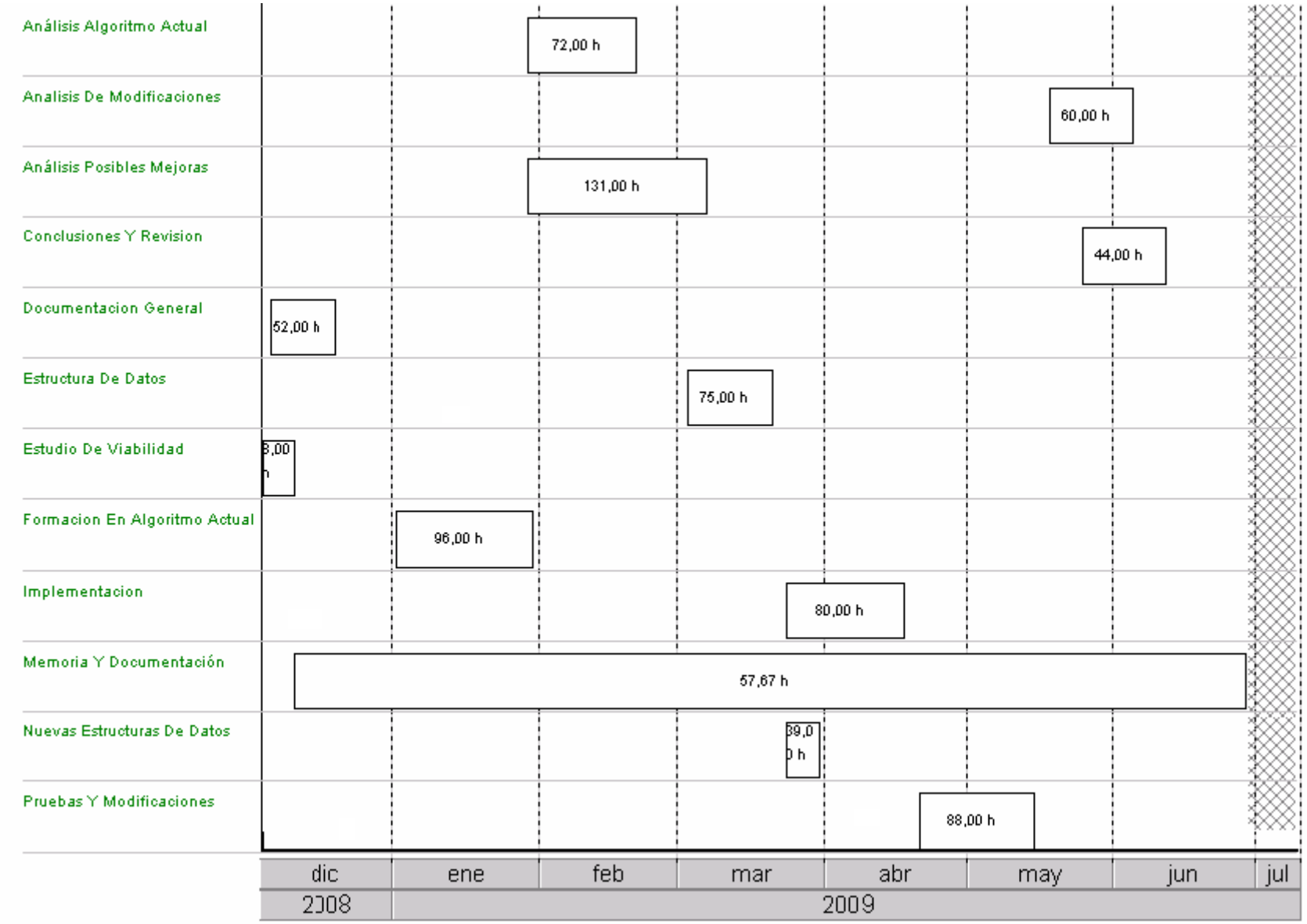
Durante la etapa de definición del proyecto, se diseñó un plan de trabajo que se ha seguido en la medida de lo posible, aunque se han producido muchos retrasos en los plazos previstos, el resultado global ha sido correcto.

Se han programado dos perfiles de desarrolladores para la realización del proyecto. Al ser un proyecto diseñado para ser llevado por una sola persona y tratándose de un caso real llevado a cabo en el seno de una empresa, se puede especificar quien realizará cada

tarea. Los perfiles previstos son el del proyectista, que engloba tanto las tareas de diseño, gestión, análisis, desarrollo… etc. y la de el tutor de proyecto por parte de la empresa Global Planning Solutions, que se ha ocupado de la formación en el uso del software y de las bases sobre las que se sustenta el módulo del algoritmo, así como todo el modelo de datos utilizado. Durante la etapa de análisis acerca de la forma en que se debería comportar el nuevo módulo participaron ambos perfiles y se contó con la inestimable ayuda de uno de los desarrolladores de BOLD APS desde sus inicios. También se contó con la colaboración de un técnico experto en desarrollo de software perteneciente a la misma empresa para mejorar aspectos de la implementación del código desarrollado.

Aunque en un inicio se diseñó la planificación del proyecto como si fuese a seguir un comportamiento lineal, en el que se preveía la realización de varias tareas de forma simultánea, lo que en la práctica ha sucedido es que nos hemos encontrado ante un desarrollo de tipo evolutivo en el que tras avanzar en una fase, nos vimos forzados a revisar las anteriores e incluso a volver a plantear desde el inicio algunas dadas por concluidas. De esta forma podemos pensar en un modelo en espiral del que veremos más detalles en posteriores capítulos.

A continuación se presenta el diagrama de Gantt con la planificación inicial. Para generarlo se ha utilizado una herramienta de planificación de personal similar a BOLD APS (BOLD WorkPlanner) pero orientada a la planificación de recursos humanos.


Análisis Algoritmo Actual
72,00 h
Analisis De Modificaciones
60,00 h
Análisis Posibles Mejoras
131,00 h
Conclusiones Y Revision
44,00 h
Documentacion General
52,00 h
Estructura De Datos
75,00 h
Estudio De Viabilidad
8,00 h
Formacion En Algoritmo Actual
96,00 h
Implementacion
80,00 h
Memoria Y Documentación
57,67 h
Nuevas Estructuras De Datos
89,0 0 h
Pruebas Y Modificaciones
88,00 h
dic
ene
feb
mar
abr
may
jun
jul
2008
2009

Capítulo 3: Conceptos teóricos

Cualquier empresa tiene un objetivo básico: obtener beneficios. Los beneficios pueden definirse de diversas maneras (sociales, individuales, etc), pero habitualmente se refieren a beneficios de tipo económico (la clásica cuenta de resultados de la empresa, aquella que determina su viabilidad presente y futura). Este aspecto debe quedar muy claro, ya que la orientación de cualquier estudio dedicado a mejorar la empresa deberá fijarse metas traducibles a términos económicos.

Desde el punto de vista de la planificación mediante BOLD APS, las empresas pueden dividirse en:

- *Empresas productoras*: Transforman determinadas materias primas en productos elaborados, de mayor valor añadido. Esa transformación implica un consumo de recursos que pueden ser de diversos tipos (tiempo, humanos, máquinas, agua, electricidad, dinero, productos, etc.).
- *Empresas de servicios*: Si bien de forma muy genérica podrían tratarse de la misma forma, la verdad es que el "producto" con el cual trabajan es mucho menos tangible. Trabajan con tiempo, recursos humanos y otros tipos de recursos, pero el proceso de transformación obtiene un tipo de productos que son o bien simplemente servicios diversos o en ciertos casos información.

En una primera aproximación, nos centraremos en las empresas productoras.

Los beneficios de una empresa productora pueden mejorarse:

- reduciendo costes de producción.
- reduciendo tiempos de operación.
- reduciendo consumos de energía u otros recursos (agua, vapor, etc).
- reduciendo emisiones de residuos (sólidos, líquidos o gaseosos).
- reduciendo la necesidad de mano de obra por automatización de procesos.

Los diversos métodos para mejorar los beneficios de una empresa pueden clasificarse en los siguientes tipos:

Técnicas: mejorar el proceso técnico de fabricación

Organizativas: mejorar la organización y gestión del proceso productivo

La aplicación BOLD APS se centrará en los métodos organizativos, aunque como resultado de su uso, la empresa cliente dispondrá de una herramienta que le permitirá analizar con detalle y fácilmente las posibles mejoras técnicas y productivas.

Los sistemas productivos de una empresa se pueden mejorar, desde el punto de vista organizativo, de diversas formas, dependiendo del nivel tecnológico existente en la empresa y de la inversión a realizar (tiempo, recursos humanos, dinero).

- *Homogeneizar las diferentes estructuras de información:*

  La organización de una empresa es, en la actualidad, altamente jerarquizada y estructurada en departamentos poco permeables. Es muy frecuente, aunque cada vez menos habitual, que los diferentes departamentos de una empresa sigan objetivos de importancia local y mantengan estructuras de información de ámbito local.

  Habitualmente, las necesidades de calidad y servicio al cliente, obligan a las empresas a un proceso de re-estructuración para cumplir los siguientes puntos:

- *Informatizar la empresa, para obtener sistemas organizados de información:*

  Es una posible forma de mejorar la estructura de información de la empresa. Al centralizarla en sistemas de tratamiento de la información especializados, se asegura (al menos en teoría) que la homogeneización se realiza de forma consistente y lógica.

- *Homogeneizar los diferentes sistemas de información:*

  Ciertas empresas pueden tener diferentes sistemas de información, pero independientes (no se hablan entre sí). Es también una forma de estructurar la información y hacerla más consistente. Suele ser un proceso más costoso que el anterior, pero también es lo más habitual, ya que difícilmente nadie “partirá de cero”.

- *Organizar y automatizar los procesos de análisis y toma de decisiones:*

  Al realizarse la toma de decisiones bajo una estructura departamental, es

inevitable que surjan conflictos de una cierta importancia, nuevamente entre los departamentos comercial y de producción, por ser éstos los más importantes en el proceso de la empresa.

Por ejemplo, el criterio del departamento comercial suele ser “no hay problema, su pedido será entregado a tiempo”, mientras la respuesta del departamento de producción suele ser “no me hagas esto, ahora debo parar máquinas; me has roto el plan de producción”.

Es evidente que una adecuada organización y definición de las políticas globales de la empresa ayuda a un mejor funcionamiento de la misma. Esta organización puede llevarse a cabo de forma independiente al proceso de automatización, pero no suele ser así, ya que el esfuerzo que conlleva necesita generalmente de una previsión de mejora apreciable, lo cual sucede en general al intentar automatizar las tomas de decisiones.

Esquemas productivos

Los esquemas productivos de una empresa pueden clasificarse de diversas formas, según su política de producción, según el tipo de operación de sus procesos, según la complejidad de su esquema productivo, según su política de transferencia y almacenamiento de productos intermedios, etc.

1. Según su política de producción:

- *Producción contra stock*: Las necesidades de producción se encuentran con aspectos estacionales o climáticos, los volúmenes de producción suelen ser muy altos y obtención costosa en tiempo de operación.
- *Producción bajo pedido*: Los volúmenes de producción suelen ser bajos y los tiempos de operación relativamente reducidos, lo cual permite una rápida reacción a las necesidades del mercado.
- *Producción mixta*: Como su nombre indica, estos sistemas de producción comparten características de los dos anteriores. No son muy comunes, pero a pesar de todo, la tendencia de mercado deriva en ciertos casos a tal sistema.

2. Según el tipo de operación de sus procesos:

    - *Procesos continuos*: La forma de obtener los diversos productos intermedios se realiza mediante operaciones de tipo continuo. Un ejemplo son las refinerías de petróleo y ciertas empresas del textil. Un ejemplo “casero” sería el agua de un grifo: mientras la llave de paso está abierta, sale agua de forma continua, y no cesa hasta que dicha llave se ha cerrado.
    - *Procesos discontinuos*: La forma de obtener algunos de los diversos productos intermedios y/o finales se realiza mediante operaciones de tipo discontinuo (se trabaja por cargas).

3. Según la complejidad de su esquema productivo:

    - *Procesos monoetapa*: La obtención de un determinado producto implica solo una etapa de producción.
    - *Procesos multietapa*: La obtención de un determinado producto implica más de una etapa de producción. En tales subprocesos podemos diferenciar:
        - *Secuencias lineales*: Las etapas siguen una secuencia lineal sin ningún tipo de ramificación.
        - *Secuencias con ramificaciones*: Existen etapas que pueden ser simultáneas a otras dentro del mismo proceso.

4. Según su política de transferencia y almacenamiento de intermedios:

    - *Política ZW*: El producto obtenido en una de las etapas es inestable y debe ser procesado inmediatamente en la siguiente etapa del proceso. Es lo que se conoce como política *ZW* (*Zero Wait*) o tiempo de espera nulo.
    - *Política UIS*: El producto obtenido es estable y puede ser almacenado “en cualquier sitio”, sin limitaciones prácticas en cuanto a la disponibilidad de espacio de almacenamiento. Se conoce como UIS (Unlimited Intermediate Storage) o almacenamiento ilimitado.
    - *Política FIS*: El producto obtenido tiene una cierta estabilidad, y puede ser almacenado en los depósitos o contenedores específicamente aptos para él. Se conoce como *FIS* (*Finite Intermediate Storage*) o almacenamiento limitado,

debido a límites en número, capacidad y disponibilidad de los contenedores posibles.

- *Política NIS*: El producto obtenido tiene una cierta estabilidad, y puede ser almacenado en el mismo equipo de proceso que lo ha generado. Se conoce como *NIS* (*No Intermediate Storage*), debido a que el almacenamiento se realiza, en caso de ser posible, sin el uso de ningún equipo adicional.
- *Politicas mixtas* Básicamente se trata de políticas de tipo *NIS/FIS*, con lo que existe la posibilidad de almacenar temporalmente el producto en el equipo que lo ha generado, o bien transferirlo a uno de los depósitos aptos para él.

Planificar la producción

De forma totalmente genérica implica definir una serie de acciones futuras en base al conocimiento presente del estado de un determinado sistema, de acuerdo con unos determinados objetivos, los cuales se desea conseguir de la mejor manera posible. De forma más concreta, en el ámbito de empresas productoras, planificar significa satisfacer las demandas del mercado con los recursos disponibles en la empresa.

Es decir, a partir de una serie de datos de entrada:

- *Demandas*: Pueden ser demandas confirmadas o previsiones de venta.
- *Recetas*: Indican como obtener los productos de cada demanda, es una descripción del proceso productivo necesario para conseguir un determinado producto.
- *Recursos*: Entidades que se usan en una receta para producir. En esta categoría entran tanto los recursos reutilizable (máquinas, personas…) como los no reutilizables (materias primas, productos intermedios).
- *Restricciones*: Indican ciertos aspectos que pueden o deben (depende del caso) considerarse en la planificación
    - *Estado actual de la planta*: qué se está produciendo (que productos, en que cantidad, de que demanda), donde se está produciendo (recursos

principales y secundarios usados) y cómo se está produciendo (en que estado se encuentra cada operación).

- *Disponibilidad actual de los recursos*: Niveles actuales de stock (disponibilidad) de todas las materias primas, productos intermedios y finales, así como la disponibilidad del resto de los recursos. La disponibilidad de los productos puede ser tanto la real como la prevista pero confirmada como la prevista y pendiente de confirmación.
- *Rupturas de disponibilidad*: Implica considerar o no los niveles máximos y mínimos de disponibilidad de los recursos. En ciertas situaciones se puede desear la "relajación" de las mismas.

- *Costes*: Sirven para valorar un plan de producción. Se trata de un conjunto de términos económicos (sean de definición objetiva o subjetiva). Pueden ser constantes (costes fijos) o variables (tienen en cuenta el área de impacto de un hecho concreto), o funciones más complejas sobre diferentes propiedades del sistema, tanto respecto los datos de entrada como respecto los resultados del proceso de planificación.

Se debe obtener como resultado:

- *Plan de producción*: Conocer el uso de los recursos de la planta, tanto en cuanto al nivel de uso como al tiempo de utilización de los mismos.
- *Necesidades de materias primas*: A partir del plan de producción, y conociendo que materias primas son necesarias, en que fecha, y en que cantidades (partiendo de los balances másicos definidos en la receta de obtención del producto, lo que habitualmente se conoce como *BOM* o *Bill of Materials*), se puede generar la lista de necesidades de materias primas, que podrá utilizarse para decidir el plan de compras de las mismas.

El proceso de planificación implica decidir que fracción de las Demandas se cubre a partir del stock de productos finales, que Demandas pueden producirse de forma conjunta (agregación) o bien de que forma se dividen (lotificación) por aspectos técnicos, económicos o de política productiva, que Recursos se utilizan (asignación), en que intervalos de tiempo (temporización), y en que orden se procesan los diferentes trabajos o tareas en cada Recurso (secuenciación).

La sistematización y automatización del proceso de planificación se traduce en lo que habitualmente se conoce como algoritmo de planificación, que “simplemente” se encargará de tomar las mejores decisiones para obtener un plan de producción razonablemente adecuado de acuerdo con los datos disponibles.

En forma gráfica, la obtención de un plan de producción seguiría el esquema:

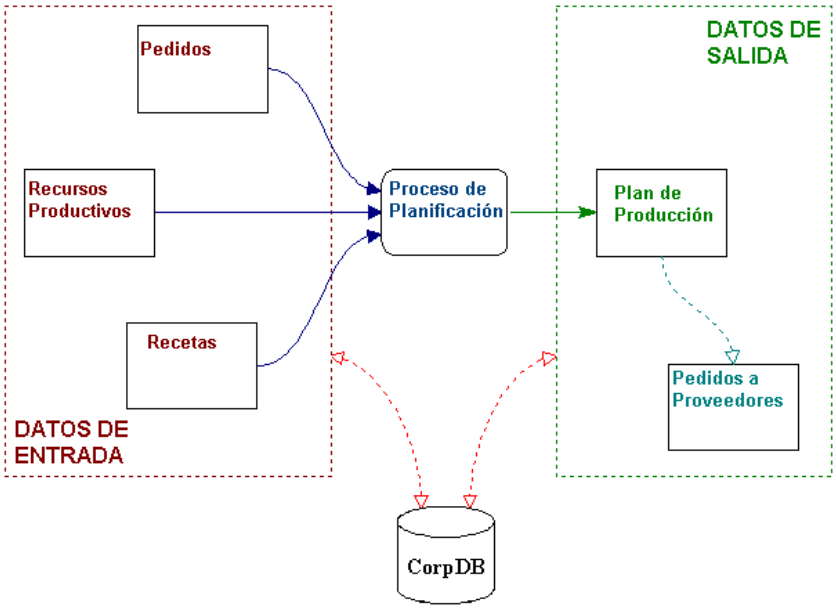


Figura 1 Esquema de obtención de un plan de producción

Claramente, la aplicación BOLD se insertaría en el esquema anterior según:

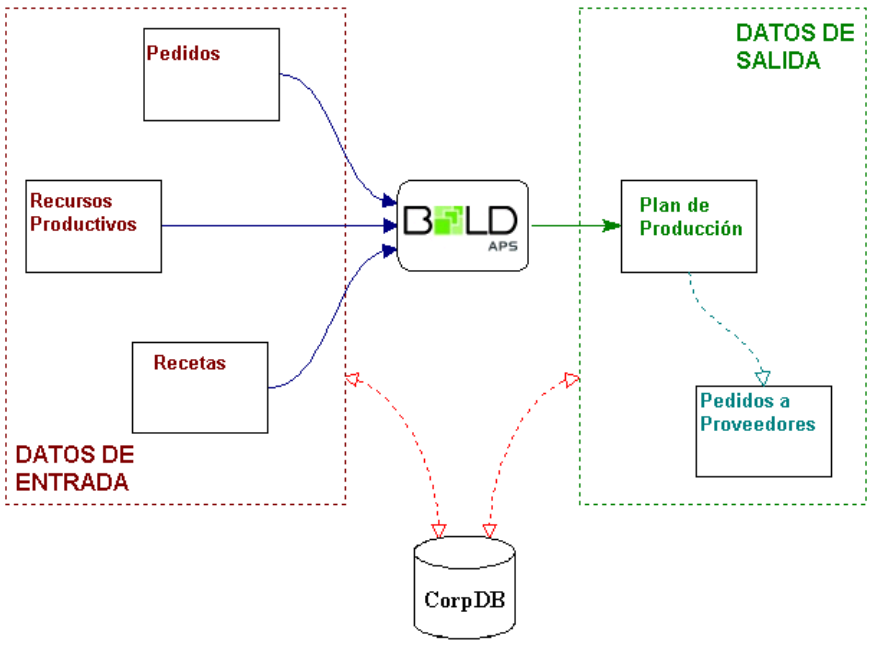


Figura 2 BOLD y la obtención de un plan de producción

Capítulo 4: La herramienta BOLD APS

En este capítulo describiremos el modelo de datos de BOLD APS y el flujo de toma de dediciones del algoritmo, con el fin de entender la solución que aporta y los cambios que se proponen.

Modelo de datos en BOLD APS

La herramienta ha de ser alimentada con una serie de datos maestros a partir de los cuales se generará el plan de producción, la estructura de éstos y sus características principales se exponen a continuación.

Producto

Propiedades inherentes al producto:

- *Estabilidad:* La *estabilidad* (o *caducidad*) de un producto, permite determinar en que momento pierde su identidad original por modificación de alguna de sus propiedades físico-químicas. Esta característica sirve, básicamente, para decidir durante cuanto tiempo puede estar almacenado antes de ser usado (servido al cliente, procesado para obtener otros productos, etc.). Evidentemente, superado el tiempo de estabilidad, el producto es inservible y debe ser eliminado (en cierta forma debe considerarse como un residuo que habría que tratar).
- *Propiedades auxiliares del producto:* Dependiendo del sector industrial de la empresa productora, pueden existir otras propiedades relevantes en el sentido de utilizarse para tomar decisiones y no a título puramente informativo. En realidad, la estabilidad podría aparecer como una propiedad auxiliar más, máxime si tenemos en cuenta que en algunos casos puede no tener sentido, si el producto es, por definición, no perecedero.

  Por ejemplo, podemos diferenciar las siguientes propiedades auxiliares

  - *Propiedades físico-químicas*: concentración de ciertas substancias, etc.
  - *Propiedades colorimétricas:* color, brillo, intensidad, lustre, etc.
  - *Propiedades termomecánicas:* dureza, fragilidad, etc.
  - *Propiedades organolépticas:* olor, sabor, etc.

- *Propiedades geográficas:* localización de los almacenes y contenedores donde se guarda el producto.

Propiedades asociadas al producto (política productiva)

- *Tamaños mínimo y máximo de lote:* Derivados de la política productiva y/o de restricciones de tipo técnico.
- *Nivel mínimo de stock:* Se trata de una restricción de política productiva según la cual se desea mantener en almacén una cierta cantidad de producto (cantidad también llamada stock de seguridad, nivel de ruptura) para asegurarse una mínima capacidad de reacción delante de posibles variaciones del mercado. Suele ser bastante habitual en sectores industriales donde se produce contra stock, ya que se trabaja con previsiones que son obtenidas como valores probables y por ello con un cierto grado de confianza.
- *Nivel máximo de stock:* A pesar de parecer a primera vista un contrasentido físicamente imposible, este tipo de restricción también es frecuente, ya que permite evaluar la posibilidad de "subcontratar" una cierta capacidad de almacenaje o bien ampliar la capacidad actual. Naturalmente, al estar definida a nivel de cada producto, cuando es posible que el/los /almacén/es pueden ser compartidos, debería existir una forma de asegurar cuando se supera la capacidad "total" de el/los almacén/es.
- *Coste asociado al stock:* De forma estricta, si un producto debe almacenarse durante un tiempo, existe un "coste de inmovilizado" asociado al tiempo de uso del almacén y la cantidad almacenada. En caso de no tratar de manera explícita el concepto de almacenes y contenedores de producto (silos, palets, etc.), entonces debe asociarse al producto (se trata el almacén de forma implícita). Dichos costes pueden ser de tres tipos:
  - *Coste de stock:* Es el coste asociado al hecho de almacenar una cierta cantidad de producto durante un cierto tiempo.
  - *Coste de ruptura superior de stock:* Es el coste asociado al hecho de almacenar una cierta cantidad de producto durante un tiempo en el cual la cantidad supera el nivel máximo de stock previamente definido.
  - *Coste de ruptura inferior de stock:* Es el coste asociado al hecho de almacenar una cierta cantidad de producto durante un tiempo en el cual la cantidad está

por debajo del nivel mínimo de stock previamente definido. Normalmente hace referencia a costes de oportunidad.

- *Nivel máximo de anticipación*: Es una característica asociada a una política productiva. Sirve para decidir los niveles de "rotación" de stock en el almacén. Por ejemplo, en producción contra stock, no producir con más de un mes de antelación. Tiene una cierta relación con la estabilidad o caducidad del producto, en el sentido que no puede ser ni mayor ni igual a la estabilidad del producto. En realidad debería ser bastante menor, si tenemos en cuenta que algunos productos deben enviarse a grandes superficies que lo distribuyen a otros intermediarios antes de llegar al público, el cual debe poderlo "consumir" con tranquilidad y de forma no inmediata.

Cliente

Algunas de las características más importantes del cliente (desde el punto de vista de la planificación) son las siguientes:

- *Prioridad*: Por ejemplo, si el cliente es Nacional o bien corresponde a una Exportación, o bien otro tipo de prioridades basadas en la confianza mutua entre las empresas, la solidez financiera del cliente, su tamaño empresarial (volumen de facturación anual), etc.
- *Coste asociado a retrasos:* El coste asociado a los retrasos puede tener una componente fija y una componente variable, en función del tiempo y/o de la cantidad retrasada.

Pedido del cliente

Algunas de las características más importantes del pedido (desde el punto de vista de la planificación) son las siguientes:

- *Pedido divisible*: Indica si las cantidades de los elementos que conforman el pedido se pueden servir de forma independiente en el tiempo, o bien deben servirse de forma simultánea.
- *Fecha de solicitud*: Es la fecha en la cual el cliente solicita el pedido. Obviamente (y básicamente a efectos de simulación de escenarios productivos), es imposible programar, en fecha anterior a la fecha de solicitud, una acción destinada a servir algún elemento de dicho pedido.

Demanda del cliente

Algunas de las características de la demanda que pueden ser importantes en el proceso de toma de decisiones de planificación son las siguientes:

- *Demandas o previsiones:* La demanda puede corresponder a una *previsión de ventas* (no tiene un cliente asociado), una *demanda prevista* (con cliente asociado pero no confirmada, también llamada *contrato*) o una *demanda confirmada*.
- *Cantidad deseada:* y producto asociado.
- *Fecha de entrega*: Es la fecha en la cual el cliente desea obtener la cantidad de producto.
- *Entrega parcial*: Indica si los lotes correspondientes a la demanda se pueden servir de forma independiente en el tiempo o bien deben servirse de forma simultánea.
- *Demanda ficticia:* A efectos de simulación, se puede tratar una determinada demanda como ficticia.

Recurso

Según su comportamiento respecto la forma de realizar la asignación de los trabajos, los recursos se pueden dividir en dos grandes categorías: renovables y no renovables.

- *Recursos renovables:* Los recursos renovables tienen la característica de que una vez se ha programado (planificado) la utilización de una cierta cantidad de dicho recurso durante un determinado periodo de tiempo, dicho recurso queda liberado (vuelve a estar disponible) en el instante en que se finaliza su uso. Es decir, las unidades utilizadas no se consumen, sólo se utilizan. Ejemplos de este tipo de recursos: máquinas, personal, líneas eléctricas, etc.

- *Recursos no renovables: L*os recursos no renovables se consumen, es decir, la cantidad utilizada no se vuelve a recuperar al final de su uso, sino que debe pasar por un proceso de reposición adicional (compra de materias primas, generación de productos, etc.). En esta categoría se encuentran los productos que a nivel interno se tratan como un tipo especial de recursos (materias primas, productos intermedios, etc.), o incluso el propio dinero.

- *Recursos unitario y recursos compartibles:* En recursos cuyo nivel sea binario (disponible o no disponible) se habla de recurso unitario. En caso de poder ser utilizado de forma simultánea para diferentes trabajos, se habla de recurso compartible, como por ejemplo productos, personal, electricidad, refrigeración, etc.

- *Recursos discretos y recursos continuos:* Podemos hablar también de recursos discretos (cuando involucran entidades no divisibles: equipos, productos en forma de piezas, etc.) o recursos continuos (cuando involucran entidades divisibles: refrigerantes, electricidad, productos líquidos o sólidos, etc.).

En relación a los recursos renovables, algunas de las características que pueden ser importantes en el proceso de toma de decisiones de planificación son las siguientes:

- *Tipo de operación (continua o discontinua):* Hablaremos de operación continua cuando las unidades de producto se van obteniendo continuamente (por ejemplo, una línea de producción). La operación será discontinua cuando no está disponible ninguna unidad de producto hasta que no se ha procesado el lote entero (por ejemplo, un reactor).
- *Capacidad mínima y máxima de utilización*: Aquí podríamos definir varias capacidades. Actualmente cada recurso tiene una única dimensión característica (masa, volumen, etc.) con un calendario de disponibilidad asociado a la misma, pero se pueden definir otras dimensiones características. Si no fuese necesario asociarles un perfil, podrían ser propiedades auxiliares, pero en ciertos casos no será así (Por ejemplo: peso, alto, ancho, largo en hornos de siderurgia).
- *Propiedades auxiliares del recurso:* Otras propiedades relevantes en la toma de decisiones (tamaño del recurso, niveles de temperatura a los que trabaja, etc.).

Calendario

Algunas de las características que pueden ser importantes en el proceso de toma de decisiones de planificación son las siguientes:

- *Calendarios maestros o normales:* El calendario maestro es un calendario padre de otros calendarios normales. No está asociado a ningún recurso (por ejemplo, el calendario correspondiente a una empresa). Un calendario normal es un calendario sin “hijos” asociado a uno o más recursos (por ejemplo, el calendario correspondiente a los recursos de producción de una empresa).
- *Calendarios normales o normales unitarios:* El calendario normal unitario está asociado a recursos de tipo unitario (su nivel de disponibilidad sólo puede ser 0 o 1). Al contrario, el calendario normal (no unitario) se asocia a recursos compartibles, pudiendo ser su nivel de disponibilidad cualquier nº (Por ejemplo: 3 personas).

- *Calendarios compartidos o únicos:* Un calendario compartido es aquel que está asociado a más de un recurso. Si no es así, hablaremos de calendarios únicos.

Recetas

La receta en BOLD APS se puede definir a tres niveles diferentes, (según modelo para representar el proceso de producción). Los niveles posibles son los siguientes: Proceso, Etapa y Operación Básica, de los cuales el tercero, más detallado, es opcional.

Proceso

A nivel de proceso se detallan los productos de entrada y de salida (E/S), indicando la proporción existente entre cada uno de ellos respecto de los otros (balance de material de E/S).

En líneas generales, en una receta aparecerá más de un proceso cuando la secuencia de fabricación sea divisible entre varios grupos de operaciones y cuando entre dichas operaciones se generen productos intermedios almacenables por un cierto periodo de tiempo antes de ser tratados en el siguiente proceso.

BOLD APS permite definir un balance de materias que no cumpla las restricciones estequiométricas clásicas (masa total de los productos de entrada igual a masa total de los productos de salida) e incluso permite definir únicamente los productos de salida que se desee contemplar, obviando la necesidad de definir subproductos u otros residuos generados en la transformación.

Etapa

Una vez definidos los procesos que componen una receta, se debe detallar la secuencia de etapas de cada proceso. Para cada etapa se deben detallar los recursos necesarios, indicando todas las alternativas posibles si existe más de una situación posible.

Por etapa se entiende una operación (simple o compleja) a lo largo de la cual se utilizará de forma permanente un recurso principal (por ejemplo una máquina), al que nombraremos como recurso principal o primario.

Para cada etapa se deben definir los recursos necesarios para realizarla, indicando todas las alternativas si existe más de una situación posible. Cada alternativa compone lo que se define como grupo de recursos. Una etapa compleja puede estar compuesta de un

conjunto de operaciones elementales llamadas operaciones básicas.

Algunas de las características más importantes respecto la definición de Etapas de un Proceso, en relación a las decisiones de planificación, son las siguientes:

- *Divisible temporalmente*: Debe conocerse si la Etapa puede pararse temporalmente y reanudarse más tarde.
- *Permitir cambio de tarea*: Esta característica indica, para Etapas divisibles, la posibilidad de programar otros trabajos entre las 2 o más divisiones de la misma Etapa. Además, podría analizarse la opción de asignar el "trozo" restante a otro recurso principal.
- *Precedencia temporal*: Debe conocerse con detalle el tipo de precedencia temporal entre etapas, que nos definirá la política de transferencia de intermedios.

Operación básica

Cada etapa puede descomponerse a su vez en una serie de operaciones (llamadas operaciones básicas), que suelen coincidir con la definición, en Ingeniería de Procesos, de operaciones elementales de transformación físico-química de los productos.

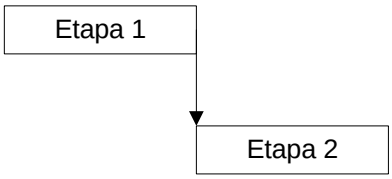
Las operaciones básicas de una etapa se definen como una secuencia de operaciones a realizar sobre los recursos principales definidos a nivel de etapa. Este nivel existe para poder definir con detalle procesos de fabricación complejos, como pueden ser las diferentes cargas de material, calentamientos previos, reacciones químicas en etapas diferenciadas, descargas parciales de subproductos, etc.

Enlaces temporales entre operaciones (etapas u operaciones básicas)

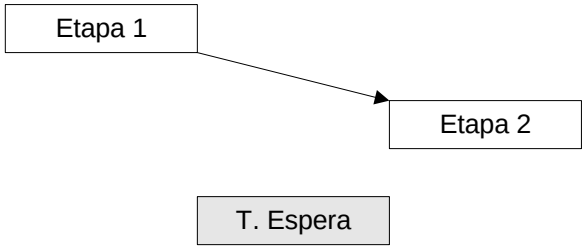
Cada una de las etapas de un proceso debe tener definida de qué forma se relaciona con las siguientes, así como el posible uso de tiempos de espera y de capacidad adicional. Si bien pueden existir otras situaciones más complejas (e incluso situaciones mixtas), la definición más habitual corresponde a una de estas cuatro posibilidades:

- Espera nula (ZW, "*Zero Wait*"): La segunda etapa debe procesarse inmediatamente después de la finalización de la primera etapa, habitualmente por características

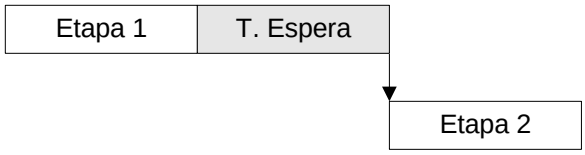
físico-químicas (caducidad, temperatura, etc.). Este enlace implica un tiempo de espera nulo y ningún uso de capacidad adicional.



- Espera ilimitada (UIS "*Unlimited Intermediate Storage*"): La segunda etapa puede procesarse un cierto tiempo después de la finalización de la primera etapa. Este enlace implica un tiempo de espera que puede ser positivo (y que debe ser menor o igual que un cierto valor máximo) y ningún uso de capacidad adicional (equivale a decir que se espera "en planta", y no ocupa recursos productivos).

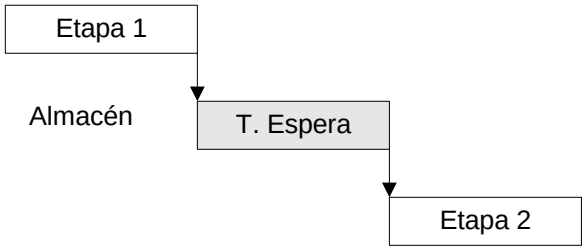


- Espera en máquina (NIS "*No Intermediate Storage*"): La segunda etapa puede procesarse un cierto tiempo después de la finalización de la primera etapa. Este enlace implica un tiempo de espera que puede ser positivo (y que debe ser menor o igual que un cierto valor máximo) y el uso de capacidad adicional (se espera "en la misma máquina", y ocupa recursos productivos).

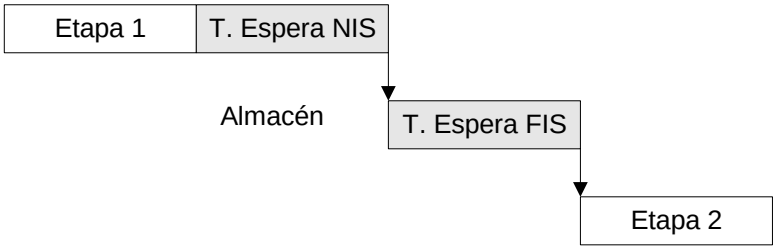


- Espera en depósito (FIS "*Finite Intermediate Storage*"): La segunda etapa puede procesarse un cierto tiempo después de la finalización de la primera etapa. Este

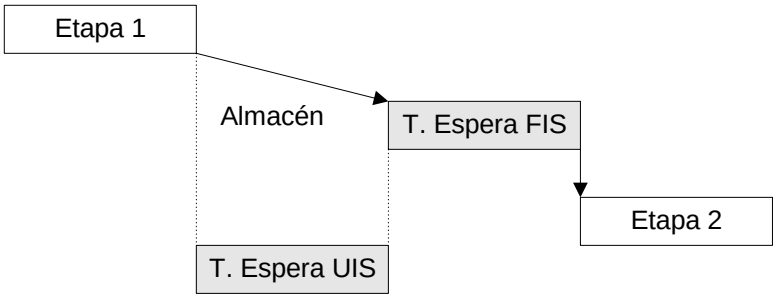
enlace implica un tiempo de espera que puede ser positivo (menor o igual que un cierto valor máximo) y uso de capacidad adicional (se espera “en una máquina dedicada”, y ocupa recursos productivos).



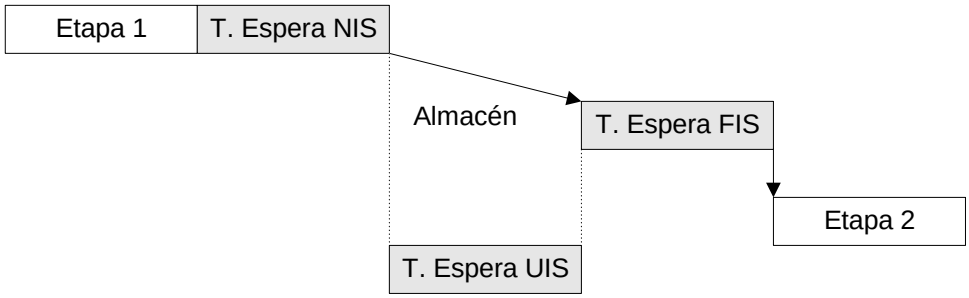
- Ejemplos de situaciones mixtas:
  - Espera mixta (NIS/FIS)



  - Espera mixta (UIS/FIS)



  - Espera mixta (NIS/UIS/FIS)



Grupo de Recursos

El grupo de recursos se asigna a cada una de las etapas y/o operaciones básicas de la receta e indica las diferentes alternativas de realización de cada una de ellas. Para cada recurso de cada grupo se especifica la cantidad y tiempo necesario para realizar la etapa en cuestión.

Cada etapa y/u operación básica tendrá asociados uno o más grupos alternativos de recursos para poder ser llevada a cabo. Esta definición puede realizarse de forma específica (cada combinación de recursos genera un grupo alternativo) o genérica (a nivel de categoría si hemos clasificado previamente los recursos según sus características).

El proceso de planificación

Definiciones preliminares

Plan

Un Plan es la representación del uso de los recursos de la planta, tanto en cuanto a la cantidad como al tiempo de utilización de los mismos. Dicha representación contiene aquella información necesaria para conocer de qué manera se cubren las necesidades de los clientes. El Plan consta de un conjunto de Lotes de diferentes tipos (lotes de producción, lotes de reserva y lotes de cambio).

Lote

Un Lote es una instancia de un Proceso de una Receta. Un Lote contiene la siguiente información:

- demandas (qué demanda o demandas cubre el lote).
- producto (qué se produce).

- cantidad (en qué cantidad se produce).
- proceso - receta (qué receta y proceso se utiliza para obtenerlo).
- productos de entrada (qué productos de entrada necesita y en qué cantidad).
- grupos de recursos (qué recursos realizan las operaciones básicas y etapas).
- localización temporal (cual es el instante de inicio y finalización de cada operación básica).

Cada lote está formado por un conjunto de operaciones asignadas y puede encontrarse en diversos estados mientras el plan de producción evoluciona. Estos estados son los siguientes:

- *Lote inicial*: Al arrancar la ejecución del algoritmo solo se conoce la Cantidad de Producto que debe obtenerse.
- *Lote procesado*: Se conoce además la Receta y Proceso.
- *Lote asignado*: Se conoce además el/los Grupos de Recursos a utilizar en cada Etapa/OpBasica
- *Lote temporizado*: Se conocen además las cantidades y los tiempos de inicio y finalización del uso de cada uno de los Recursos de los Grupos utilizados en cada Etapa/OpBasica del Proceso y Receta elegidos.

Un lote puede ser de diferentes tipos:

- *Lote de producción*: cuando las demandas a las que hace referencia se cubren por producción.
- *Lote de cambio*: corresponde a una tarea de tiempo de cambio.

Trazabilidad

Un lote puede actuar como productor y/o consumidor de ciertos productos intermedios. La trazabilidad de materiales implica conocer con exactitud que lotes actúan como proveedores de una cierta cantidad de un producto para ser consumidos por un determinado lote posterior. Este aspecto resulta de gran importancia cuando se trabaja en sectores relacionados con la alimentación, pues en caso de detectarse un lote defectuoso es imprescindible tener la oportunidad de identificar los lotes que están relacionados con él, de forma que nos resulte sencillo retirarlos del mercado o llevar a cabo las acciones que se consideren oportunas.

Operación Asignada

Una operación asignada es la instancia de una Etapa/OpBasica de un Proceso de Receta. La operación asignada es la entidad que informa de cada operación básica del proceso, indicando que grupo de recursos la realiza y cual es la duración y cantidad de uso de cada recurso. Una operación asignada consta de una o más Tareas.

Tarea

Dado un recurso, la tarea informa de su uso mediante una operación asignada, tanto en cantidad como en tiempo. Dicho de forma más formal, una Tarea es la instancia del uso de un Recurso del Grupo de Recursos utilizado en una Etapa/OpBasica. Las tareas pueden ser de diversos tipos:

- *Productivas*: realizan transformaciones físico-químicas.
- *No productivas*:
    - Cambios.
    - Espera.
    - Transporte.
    - Mantenimiento.
        - *Mantenimiento periódico*: Por ejemplo cada X horas.
        - *Mantenimiento productivo*: Por ejemplo cada X horas de producción.

Etapas en la Planificación

El proceso de planificación implica decidir qué fracción de las *Demandas* se cubre a partir del stock de productos finales (**reserva**) o por compra externa (**compra**). Qué *Demandas* pueden producirse de forma conjunta (**agregación**) o bien de qué forma se dividen (**distribución, lotificación**) por aspectos técnicos, económicos o de política productiva; qué Recursos se utilizan (**asignación**), en qué intervalos de tiempo (**temporización**), y en qué orden se procesan los diferentes trabajos o tareas en cada *Recurso* (**secuenciación**).

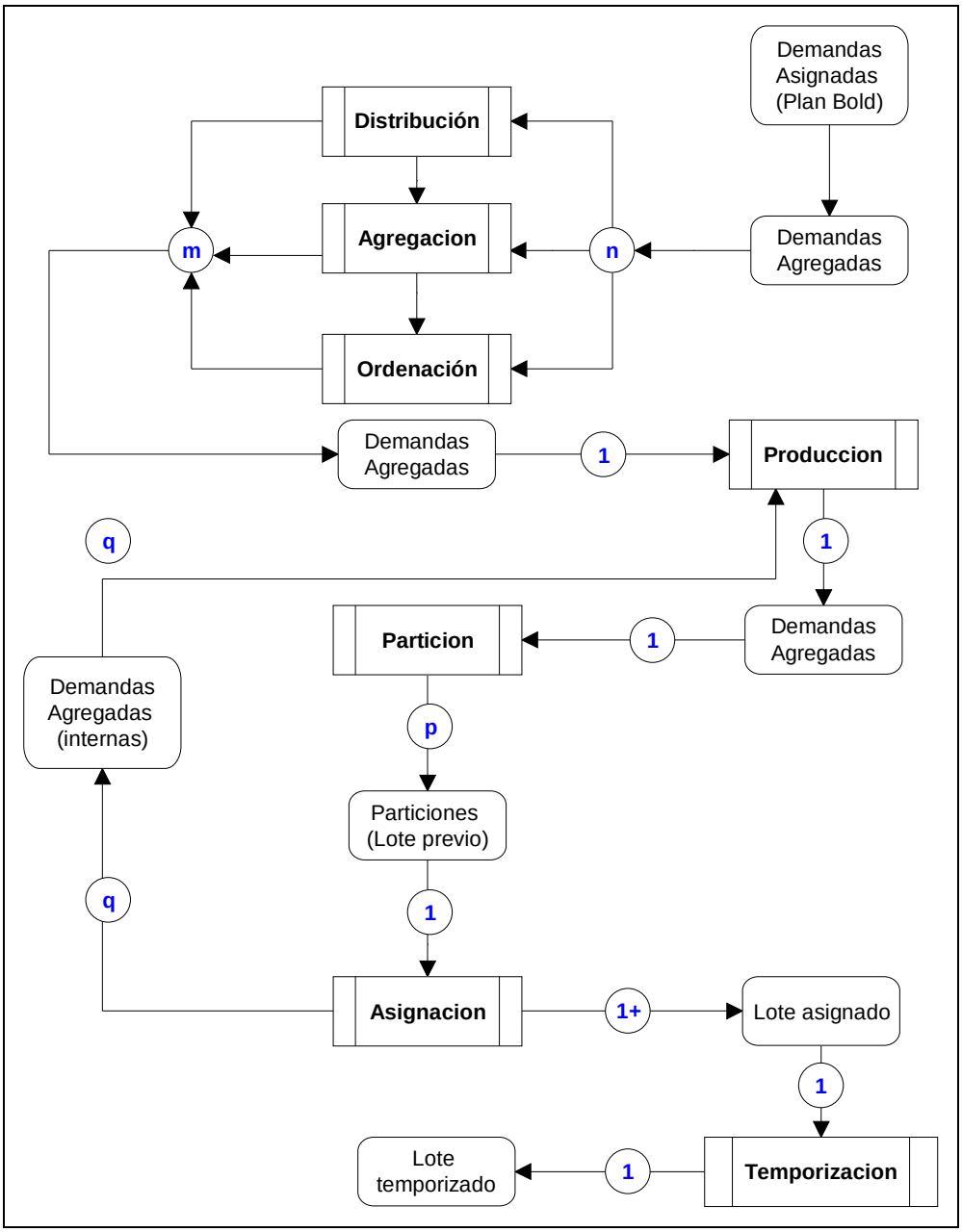


Figura 3 Esquema general del algoritmo

Distribución

Una determinada demanda, habitualmente una previsión de ventas a nivel mensual, *puede distribuirse* en demandas con mayor detalle temporal (quincenal, semanal, diario) para permitir una mayor flexibilidad en la planificación de las mismas, y/o para permitir la adaptación de tales demandas a algún perfil específico, como puede ser una distribución estadística según los datos históricos de ventas reales.

Agregación

Un agregado está formado por un conjunto de demandas. Éstas, pueden corresponder todas a un solo producto o a más de un producto de la misma familia, entendiendo por familia un grupo de productos con ciertas características similares (cualesquiera).

Dicho agregado se identificará por un producto o familia de productos y una fecha de entrega, que puede ser la mínima, la media, la máxima, etc. Estos son algunos de los aspectos que pueden tener en cuenta en el momento de realizar la agregación:

- Dispersión de las fechas de entrega de las demandas que forman el agregado.
- Tamaño mínimo y/o máximo del agregado.
- Fecha de entrega del agregado (la mínima de las fechas de entrega, la máxima, la media, etc.).

En esta etapa se agrupan demandas escogiéndolas según similitud en las características antes mencionadas para formar un agregado, para tratarlas de forma conjunta.

Ordenación

Una vez tenemos un conjunto de agregados, lo ordenamos en función de unos criterios con el fin de dar prioridad a los conjuntos que encabecen la lista. El objetivo de dicha ordenación es facilitar y acelerar la obtención de los planes de producción.

Estos son algunos de los criterios que se pueden tener en cuenta en el momento de realizar la ordenación:

- Instante máximo de inicio (mayor urgencia).
- Prioridad del agregado (demandas, clientes).
- Coste de retrasos.
- Duración del proceso de obtención (planificar antes las demandas "más cortas").

- Tiempos debidos a los cambios de producto.

Producción

Una vez tengamos los agregados ordenados, deberemos decidir si los cubrimos por stock, por compra externa o por producción. Estos son algunos de los criterios que se pueden tener en cuenta en el momento de decidir la producción:

- Stock existente en el momento actual.
- Stock previsto en el futuro (ya sea por compra o por producción).
- Fecha de entrega del agregado.
- Ocupación de los recursos productivos.

Lotificación

La lotificación consiste en partir el agregado en uno o más lotes de menor tamaño. Aunque el agregado esté formado por distintos productos, cada lote estará formado por un único producto. Estos son algunos de los aspectos que se pueden tener en cuenta en el momento de realizar la lotificación:

- Tamaño mínimo de lote.
- Incremento mínimo de lote.
- Tamaño máximo de lote.

Selección del proceso de receta

Consiste en decidir qué proceso utilizaremos para producir cada lote. Estos son algunos de los criterios que se pueden tener en cuenta en el momento de realizar la selección del proceso de receta:

- Prioridad definida por el usuario.
- Coste de producción unitario.
- Productividad del proceso.
- Instantes de disponibilidad de los recursos necesarios.
- Materias primas necesarias y existencias actuales o futuras de éstas.
- Productos intermedios necesarios y procesos para obtenerlos.
- Cantidad y tipo de residuos generados por el proceso.

Asignación

Una vez hemos decidido qué proceso producirá nuestros lotes (ya estemos en el caso de productos finales o intermedios), deberemos decidir, para cada etapa u operación básica,

qué recursos la van a realizar. Estos son algunos de los criterios que se pueden tener en cuenta en el momento de realizar la asignación del grupo de recursos:

- Prioridad definida por el usuario.
- Continuidad.
- Tiempos de cambio.
- Margen previsto.
- Posibilidad de rotura.
- Coste (pts/pcs).
- Cantidad de recursos necesaria (u/h).
- Adecuación de los recursos.

Temporización

Cuando conocemos los recursos que van a realizar cada una de las etapas u operaciones básicas del lote, sólo nos falta temporizarlo, es decir, decidir cuando empezamos a procesar el lote y durante cuanto tiempo. Estos son algunos de los aspectos que se pueden tener en cuenta en el momento de realizar la temporización de un lote:

- Existencia de productos intermedios.
- Existencia de materias primas.
- Enlaces temporales.
- Stocks de productos intermedios.
- Stocks de materias primas.
- Tiempos máximos de funcionamiento ininterrumpido.

Capítulo 5: El modelo del caso a resolver

En este capítulo se expone un caso que nos propone desde la empresa Torrent i Dedeu un consultor especialista en la aplicación práctica de las técnicas DBR (Drum Buffer Rope).*

La característica que nos ha llevado a escoger este caso y no otro es que al traspasar el modelo teórico a BOLD APS y ejecutar el proceso de planificación automática, la solución ofrecida distaba bastante de ser la mejor por causas intrínsecas a la forma en que actúa el algoritmo. La situación es la siguiente:

Demanda:

Producto	**Cantidad**
Pieza 1	450 piezas / dia
Pieza 2	450 piezas / dia
Pieza 3	450 piezas / dia
Pieza 4	450 piezas / dia
Pieza 5	450 piezas /dia

* DBR, **Drum** (tambor), **Buffer** (amortiguador), **Rope** (cuerda). Basan su potencia en identificar el factor limitante (cuello de botella) de una empresa para conseguir mayores beneficios y subordinar todo lo demás a conseguir que ese factor limitante rinda lo máximo posible. Para conseguirlo, se hace que el resto de la fábrica siga el ritmo de producción del cuello de botella, de ahí la palabra **DRUM** (que marca el ritmo de producción). Para evitar imprevistos, este ritmo de producción ha de ser el adecuado para no detener las máquinas en el caso de que hayan pequeñas variaciones en las capacidades productivas generales. Para conseguirlo, se ha de reservar un **buffer** de materia prima que impida en situaciones generales roturas de stock. Es importante que este buffer no sea tan alto como nos permita la producción sino que se ha de encontrar limitado a una cantidad concreta con el objetivo de no saturar almacenes o evitar producciones innecesarias.

Para cumplir con esas demandas contamos con 5 máquinas, cada una de ellas capaz de producir uno de los productos finales requeridos según la siguiente tabla:

Producción:

Recurso	**Capacidad de producción**	**Producen**	**Consumen**
Recurso B	30 piezas / hora	Pieza 1	Intermedio 1
Recurso C	30 piezas / hora	Pieza 2	Intermedio 2
Recurso D	30 piezas / hora	Pieza 3	Intermedio 3
Recurso E	30 piezas / hora	Pieza 4	Intermedio 4
Recurso F	30 piezas / hora	Pieza 5	Intermedio 5

Contamos también con otra máquina (Recurso A) que es la encargada de fabricar los productos intermedios que consumen las otras 5. Esta máquina es capaz de producir 300 piezas de productos intermedios por hora.

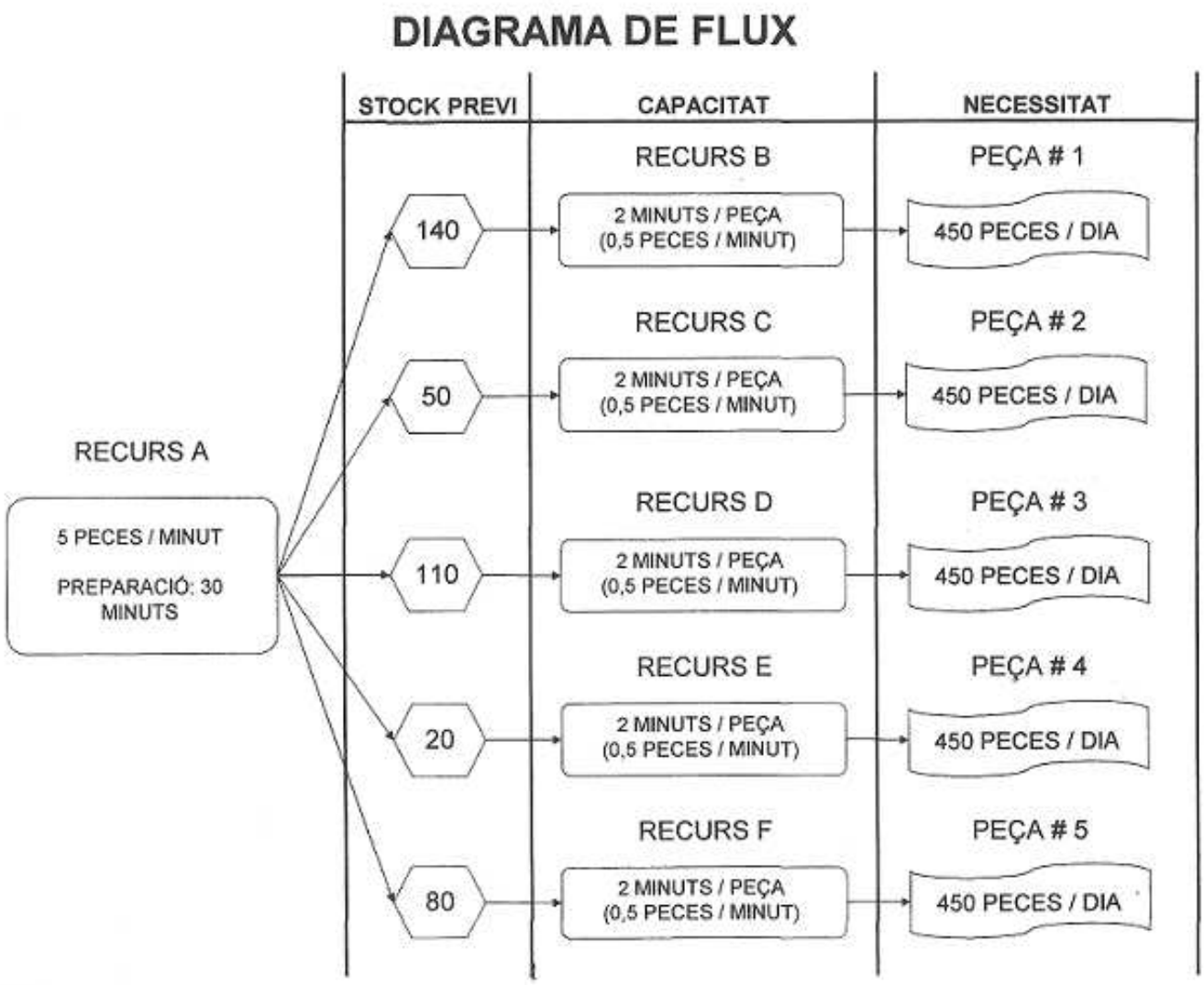
En el recurso A se ha de dar lugar un tiempo de 30 minutos entre cada cambio de tipo de producto que se lleva a cabo en él. Así, si por la mañana decidimos fabricar piezas de Intermedio 1 y por la tarde piezas de producto Intermedio 2, hemos de emplear 30 minutos de transición entre uno y otro, que emplearemos en adaptar la máquina a esta nueva producción.

Al iniciar la producción del día contamos con un stock inicial de productos intermedios, con lo que las máquinas que los consumen pueden empezar a trabajar sin necesidad de poner en marcha el Recurso A.

Stock Inicial:

Producto	**Cantidad**
Intermedio 1	140 piezas
Intermedio 2	50 piezas
Intermedio 3	110 piezas
Intermedio 4	20 piezas
Intermedio 5	80 piezas

Se trabaja en 2 turnos diarios, que es el equivalente a 15 horas productivas al día.



Bases acerca de Teoría de las Limitaciones

Teoría de las limitaciones (Theory of Constraints ;TOC) hace referencia a un conjunto de ideas generales de amplio espectro de aplicación. Eliyahu Goldratt, en su libro "La meta"[2], expone los principios de la aplicación práctica de este sistema de gestión a un entorno productivo. En esta publicación se nos ofrece de forma bastante clara y sencilla de entender una serie de conceptos e ideas ilustrativas acerca de lo que significa la Teoría de las Limitaciones.

Según expone E.Goldratt, las empresas tienen al menos una restricción que les limita en el camino de aproximarse a lo que él llama "La Meta". Como hemos explicado brevemente

[2] Goldratt, E. "La Meta". (Ver nota al pie del capítulo 5)

en capítulos anteriores, la meta de cualquier empresa es ganar dinero, y por supuesto, todas las organizaciones han de tener una limitación que restringe de alguna manera la forma en que lo consiguen, pues si fuese de otra manera, dicha empresa nunca terminaría de aumentar sus ganancias, cosa que sabemos que en el mundo real no sucede.

Las limitaciones pueden tener diferentes puntos de origen, como la propia capacidad de mercado, las características técnicas de nuestro entorno de producción, determinadas políticas de empresa o de gobierno… etc.

La teoría de las limitaciones propone un proceso determinado para mejorar el rendimiento de una organización. Este proceso se divide en cinco pasos:

1. **IDENTIFICAR** las restricciones del sistema total
2. Decidir cómo **EXPLOTAR** las restricciones
3. **SUBORDINAR** todo lo demás a las restricciones
4. **ELEVAR** las restricciones (sólo después de explotarlas)
5. Si se eliminó la restricción, **regresar al Paso 1**

Las empresas tienen al menos una limitación, que podremos identificar, pues son aquellos aspectos de la misma o su entorno que impiden llegar a la meta. Puesto que el ente limitante es la restricción, la empresa consigue su objetivo en la medida que ella se lo permite. Si tenemos una capacidad de producción fantástica pero nuestra limitación (por ejemplo la escasez de mercado) nos impide transformarla en camino hacia la meta, evidentemente deja de tener importancia la capacidad y deberíamos preocuparnos de conseguir un mayor mercado.

Hay que subordinar todas nuestras decisiones a explotar la limitación de forma que nos encontremos en su límite. Es posible que este procedimiento acabe por transformar situaciones que antes de los cambios no formaban parte de la limitación. Como consecuencia de nuestro nuevo ritmo de avance hacia la meta, podría suceder que ahora, la limitación sea otra, y no la que identificamos inicialmente como tal, debiendo nuevamente efectuar un análisis para procurar llevar también al límite (si no se encontrasen ya en él) a las nuevas limitaciones emergentes.

Elevar la limitación hace referencia a superar las restricciones marcadas por su falta de capacidad. Si, por continuar con el mismo ejemplo, existe un déficit de producción que es el único impedimento para aumentar nuestras ganancias, una buena idea sería invertir en aumentar esa capacidad, por ejemplo comprando un mayor número de máquinas. Esta acción se debe de llevar a cabo con cautela pues no en pocas ocasiones, una vez que se explota al máximo la capacidad de los recursos limitantes, la limitación desaparece, o mejor dicho, se transforma en otra.

Para facilitar la comprensión de lo que significa la limitación como tal, explicaremos el concepto de cuello de botella, que es aquel recurso cuya capacidad productiva es igual o inferior a la demanda. En el caso expuesto vemos la existencia de 5 máquinas cuya capacidad productiva es exactamente igual a la demanda que exige el mercado, por lo que todas ellas son nuestro cuello de botella. Por lo tanto deberíamos subordinar todo lo demás (el recurso A) a explotarlas al máximo.

Análisis del caso

Al observar los datos del caso antes expuesto, nos debemos dar cuenta que los recursos con los que obtenemos producto final tienen todos una capacidad de producción exacta a la demanda existente. Todos producen 30 piezas por hora y debemos producir 450 piezas en un día de trabajo, que corresponde a 15 horas de uso de máquinas. 450 piezas / 15 horas = 30(piezas/hora).

También podemos comprobar, con unos sencillos cálculos, que la capacidad de producción del recurso A debería ser más que suficiente para suplir a las otras máquinas, puesto que su capacidad es de 5 piezas por minuto, debería poder producir 4500 piezas en un día. Como cada uno de los otros recursos son capaces de consumir como máximo 450 piezas al día, nos encontramos con que el consumo máximo diario sería 450*5=2250 justo la mitad de la capacidad de producción de la máquina A.

En este breve análisis no hemos tenido en cuenta el efecto de los tiempos de cambio. Los

mínimos necesarios se darían al producir toda la necesidad diaria de cada producto intermedio de golpe, (con el objetivo de minimizar el tiempo no productivo). De esta forma habría que contar con 4 operaciones de cambio. Como cada una es de 30 minutos, tendríamos acumuladas 2 horas de tiempo de cambio, con lo que ahora realmente, contamos con 13 horas en lugar de 15 para producir piezas de producto intermedio en el recurso A. En 13 horas seríamos capaces de producir 3900 piezas, una cantidad bastante mayor a la necesaria (recordemos era de 2250).

Una vez en este punto podemos optar por una de las dos siguientes opciones. Emplear el tiempo productivo sobrante en continuar haciendo piezas más allá de las 2250 necesarias y almacenar ese excedente para el futuro, o permitir que una vez alcanzado este número el Recurso A permanezca detenido lo cual tampoco sería grave si logramos satisfacer la demanda.

El problema de actuar de esta forma, sería que como la producción del recurso A no es instantánea al principio de la jornada, sino que se divide a lo largo de toda ella, con seguridad la máquina que consuma del recurso que produzcamos al principio de la mañana, tendrá disponible materia prima con la que trabajar durante todo el día. Pero aquella otra que consuma del último producto intermedio que hemos decidido comenzar a fabricar seguramente estará detenida durante toda la mañana, a la espera de que el recurso A termine algo para ella.

Por lo que si queremos cumplir los plazos de entrega de los pedidos, hemos de conseguir que los recursos no se encuentren nunca con motivos para detenerse. En caso contrario, al necesitar una jornada de trabajo diario completa para completar nuestro pedido, necesitaremos aguardar al comienzo del turno siguiente para poder enviar la producción al cliente, incurriendo por tanto en un retraso.

De ese modo nuestro principal problema será averiguar la secuencia de producción del Recurso A que es el que “alimenta” a los otros 5, de forma que su producción se adecue a los consumos.

A priori, sabemos que necesitamos 2250 piezas de producto intermedio, y que el recurso A es capaz de producir 3900 teniendo en cuenta las 4 operaciones de cambio. El mayor número de operaciones de cambio que podríamos soportar sería de 15, ya que de ese modo utilizaríamos el recurso A 7 horas y media en tiempo productivo y 7 horas y media

en tiempo no productivo. Estas 7 horas y media son 450 minutos, que a un ritmo de 5 piezas por minuto dan un total de 2250. Cantidad necesaria para que cada máquina se mantenga produciendo correctamente.

Solución teórica del problema

La solución teórica del problema plantea aprovechar las 15 operaciones de cambio que tenemos como margen para producir cada producto intermedio en el momento justo en que son necesarios para no detener los recursos B,C,D,E,F.

La forma de proceder para llegar a esta solución, es producir en cada momento aquello de lo que tengamos menor stock. De esta forma, comenzaremos produciendo Pieza 4. Más complicado es averiguar cuanta cantidad llevar a cabo de forma continúa, es decir, sin plantearnos cual es el próximo con menor stock.

Esa cantidad ha de ser la mínima suficiente como para dar autonomía al consumidor, durante el tiempo que se tarde en fabricar los productos intermedios para las otras 4 máquinas y se vuelva a producir aquel producto intermedio por el que comencé. En este caso, la cantidad puede variar entre 150 y 160 piezas.

Al aplicar la regla que mencionamos con las cantidades especificadas tenemos que la secuencia de fabricación del recurso A es:

Secuencia de producción recurso A: 4 - 2 - 5 – 3- 1- *4 – 2 – 5 – 3 – 1 –* **4 – 2 – 5 – 3 - 1**

La homogeneidad de consumos y producciones de los recursos hacen que la solución propuesta sea producir productos intermedios, siguiendo un ciclo que coincide exactamente con los recursos que tenían menos stock inicialmente.

Decimos que esta solución está basada en un sistema DBR porque hemos identificado el llamado cuello de botella (en este caso hay 5 cuellos de botella que son las máquinas fabricantes de producto final) y hemos subordinado el resto de la fábrica (el recurso A) a mantener al máximo de su capacidad al cuello de botella, pues esta es realmente la capacidad total de la fábrica. Evidentemente por muchas piezas que podamos producir de producto intermedio, no nos serán útiles hasta que sean procesadas por alguno de los cuellos de botella.

Ventajas de la solución propuesta: Se mantienen ocupados todos los recursos durante los dos turnos productivos, regresando al final de la jornada a un estado de stock idéntico al de partida, habiendo conseguido además producir las 450 piezas diarias de cada tipo exigidas.

Inconvenientes: Al efectuar esa política de producción en el recurso A estamos disminuyendo su productividad media diaria, que podría ser mucho mayor y aunque en este caso concreto el excedente no sea aprovechable por estar centrándonos en la producción de un único día, podríamos ir mucho más seguros en cuanto a existencias de productos intermedios a partir del día 2 pues realmente tenemos un exceso de capacidad que se vería reflejado en los stocks, lo que probablemente nos llevaría a pensar en cosas como rediseñar la fábrica para aprovechar ese excedente, o simplemente venderlo a otras empresas.

Además es importante hacer notar el coste derivado de una operación de cambio, que no solo se limita al asociado a la propia operación (una limpieza, por ejemplo), sino que también se ha de tener en cuenta el tiempo que suelen necesitar los recursos para trabajar a pleno rendimiento una vez se ponen en marcha. Por ejemplo, si se habla de hornos, tras una operación de cambio, es posible que los siguientes lotes de producción necesiten de más tiempo de recurso del habitual, pues la temperatura alcanzada no sea la óptima hasta pasados unos minutos.

Plan de pruebas

A pesar de que el caso que pretendemos resolver es muy concreto, definiremos una serie de sencillas variantes del mismo para conocer el comportamiento del algoritmo ante diferentes situaciones. Estos casos que definiremos a continuación nos servirán a su vez para comprobar el correcto funcionamiento del software a medida que se van realizando cambios en él.

Las modificaciones se han llevado a cabo variando las demandas de los diferentes productos, eliminando de esta manera la exactitud entre las demandas y la capacidad de producción que se observa en el caso inicial. El resto de parámetros tales como la velocidad de producción, consumos o tiempos de cambio se han mantenido, para facilitar la transición entre un caso y otro.

Test 0

	Demandas
Pieza	**Cantidad**
1	450
2	450
3	450
4	450
5	450

Test 1

	Demandas
Pieza	**Cantidad**
1	300
2	450
3	390
4	450
5	275

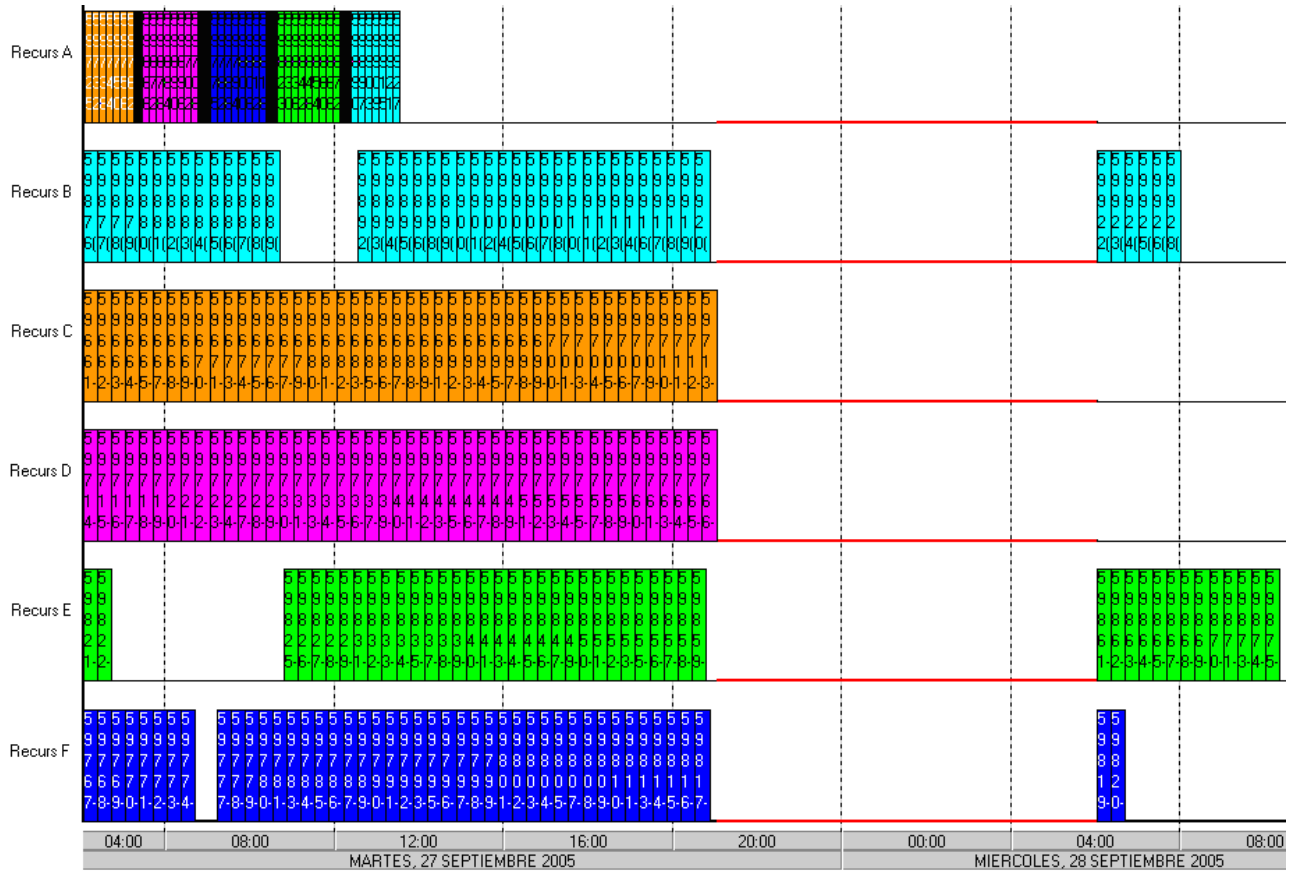
Test2

	Demandas
Pieza	**Cantidad**
1	450
2	450
3	450
4	900
5	900

Solución inicial propuesta por BOLD APS

Una vez se ha comentado el proceso que sigue el algoritmo de BOLD APS para llegar a una solución, mostraremos el resultado obtenido al introducir los datos del problema en la herramienta. En el capítulo 4 se habla del modelo del caso que se ha planteado resolver junto a un breve análisis del mismo.

Lo solución que propone inicialmente BOLD APS minimiza los tiempos de cambio existentes en el sistema. Los tiempos de cambio vienen representados en la imagen como franjas gruesas de color negro. En el caso que tratamos solo se producen en el recurso A. Una minimización de los cambios en ese punto tiene como consecuencia los huecos (franjas sin cajas de colores) que vemos en el recurso B, E y F en la imagen.



Estos huecos, representan a las máquinas sin actividad alguna, es decir, detenidas y son debidos como explicamos anteriormente a que la fabricación de los productos intermedios, excepto en el caso del Recurso C y D, no se desarrolla en el momento en

que es necesaria, sino más tarde.

Con los parámetros del problema propuesto, excesivamente ajustados a la capacidad de producción, nos encontramos que una falta de materia prima en cualquiera de los recursos que llevan a cabo producto final acaba generando un retraso de un día completo en la entrega del material, pues hay que tener en cuenta que la fábrica de estudio no se encuentra en funcionamiento las 24 del día sino que cierra durante el turno de noche. Es por este motivo que en aquellos recursos en los que se produce alguna parada durante el primer día (por muy pequeña que sea) observamos un “hueco” considerable sin producción, correspondiente a las horas nocturnas, en las que no se trabaja.

Durante el proceso de planificación, el algoritmo ha encontrado demandas de productos similares (en cierta forma, están relacionados pues todos los productos, para llevarse a término han de consumir de un intermedio que produce el recurso A) y que se han de entregar en la misma fecha. Estos han sido motivos suficientes para generar un agregado con todas las demandas existentes en el plan, lo que quiere decir que en siguientes pasos, se tratará cada una de las demandas del agregado de forma secuencial, una detrás de otra, resolviéndolas de forma completa. De esta forma, se empieza por atender la demanda de producto 3, y se lanza toda la producción de su producto intermedio. Luego se analiza de producto 2 y se lanza toda la producción de su producto intermedio, y así hasta acabar con todas las demandas del agregado. Esto da lugar a lo que vemos en el diagrama de Gantt.

Se podría llevar a cabo una configuración especial para que el algoritmo no considerase similares a los productos finales y así las demandas fueran a parar a diferentes agregados, pero el resultado sería el mismo pues se trataría agregado tras agregado de forma igualmente secuencial, lanzando las producciones de productos intermedios de igual manera que en el caso explicado anteriormente.

Con un análisis rápido puede parecer una solución sin sentido, pues nos hace incurrir en retrasos y dejar las máquinas (incluido el recurso A) detenidas durante mucho tiempo. Pero hay que hacer notar que en el caso de que nos encontrásemos en un entorno real, al día siguiente seguramente necesitaríamos producir también otras 450 piezas, que podrían ser llevadas a cabo con los productos intermedios fabricados un día antes, pues esta solución aprovecha al máximo la capacidad productiva del recurso A, lo que lleva a que a partir del día 2, no exista nunca ninguna rotura de stock, consiguiendo aprovechar también

al máximo los recursos restantes. Incluso podríamos darnos cuenta de que tenemos un problema de diseño en nuestra fábrica pues hay un recurso con capacidad desaprovechada. Probablemente si comprásemos una máquina adicional de características similares a los recursos que fabrican producto final podríamos absorber la capacidad extra que posee el recurso A y así aumentar nuestra producción total.

Antes de comenzar a modificar la estructura del programa actual se anotan los resultados obtenidos tras efectuar una planificación con los parámetros de las baterías de pruebas que se comentan en el capítulo 5.

De esta forma, el algoritmo de planificación de BOLD APS previa cualquier modificación relacionada con este proyecto ofrecía los siguientes resultados:

Test0

	Demandas	**Algoritmo original**
Pieza	**Cantidad**	**Fecha final entrega**
1	450	28/09/2005 08:40
2	450	27/09/2005 21:00
3	450	27/09/2005 21:00
4	450	28/09/2005 10:40
5	450	28/09/2005 07:00

Test1

	Demandas	**Algoritmo original**
Pieza	**Cantidad**	**Fecha final entrega**
A	300	27/09/2005 16:00
B	450	28/09/2005 10:00
C	390	27/09/2005 19:00
D	450	28/09/2005 09:00
E	275	27/09/2005 15:20

Test2

	Demandas	**Algoritmo original**
Pieza	**Cantidad**	**Fecha final entrega**
A	450	27/09/2005 21:00
B	450	27/09/2005 21:00
C	450	27/09/2005 21:00
D	900	29/09/2005 13:40
E	900	29/09/2005 08:40

Se ha considerado interesante como factor único de medida la fecha final de entrega de cada producto, aunque evidentemente se podrían evaluar otros parámetros como el número de tiempos de cambio y la localización de los mismos, el stock de productos intermedios generados, el tiempo de inactividad,… pero todo ello repercute de una forma u otra en la fecha de entrega final, por lo que a modo de resumen nos sirve para el cometido de este proyecto el valor de medida adoptado.

Capítulo 6: Modificaciones sobre el código

Pasos previos

Antes de realizar cualquier tipo de modificación sobre el código existente, al tratarse de un proyecto de larga duración y que conllevará grandes cambios y periodos de inestabilidad de la herramienta, crearemos lo que se conoce como un branch en el mundo del software de control de versiones.

Un branch se puede ver como una línea de evolución de un módulo. Por defecto se trabaja en un solo branch llamado HEAD. En ese branch hay una evolución del módulo, pero solo existe un antes y un ahora, no pueden haber desarrollos aislados o paralelos. Al usar branches adicionales se puede generar una línea paralela de desarrollo del repositorio CVS. Esta línea paralela se inicia en un estado determinado del repositorio y luego sigue su camino independiente con su propio conjunto de cambios y números de revisiones. Se pueden hacer todos los cambios que se deseen en la línea paralela, así como en la línea de desarrollo original, sin que interfieran entre ellos.

Esta funcionalidad puede permitir que un equipo se encuentre trabajando en características nuevas de interfaz visual, mientras que otro equipo trabaja en bug-fixes de una versión estable.

Cada rama tiene un nombre definido por el usuario, salvo la rama principal que se conoce como HEAD.

En este caso utilizamos el “Tortoise SVN” como programa de control de versiones que es software libre. Creamos un branch a partir del HEAD el día 13 de marzo de 2008 y será en este hilo de evolución donde efectuaremos los cambios relacionados con el presente proyecto, no incluyendo las mejoras en el HEAD mientras no se consiga una nueva versión estable.

Fase 1. Reestructuración del código

Evaluación de riesgos

Esta primera etapa del proyecto cuenta con escasos elementos de riesgo. El más importante es el tiempo dedicado a concluir los objetivos de esta fase.

Como resulta complicado definir cuando se encuentra suficientemente bien organizado un código fuente, o averiguar si existen porciones inutilizadas que se podrían eliminar, es importantísimo respetar el plazo definido inicialmente para la conclusión de este apartado.

Otro factor de riesgo importante es la posibilidad de que una vez reorganizado el código, éste, durante su ejecución, se comporte de manera diferente a como lo hacía con anterioridad, es por ello importante realizar una fase de pruebas al concluir las modificaciones.

División del código

Durante todo el proceso de formación, tanto en la estructura de datos en la que se basa el algoritmo como en el propio flujo, se fue llevando a cabo un trabajo de limpieza y ordenación del código. Cuando este proyecto comenzó, el módulo del algoritmo estaba contenido en 23 ficheros. Un resumen del estado inicial de la organización del código fuente referente al algoritmo se puede ver mediante las siguientes métricas:

Número de ficheros: 23

Número de clases. 45

Número de funciones: 1223

Líneas totales: 44650

Líneas en blanco: 3165

Líneas de código: 35366

Líneas de comentarios: 6540

Definiciones: 9318

Ratio comentarios-código: 0,18

Cada uno de los ficheros contenía más de una clase, llegando a darse el caso de que la declaración de algunas de ellas se encontraba en un lugar mientras las definiciones de sus métodos se encontraban en distintas localizaciones. Por este motivo se creyó necesario como primer paso antes de comenzar con tareas mayores efectuar una división lógica del tipo: un fichero, una clase, ya que de esta forma resulta mucho mas sencilla la navegación por el mar de líneas existente, más aun para una persona a la que este código le resulta ajeno completamente, como es mi caso. De esta forma, después de efectuar esta división acabamos con 86 ficheros en el directorio asociado al módulo del algoritmo.

El nombre de cada uno de los ficheros ha sido escogido en función del nombre de la clase que contienen. Caso especial a comentar es el AlgoMultiProceso que tenía asignados tres ficheros .cpp y tan solo un .h. Esto ha sido así pues el compilador utilizado C++ Builder 6.0 tiene una limitación a la hora de trabajar con ficheros extensos, y de las 40.000 líneas de código pertenecientes al módulo del algoritmo prácticamente la mitad de ellas se encuentran en AlgoMultiProceso, lo que nos ha obligado a dividirlo. Se han hecho agrupaciones lógicas de métodos procurando almacenar en el fichero AlgoMP_Agregador.cpp toda aquel código relacionado con la etapa de la agregación, en AlgoMP_Escritura.cpp aquellas funciones cuyo objetivo final es modificar el estado de un plan de producción, en AlgoMP_Lectura.cpp lo relacionado con hacer consultas sobre el estado de un plan de producción y en AlgoMultiProceso.cpp funciones generales.

Regeneración del fichero de compilación (.mak)

El modificar el fichero de directivas de compilación resultó relativamente sencillo, pues el trabajo consistió simplemente en añadir la nueva lista de ficheros a compilar. Al no haberse creado nuevas dependencias de librerías externas, el procedimiento es rutinario. Sin embargo sí que ha resultado un esfuerzo mayor del esperado el tener que editar los includes a los que hace referencia cada uno de los nuevos ficheros generados.

En c++ como en muchos otros lenguajes de programación, se ha de especificar qué ficheros incluyen qué otros para de esta forma poder identificar donde buscar las clases que se usan si estuviesen definidas en otros ficheros. Para ello se utiliza la sentencia #include. Puesto que antes contábamos con tan solo 20 ficheros, el número de includes posibles entre ellos era escaso en comparación con la nueva situación, donde nos

encontramos con muchísimos más. Dependiendo de que tipo de objetos se utilizasen en cada clase se habían de añadir los includes necesarios.

Los ficheros como Algo_lote.h, que contienen estructuras básicas muy utilizadas en diferentes puntos se han añadido a prácticamente la totalidad de los otros, sin embargo, en contraposición tenemos el caso del Algo_reordenaLotes, clase que se ha creado durante el transcurso de este proyecto y que solo es incluida desde el AlgoMultiProceso.

Eliminación de métodos obsoletos o en desuso

Al navegar por el código para tratar de asimilar el modelo teórico a través de su traducción a c++ usando las clases propias del algoritmo, se pudieron identificar segmentos de código que no formaban parte del ejecutable final y que por tanto eran susceptibles de ser eliminados.

Utilizando el Debugger del entorno de desarrollo c++ Builder 6 para conocer el flujo exacto del algoritmo en circunstancias especiales y así comprender mejor algunas zonas especialmente complejas, se encontró que el propio depurador de código no ofrecía la posibilidad de establecer Break-Points en según que métodos.

En la siguiente ilustración se puede ver como las primeras líneas tienen un punto azul en su margen izquierdo, lo que indica la posibilidad de establecer paradas de ejecución en el código, mientras que aquellas que pertenecen al método GetStockNoAsociadoAProduccionEnFecha no tienen ese punto azul, lo que nos informa que se trata de una fracción de las fuentes que no está presente en el fichero ejecutable, y por tanto es un buen candidato a ser eliminado, aunque siempre es aconsejable verificar que el método no pueda resultar útil en el futuro.

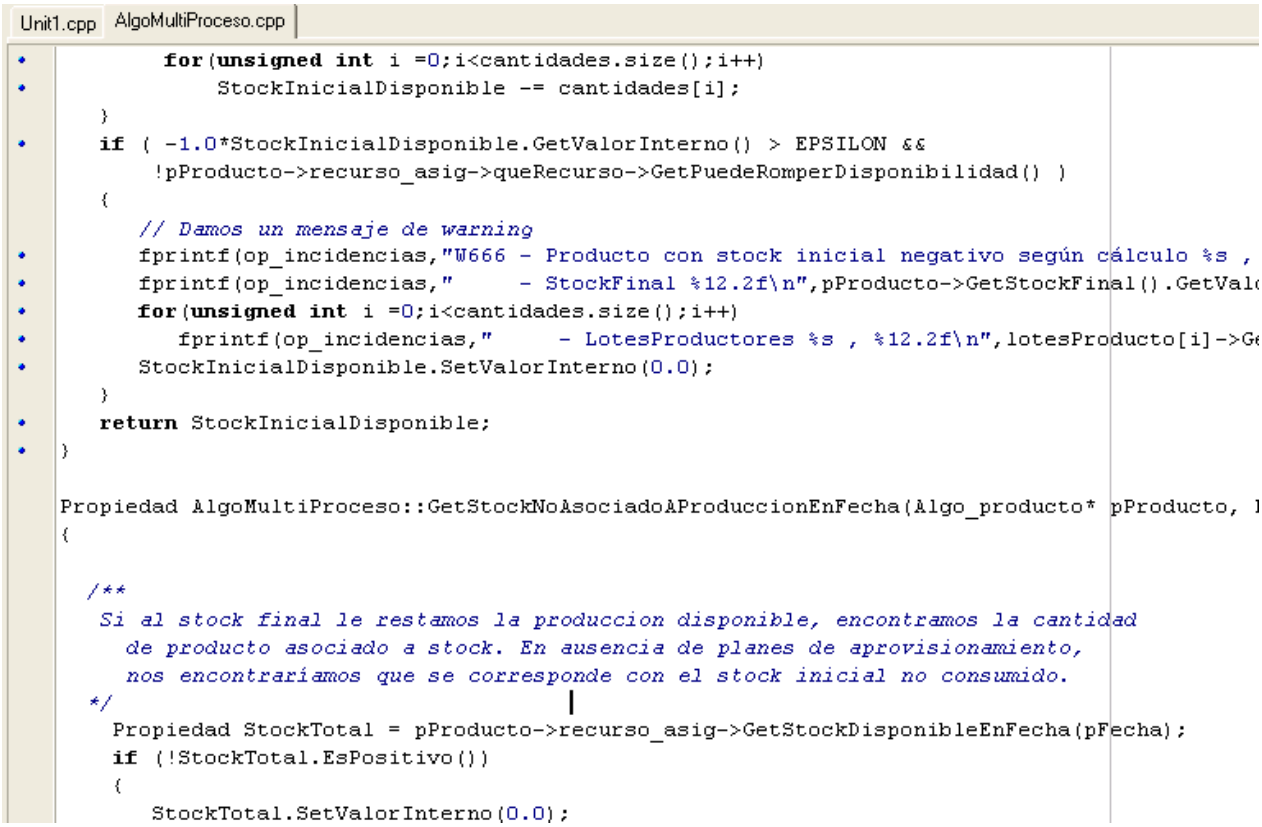


Ilustración 1.Ejemplo de código no accesible

Después de efectuar estos procedimientos, analizamos comparativamente las métricas para examinar su evolución.

	Antes	Después
Número de ficheros	23	84
Número de clases	45	48
Número de funciones	1223	1109
Líneas totales	44650	42404
Líneas en blanco	3165	3155
Líneas de código	35366	31581
Líneas de comentarios	6540	7636
Definiciones	9318	8486
Ratio comentarios-código	0.18	0.24

Siguiendo este procedimiento se han logrando eliminar unas 4000 líneas de código inválido.

La diferencia en el número de clases se debe a las nuevas que se han creado para llevar a cabo la funcionalidad característica de este proyecto.

El ratio comentarios-código ha aumentado, pues algunas funciones que no se estaban utilizando pertenecían a desarrollos inacabados y que quizás pueda interesar ser rescatados algún día, por lo que han acabado formando parte del código comentado.

Se han encontrado 100 métodos que no formaban parte del código ejecutable y como consecuencia de ello y de la nueva estructura del proyecto, vemos un cambio sustancial en el número de líneas de casi todos los tipos que aparecen en la anterior tabla. Consiguiendo así unas métricas de proyecto mejoradas.

Fase de pruebas

Finalizada esta etapa y tras recompilar todo el núcleo de BOLD APS satisfactoriamente, se llevaron a cabo las pruebas comentadas en capítulos anteriores para corroborar que la nueva estructura del código no afectó a la funcionalidad.

Dada la naturaleza de los cambios realizados, es bastante improbable que se produzcan alteraciones sutiles en el funcionamiento del módulo del algoritmo de planificación. De haber una perdida de funcionalidad, debería manifestarse de forma generalizada al ejecutar los cálculos de planificación, siendo fácilmente identificable.

Las pruebas realizadas han consistido en verificar que los resultados obtenidos tras la planificación con la versión original de los ejecutables y con la versión recién compilada eran idénticos.

Al no encontrarse problemas se decidió incluir estos cambios en el HEAD para que de esta forma resulte menos complicada la incorporación de las modificaciones al finalizar el proyecto.

Fase 2. Modificaciones funcionales en el algoritmo

Localización del punto de entrada

Evaluación de riesgos

Esta fase del proyecto es vital para el correcto desarrollo del mismo. Tomar una mala decisión en este punto puede dar lugar a problemas de gran dificultad en fases posteriores, como no tener flexibilidad en la mejora de las soluciones o encontrarnos limitados a mejoras demasiado localizadas.

Existe el peligro también de no encontrar ningún lugar adecuado donde incluir una etapa de cálculo extra que, además, interfiera poco en los procedimientos actuales de ejecución. De ser así habría que abortar el proyecto, o plantearlo desde otro punto de vista diferente.

Desarrollo

El punto en el que centrar el nuevo código se determinó al inicio de la etapa de formación. Al comentar la estructura del algoritmo, se encontró un lugar en que se podían efectuar operaciones sobre la solución sin provocar excesivos problemas para mantener toda la funcionalidad existente.

Justo después de la etapa denominada “Temporización”, tras la cual todos los lotes de un agregado pasan a formar parte del plan, se tiene la oportunidad de extraer los lotes e introducirlos en un orden diferente.

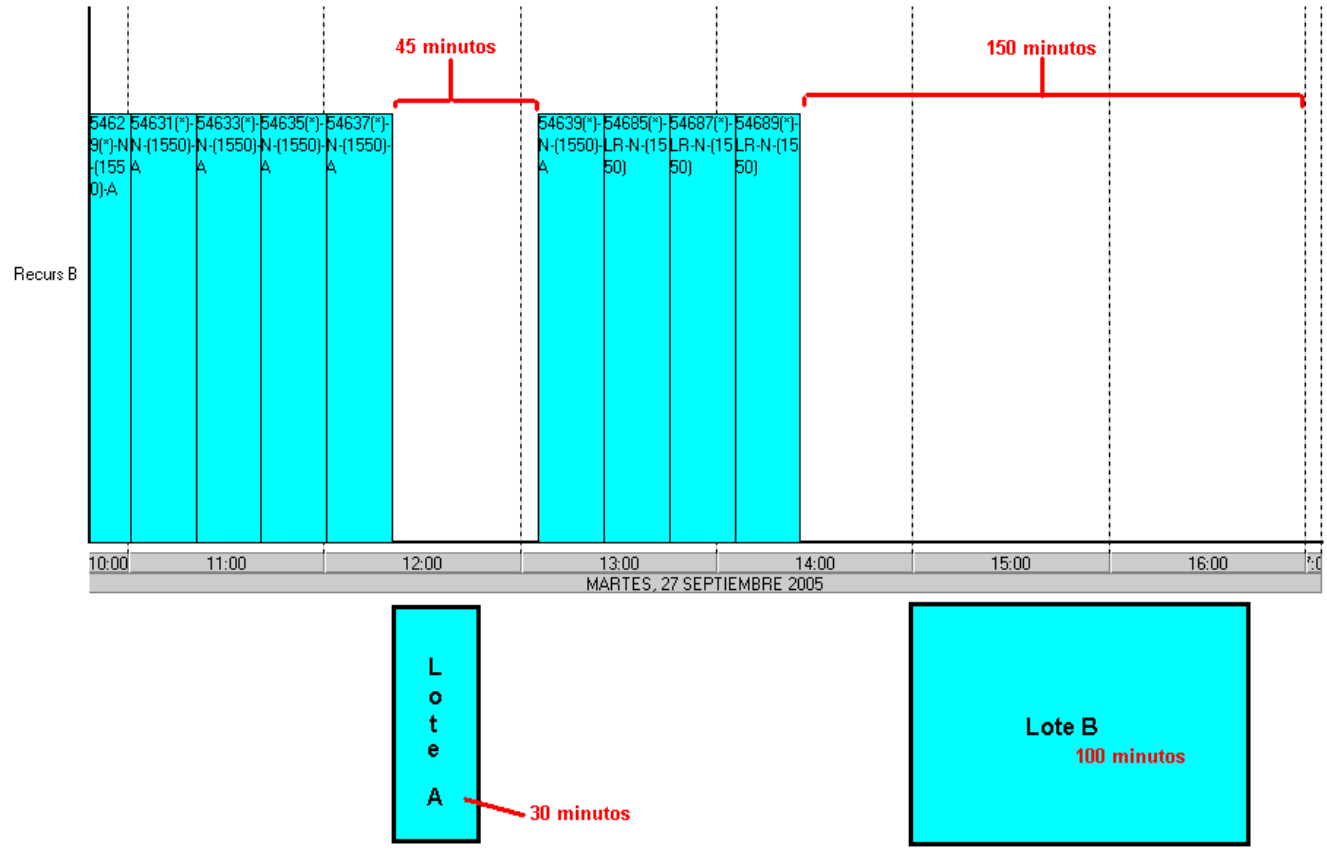
La gran ventaja de esta forma de proceder es que la mayoría de las restricciones que puedan tener los diferentes modelos de fabricación, como cuestiones de estabilidad, gestión de tiempos de cambio, limitaciones de almacenaje, etc… no afectarán, pues se utilizaran métodos que se ocupan de gestionarlas y que pertenecen a otros subsistemas. La idea es utilizar exhaustivamente los métodos de inserción y eliminación de lotes en el plan.

La eliminación actuará sacando la producción de un determinado lote y desocupando los recursos que utilizaba.

La inserción funciona enviando al plan un lote que debe añadir con una cantidad determinada de un producto definido y en el recurso especificado. Las funciones internas

del plan y de otras secciones del algoritmo se ocuparán de re-temporizar este lote en el primer hueco disponible del recurso, siempre y cuando esa posición permita cumplir las restricciones definidas.

De esta forma, el nuevo desarrollo tendría la potestad de decidir en qué orden llamamos al método “AñadirLote” y como causa directa de ello, en qué orden acaban (con bastantes posibilidades) los lotes en el plan. Es importante comprender que el hecho de introducir un lote en el plan antes que otro, no es condición suficiente para que acaben temporizados en ese orden. Esto lo explicaremos con un ejemplo muy simple.



Dada la situación de máquina que se aprecia en la ilustración, si se desean añadir 2 lotes más: uno cuya duración sea de 30 minutos (lote A) y otro de 100 minutos (lote B). Independientemente del orden en que tratemos de introducirlos en el plan, dadas las características de los mismos, encontraremos que siempre quedará temporizado el lote A antes que el B, pues a pesar de darle prioridad al segundo, no podrá ser ubicado en el primer hueco del recurso ya que este le resulta insuficiente. Esto es importante puesto que quiere decir que existen diferentes permutaciones que dan como resultado la misma

solución. Conocer a priori, (en la medida de lo posible) las permutaciones que originan idénticas soluciones podría ahorrar mucho tiempo de cálculo.

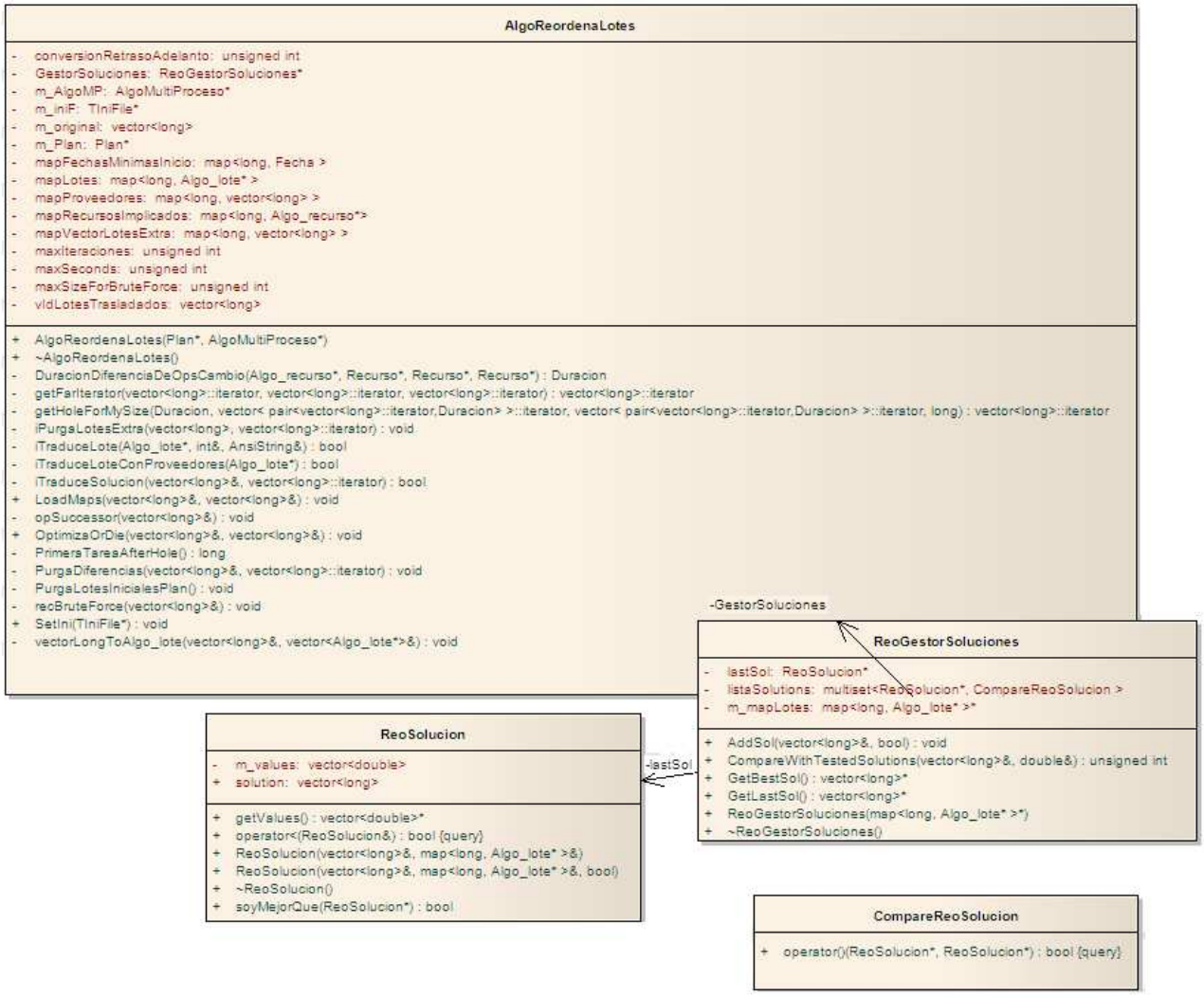
El procedimiento comentado (quitar, reordenar lotes e introducirlos nuevamente en el plan) no se ocupa de variar el tamaño de los lotes que se producen, o de decidir en qué máquina se han de llevar a cabo (en caso de que existiesen varias posibilidades), pues estas cuestiones se han determinado en fases previas de análisis, que residen en módulos ya existentes del algoritmo.

Evaluando el punto de entrada

Para evaluar las posibilidades de este punto de entrada se implementaron un conjunto de métodos básicos en la clase AlgoReordenaLotes, que sería la que alojase el contenido del nuevo desarrollo. Esta clase toma protagonismo después de la fase de temporización y recibe como entrada, una lista de identificadores de objetos “Lote” que hace referencia a la lista de IDs de lotes pertenecientes al último agregado tratado y que fueron incorporados al plan. El objetivo será reordenar esa lista (de la forma que se crea conveniente) y una vez obtenida la nueva secuencia, se introducirá en el plan para plasmar la nueva solución.

Puesto que el cometido de la clase AlgoReordenaLotes será reordenar una lista en base unos criterios que definamos, lo primero que necesitamos es almacenar la lista en algún contenedor. Para ello se ha creado la clase ReoSolucion, que se ocupa de almacenar una lista de ids y una serie de valores accesorios, por si sirvieran de ayuda a la hora de averiguar que solución es mejor que otras o para conocer el tipo de modificaciones sobre la lista original que nos han llevado a ella.

Para poder gestionar las soluciones que se van consiguiendo a lo largo del proceso de cálculo, implementamos la clase ReoGestorSoluciones, cuya principal característica es que permite mantenerlas ordenadas por diversos criterios, de forma que nos resulte cómodo realizar operaciones como obtener la ultima ordenación probada o ofrecer aquella cuya función objetivo sea mejor.



Cuando se ha introducido en el plan una posible solución, necesitamos poder extraer los lotes del plan actual de producción, para luego poder reintroducirlos en un orden diferente. El hecho de eliminar un lote del plan, conlleva por fuerza eliminar todos los lotes que lo proveían o consumían de el. No tendría sentido permitir la existencia de un plan de producción donde existiese un lote que consume de otro que hemos eliminado, o justo lo contrario, si tenemos dos lotes, uno productor y otro consumidor y quitásemos el consumidor entonces el productor dejaría de tener sentido pues nadie sacaría partido de su resultado.

Es por esa razón que se aprovecha la carga de la lista de los lotes, para preguntarle al objeto “plan” la lista de los lotes proveedores de cada uno de los que nos pasan y almacenarla en un objeto map de la stl. Resulta fundamental, consultar de nuevo los proveedores de los proveedores, y así sucesivamente en cascada hasta conseguir todo el

árbol de precedencia de lotes. Para ello se estudió la posibilidad de utilizar un procedimiento recursivo, pues la naturaleza del problema nos ofrece una forma sencilla de resolverlo de este modo, pero como se busca la máxima eficiencia posible para no aumentar el tiempo de cálculo al que está acostumbrado el cliente, se ha acabado utilizando una pila para almacenar aquellos lotes que nos quedan pendientes de revisión y así recorrerlos todos. De esta forma se ha conseguido un recorrido en amplitud iterativo. A pesar de que en el problema que estudiamos, la profundidad máxima en el recorrido de búsqueda de padres es 2 niveles y las diferencias en tiempo de cálculo entre los métodos iterativos y recursivos se agudizan al aumentar la profundidad, se ha procurado evitar el uso de la recursividad en previsión de, en un futuro poder tratar problemas de mayor envergadura.

Es importante detallar que aparte de almacenar los ID de los lotes, se guardan punteros a los propios objetos, haciendo más complejas las estructuras de datos que soportan la información básica de trabajo del AlgoReordenaLotes pero aumentando considerablemente la velocidad de ejecución, pues al tener punteros a los objetos, el trabajo de encontrarlos dentro del plan en cada momento a partir del ID resulta innecesario. Además, como utilizamos MAPS para almacenar esta información, la velocidad de acceso a los punteros a objeto es independiente del tamaño de la lista de objetos almacenados, lo que hace mucho más escalable el código a situaciones más complejas.

De este modo se tiene accesible de una forma rápida desde AlgoReordenaLotes la lista de proveedores para cada lote que forma parte de la solución y punteros a cada uno de los objetos. Ahora, ya se está en disposición de coger esa lista inicial que llega al módulo que se trata de describir, y recorrerla eliminando cada uno de los lotes y todos sus proveedores del plan.

Utilizando el procedimiento inverso se consigue un método que inserte lotes, para lo cual la única peculiaridad que se ha de atender es la de insertar antes de cualquier lote a todos sus proveedores, pues caso de hacerlo al revés, estaríamos dando lugar a una incoherencia en la que un lote se trata de lanzar antes de tener materia prima para llegar a ser ejecutado.

Una vez se tienen procedimientos capaces de sustraer y añadir lotes al plan, se hace una prueba para verificar su correcto procedimiento. Para ello, se carga toda la estructura

necesaria del nuevo módulo y se eliminan todos los lotes de la solución inicial del plan, para luego volver a introducirlos en el mismo orden exacto en que se encontraban. Si todo es correcto se debería obtener una solución idéntica a la que se ofrecía antes de pasar por el módulo AlgoReordenaLotes.

Se observa que una vez efectuado el proceso anterior, todos los lotes aparecen temporizados varias horas más tarde del lugar que originalmente ocupaban. El motivo era la necesidad de llamar a unos métodos que se ocupan de gestionar la disponibilidad de los recursos en el tiempo. Se había eliminado la producción del plan, pero sin haber liberado los recursos para que pudiesen volver a efectuar tareas. Una vez resuelto esto se consiguió llegar al punto de partida después de haber eliminado y añadido de nuevo todos los lotes en el mismo orden.

Como simple ejercicio de pruebas se trató de desordenar la secuencia, introduciendo como nueva posibilidad la ordenación inversa a la inicial. El hecho de escoger esta secuencia inversa se debe a la existencia de un método llamado ::reverse en las clases que utilizamos para almacenar las listas, y cuya función es exactamente la que parece. Cambia el orden de los objetos que contiene poniendo al final de la lista los objetos que inicialmente se encontraban al principio.

Obviamente el resultado fue un plan sin demasiada calidad, pero el procedimiento funcionó. Se obtuvo un plan de producción donde, aunque tarde, se llevaba a cabo toda la producción. Evidentemente un cambio tan simple no alteró la cantidad de tiempos de cambio producidos.

En este punto ya se contó con las funciones que ayudarían a probar las posibilidades de mejora de esta forma de proceder. Manualmente se forzó la reordenación de los lotes correspondiente a la solución que se acepta como adecuada para el caso de estudio expuesto, para averiguar si el hecho de “tan solo” eliminar lotes, reordenarlos de la forma adecuada, y volverlos a introducir contenía potencial de mejora. Efectivamente, el sistema fue capaz de eliminar el contenido de la lista que se le pasaba e introducirla siguiendo el orden que manualmente le habíamos especificado, consiguiendo así llegar a la solución considerada como buena. El trabajo de aquí en adelante consistirá en encontrar un proceso automático que transforme la lista de lotes inicial en una lista ordenada de forma adecuada para atender todas las demandas en el tiempo previsto.

La búsqueda de la solución

En este apartado se explicará el camino seguido para obtener soluciones adecuadas al problema, de forma automática.

Como se verá, se ha intentado avanzar en diferentes direcciones. Comentaremos las razones por las que se ha escogido cada una, y de que forma se han llegado a descartar. Finalmente detallaremos el método que más ha cumplido las expectativas del proyecto y sus principales características.

Evaluación de riesgos

Esta fase del proyecto es la principal y donde más problemas nos podemos encontrar.

Dada la cantidad de sistemas, filosofías y procedimientos existentes para obtener soluciones, nos es imposible probarlos todos en un tiempo adecuado. Es por ello que con sencillos y rápidos ensayos hemos de decidir cual escoger para desarrollarlo en profundidad, de lo contrario, corremos el riesgo de perder demasiado tiempo con alguna opción que a la larga será inútil. Esta forma de actuar, a su vez, tiene el problema de que no dedicando gran atención a los métodos que en un principio puedan parecer poco interesantes, perdamos opciones que estudiando con detenimiento sean realmente provechosas.

El riesgo, como ya viene siendo normal en casi todas las fases del proyecto, está relacionado con el tiempo. En este caso el que se le dedica a cada forma de mejorar las soluciones, y también el que no se le dedica. Hay que analizar cada sistema lo suficiente como para conocer su potencial, pero sin invertir demasiado como para impedir el análisis de los demás.

Además es muy posible que los sistemas que encontremos para mejorar soluciones utilicen demasiado tiempo de cálculo, lo que para este caso es inaceptable.

Función de evaluación

La forma en que, a partir de una solución, se obtiene un valor numérico que indique la bondad de la misma es a través de la llamada “Función de Evaluación”. El determinar en base a qué criterios se debe de guiar es clave. No se trata sólo de definir qué es malo y qué bueno, sino también de decidir cuánto de bueno o malo aporta cada aspecto de cada

solución al valor final de evaluación.

Para este proyecto, se empleará una función de evaluación general que utilizará como criterio principal la cantidad de producto retrasado, es decir, las unidades que no se han podido entregar en la fecha solicitada por el cliente. Además, se ha de penalizar el hecho de alejarse de la fecha de entrega prevista, pues no lo es mismo, sufrir un retraso de un día que de una semana.

Como con total seguridad surgirán soluciones cuyos valores de retraso sean idénticos, definimos un segundo criterio, para discriminar los casos de empate. El “makespan”, un valor que indica el tiempo transcurrido entre el inicio de fabricación del primer lote y la finalización del último de ellos, o lo que es lo mismo, el tiempo total que hemos necesitado para servir todos los pedidos. Aquellas soluciones con un makespan menor las tomaremos como mejores.

Recorrido enumerativo (brute force), y aleatoriedad.

Lo primero que se planteó, para evaluar la eficiencia de los métodos de inserción y eliminación de lotes en el plan, fue la posibilidad de enumerar todas las permutaciones factibles a partir de la secuencia original.

Para conseguirlo, se desarrolló un método denominado “opSuccessor” que es capaz de, a partir de una secuencia de ids, generar la siguiente, de forma que al utilizar esta función un número adecuado de veces, se llegue de nuevo a obtener la lista inicial, sin haber repetido ninguna secuencia.

Hay que hacer notar que el número de posibles ordenaciones, depende únicamente del tamaño de la secuencia a ordenar, siguiendo la función factorial. De esta forma, con una situación inicial compuesta por 8 lotes, se podrían llegar a generar 40.320 soluciones diferentes. Lo cual, si no se dispone de una forma de descartar algunas de ellas, puede suponer una pérdida colosal de tiempo el sólo hecho de tratar de insertarlas en el plan y luego eliminarlas. Más aún si se tiene en cuenta que el problema que se toma como guía cuenta con 400 lotes, lo que da lugar a una cantidad de posibilidades cercana a $6.4*10^{868.}$ Reconocer la magnitud de este número nos lleva a la necesidad de averiguar cuantas de ellas se pueden llegar a poner a prueba en un tiempo razonable.

Como comentamos en capítulos anteriores, uno de los objetivos es que no se vea aumentado enormemente el tiempo de cálculo, manteniendo la posibilidad de obtener

soluciones adecuadas en pocos minutos.

Sirviéndonos de la función opSuccessor, se procede a determinar el número de pruebas que se pueden realizar en un tiempo de un minuto. Para ello se desarrolla un método llamado “recBruteForce” que se ocupa de probar un numero N de posibilidades, anotando el tiempo que le ha tomado hacerlo.

Los resultados obtenidos se muestran en la siguiente tabla:

Iteraciones	Tiempo
100	15 segundos
500	75 segundos
1000	160 segundos

Al ver estos datos se descartan completamente las posibilidades de utilizar el procedimiento enumerativo, o aleatorio a través del espacio de soluciones para llegar a mejorar la propuesta inicial. Pues siendo tan reducido el número de pruebas que se pueden realizar en un tiempo de unos dos minutos, resulta más que remoto el llegar a buen a puerto dado el enorme número de posibilidades. A pesar de ello, se desarrollan métodos que permiten evaluar las soluciones propuestas, a través del plan, tanto para facilitar otras posibles vías, como para corroborar que el camino de probar soluciones de forma aleatoria no es el adecuado.

Contando con la función de evaluación que vimos anteriormente, se pone en marcha el recorrido aleatorio, para conocer el nivel de mejoría conseguido.

Tras diversas pruebas, con diferentes números de iteraciones, no obtenemos ninguna mejora en el makespan, aunque sí en cuanto a la cantidad retrasada. Sutiles cambios en algunos lotes producían una levísima mejora, convirtiendo el retraso de algunos lotes de 8 horas a 7 horas 58 minutos.

Conclusiones*:* Conociendo el enorme espacio de soluciones, el número de intentos que podemos llevar a cabo sin afectar el tiempo de respuesta del software y los resultados

obtenidos mediante las pruebas realizadas, se descarta completamente tanto el procedimiento de enumerar todas las posibles soluciones como el de recorrerlas de forma aleatoria. Resulta fundamental conseguir un grado de mejora significativamente mayor a los obtenidos hasta el momento.

Meta heurísticas, algoritmos genéticos

Se decide estudiar la posibilidad de aplicar algún tipo de meta heurística, pues este tipo de procedimientos suelen permitir el tratamiento de problemas numéricos muy grandes, con buenos resultados.

En un primer momento se considera especialmente interesante el método llamado “Swarm Optimizacion” que consiste en simular el procedimiento que llevan a cabo colonias de animales como las hormigas, abejas o sardinas para encontrar alimento, o huir de depredadores. Explicado de forma breve, este sistema se basa en la realización masiva de toma de decisiones muy simples, llevadas a cabo entre conjuntos reducidos de posibles soluciones. Se parte de una situación inicial con una población de soluciones distribuida de forma aleatoria, y aquellas que se encuentran próximas entre sí, se informan mutuamente acerca de su estado.

–Por aquí hay comida, por aquí no. Huelo a soluciones muertas por esta zona….

Estas indicaciones simples, sirven a las soluciones cercanas para decidir como desplazarse por el espacio hacia otra diferente y para dejar huellas en el camino, de forma que otras soluciones puedan seguirlo, si las huellas eran positivas, o evitarlo si eran negativas.

En nuestro caso, el poder evaluar si una dirección es un buena o mala, pasa por introducir la solución en el plan, lo que resulta demasiado costoso si pretendemos emplear una técnica de este tipo, que precisa de funciones de evaluación casi inmediatas. Por esta razón se intentará encontrar otra vía de desarrollo alternativa.

Dentro de las meta heurísticas, otra posibilidad interesante se encuentra en los algoritmos genéticos, que basan su potencial en la posibilidad de mezclar dos permutaciones y, a partir de ellas, obtener una tercera, que haya heredado las mejores características de las dos originales. Para ello se necesitan:

- Población inicial: Un conjunto de soluciones iniciales, tomadas al azar, a partir

de la cual surgirán las nuevas soluciones.

- Función de evaluación: Nos permitirá conocer qué soluciones son mejores, para trabajar a partir de ellas, y descartar las peores.
- Operadores de cruzamiento: Encargadas de recibir dos soluciones, y ofrecer una tercera, generada a partir de las dos primeras.

Ya hemos explicado, tanto un método para obtener la población inicial como la función de evaluación, con lo que sería necesario desarrollar los operadores de cruzamiento, para poder implementar un algoritmo genético básico.

Se comenzó a desarrollar operadores sencillos, que generaban soluciones a partir de fragmentos de las soluciones de entrada y se realizaron pruebas, creando un método que escogiese dos candidatos del gestor de soluciones cuyo valor de la función de evaluación fuese mejor que la media, procurando escoger las mejores, pero con cierto de grado de laxitud. El hecho de no escoger directamente las mejores soluciones para generar las siguientes tiene como fin el fomentar la diversidad de características en todo momento.

Con dos candidatos seleccionados se efectuaba el cruzamiento, y la solución resultante se introducía en el plan, se evaluaba y se añadía al gestor de soluciones para formar parte de la población. Al transcurrir N iteraciones se recuperaba la mejor solución encontrada hasta el momento.

Los resultados obtenidos fueron idénticos a los conseguidos mediante la selección completamente aleatoria de soluciones. Un estudio más a fondo sobre los algoritmos genéticos nos revela, que uno de los factores fundamentales para conseguir una buena evolución, es conseguir una población inicial bastante variada para así evitar limitarnos a explorar una zona muy concreta de ella. Como sólo podemos realizar unos pocos miles de pruebas, nuestra población inicial se ha de medir en decenas, o cientos (de soluciones), lo que evidentemente no es suficiente para obtener una muestra significativa de permutaciones dentro de las posibles. Además la existencia de múltiples permutaciones equivalentes nos dificultan enormemente la tarea de partir de una población variada, pues aunque la lista de IDS esté realmente en un orden diferente, es posible que al introducir los lotes en el plan, la solución resulte ser idéntica, reduciendo la posibilidad de cambio en el resultado de los cruzamientos.

Conclusiones: La imposibilidad de generar una población inicial lo suficientemente grande y variada como para obtener buenos resultados nos hace descartar la opción de utilizar un algoritmo genético para resolver el problema. Además si consiguiésemos obtenerla, el número de iteraciones que requiere un algoritmo genético es abrumador desde el punto de vista de nuestra limitación en cuanto al tiempo de cálculo.

Aún conociendo estas limitaciones, el potencial de los algoritmos genéticos es inmenso aunque para explotarlo es necesario invertir un tiempo considerable en optimizar los procesos de inserción-eliminación de lotes en el plan, así como idear una forma rápida de distinguir entre listas de identificadores equivalentes. Todo ello cae, de momento, fuera de los objetivos del presente proyecto, pero se tendrá en cuenta esta línea de desarrollo como fuente de investigación en el futuro.

Mejora en las funciones auxiliares

Sabiendo que la totalidad del tiempo de cálculo se invierte en las operaciones internas del plan que eliminan o insertan lotes, se considera fundamental invertir un tiempo extra en minimizar el número de veces que se efectúan dichas acciones para poder continuar explorando nuevas vías de desarrollo de forma más cómoda.

Para ello se ha desarrollado una funcionalidad importantísima en el transcurso de este proyecto y que ha aumentado de forma significativa la velocidad de ejecución. Se ha aprovechado el gestor de soluciones, que, recordemos, se utiliza fundamentalmente para almacenar cada una de las soluciones que se van generando a través de las iteraciones y mantenerlas ordenadas conforme a su validez, para, llegado el momento obtener la mejor de todas ellas.

Modificando ligeramente el funcionamiento de esta clase, mantenemos la lista ordenada por orden de inserción, en lugar de por calidad y guardamos un puntero a la mejor solución en todo momento, pues al fin y al cabo es la que nos interesa.

Este cambio a la hora de almacenar las soluciones tratadas nos aporta beneficio justo a la hora de insertar una nueva permutación, pues nos permite analizar las diferencias entre ésta y la solución que contiene el plan actualmente (la ultima en ser introducida en el gestor), con el fin de determinar el punto a partir del cual existen diferencias entre ambas.

La utilidad que se desprende de conocer este dato, es que, al resultar inútil eliminar unos

lotes del plan que luego serán introducidos en la misma posición, solamente efectuaremos dichas operaciones sobre los lotes necesarios. Por ejemplo, si la última solución insertada en el plan fue A-B-C-D-E-F y la nueva solución es A-B-C-D-**F-E,** sólo hay diferencias a partir de la quinta posición, lo que nos ahorra extraer y reinsertar los cuatro primeros lotes. Si se trata de extrapolar este ejemplo al caso que se propone, con 400 lotes diferentes, el número de operaciones que nos hemos ahorrado resulta importantísimo.

Una vez implementada esta nueva forma de proceder, se obtienen los siguientes resultados:

Iteraciones	Tiempo
2.500	6 segundos
5.000	11 segundos
10.000	22 segundos
20.000	45 segundos

Como reflejan las tablas, la diferencia es más que fundamental. Se ha multiplicado por 30 el número de iteraciones de las que disponemos para encontrar una buena solución.

Algoritmo especializado (basado en Teoría de las Limitaciones)

Tras haber seguido de forma infructuosa varios caminos y con la limitación del tiempo siempre presente, se plantea la necesidad de escoger un camino conservador hacia la solución. Analizando las carencias existentes del algoritmo actual, nos proponemos resolverlas atacándolas de forma directa, en lugar de dar el rodeo a través de meta-heurísticas que con ayuda de la aleatoriedad y la acumulación de probabilidades positivas podrían acabar por hacerlo sin tanta intervención mental humana, ofreciendo una mayor flexibilidad a la hora de resolver problemas de mayor variedad.

De los análisis que se han llevado a cabo durante las diferentes etapas de este proyecto, se desprende que reordenando de diferentes formas los conjuntos de lotes que el proceso de agregación ha formado, se pueden obtener importantísimas mejoras. Algo que también conocemos bien después de analizar algunas soluciones del caso que tratamos es que

nos conviene no detener ninguna máquina que fabrique producto final, pues estas son las que determinan la fecha de entrega del pedido completo. Es por ello que nuestro procedimiento de mejora de soluciones ha de buscar la máxima utilización de estos recursos.

En el caso que estudiamos, el único método del que disponemos para que las máquinas produzcan sin detenerse es asegurar la permanente disponibilidad de materia prima para continuar activas.

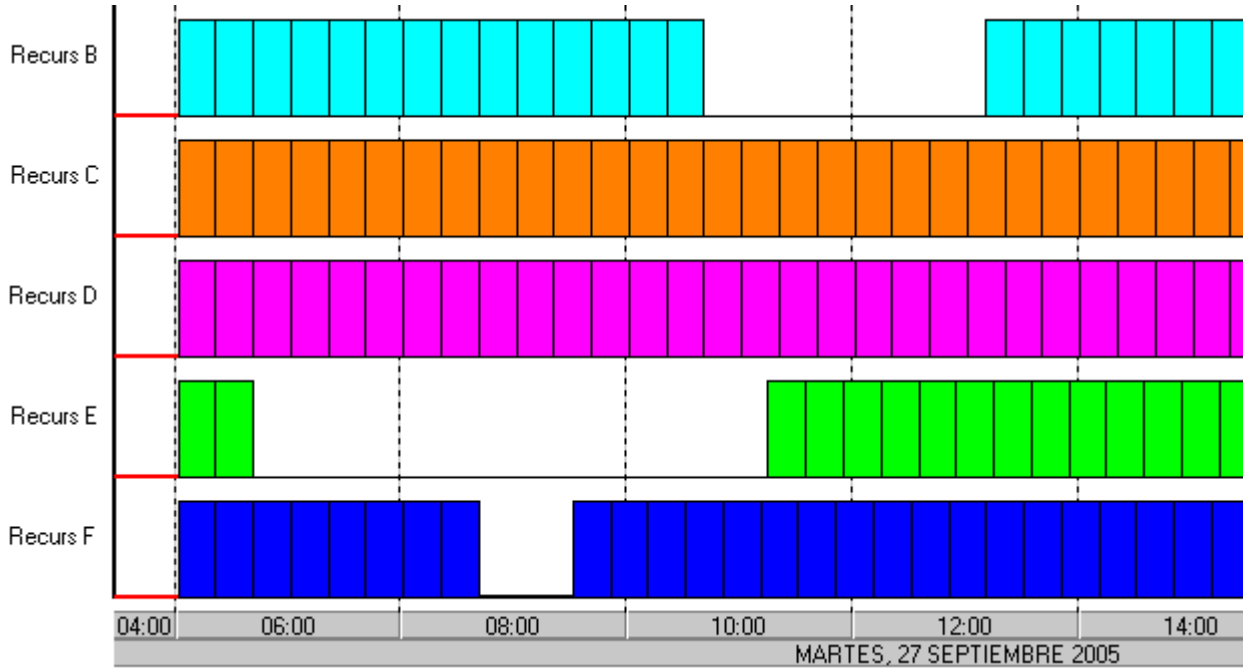
Dada una solución cualquiera (por ejemplo, la inicial, propuesta por el algoritmo original de BOLD APS), lo primero que se hace en el nuevo procedimiento es separar los lotes que consumen materia prima de la existente en stock de los que consumen de la materia generada directamente por el recurso A, pues se considera que los primeros no dan ningún tipo de problemas de decisión, pues siempre se sitúan los primeros en cada recurso. Es lógico consumir de forma preferente de stock antes de ordenar la producción de nuevo producto intermedio. De esta forma, el nuevo algoritmo propondrá siempre y sin invertir tiempo de cálculo extra el mantener las máquinas ocupadas en consumir el stock de materia prima (siempre que sea necesario). Con ello, teniendo en cuenta la materia prima con que se cuenta al comenzar el primer turno de trabajo tenemos que las máquinas disponen de la siguiente autonomía:

Producto	**Cantidad**	**Autonomía**
Intermedio 1	140 piezas	140/30 = 4,6 horas
Intermedio 2	50 piezas	50/30 = 1,6 horas
Intermedio 3	110 piezas	110/30 = 3,6 horas
Intermedio 4	20 piezas	20 / 30 = 0,6 horas
Intermedio 5	80 piezas	80 / 30 = 2,6 horas

Con esta visión del estado inicial del problema, se puede ver fácilmente que, salvo se comience a producir producto intermedio 4 antes de 0,6 horas, tendremos la máquina que

lo consuma detenida de forma irremediable.

Tras la primera fase, en la que hemos asignado las materias primas de stock a los recursos consumidores, nos encontramos ante un estado en el que todas las máquinas consumen hasta agotar existencias, mientras en el recurso A, se producen productos intermedios de forma que se minimicen los tiempos de cambio.



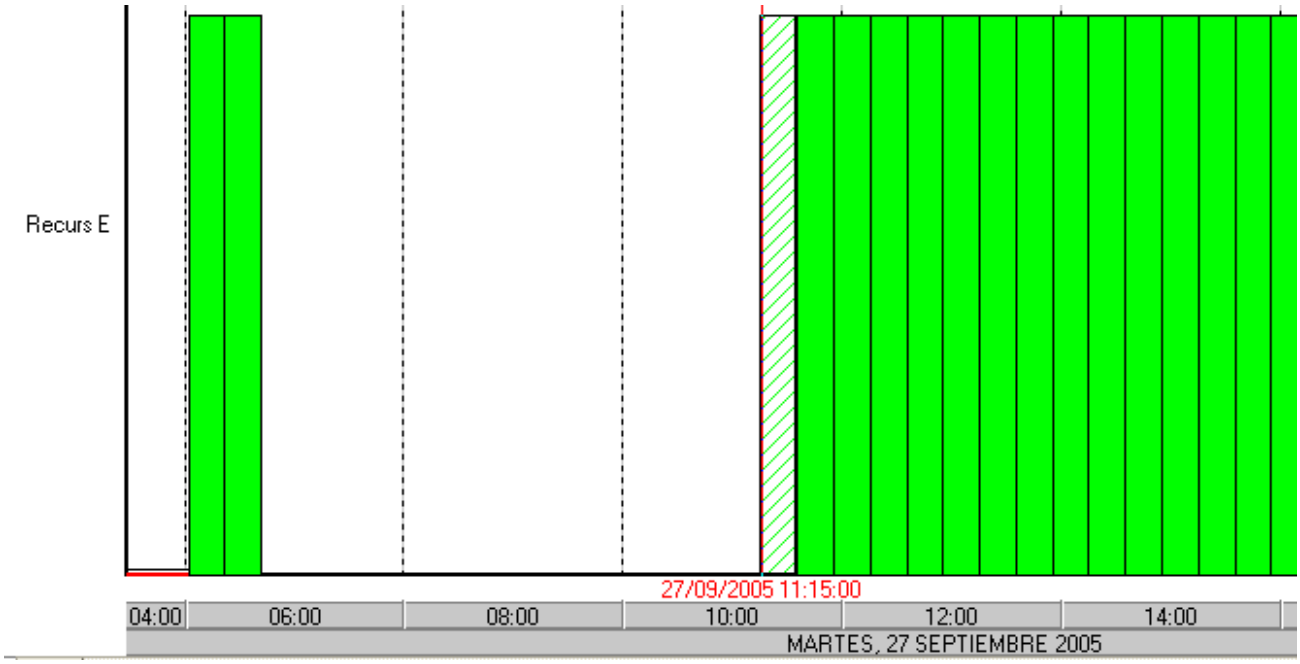
Aplicación de la teoría de las limitaciones al nuevo algoritmo

En el capítulo 5 se exponen las bases de la teoría de las limitaciones y se muestra un proceso que consta de 5 pasos cuyo objetivo (aplicado a nuestro caso) es mejorar el rendimiento global de una fábrica.

En este apartado, se explicará como se ha intentado aplicar este proceso a la hora de diseñar el algoritmo de mejora de soluciones.

El primero de los cinco pasos es **IDENTIFICAR** las restricciones del sistema total.

> Para el caso que tratamos, dada una determinada solución, la restricción inmediatamente limitante, se encontrará en el recurso que tiene el primer lote que se ha visto obligado a retrasarse por falta de materia prima. En este caso, esta situación se produce en el recurso E.



En la imagen, vemos seleccionado un lote que comienza a producirse a las 11:15, cuando la máquina necesaria para llevarlo a cabo se encuentra disponible desde las 6:40. Este retraso, producido por falta de materia prima es el que trataremos de solventar en primer lugar.

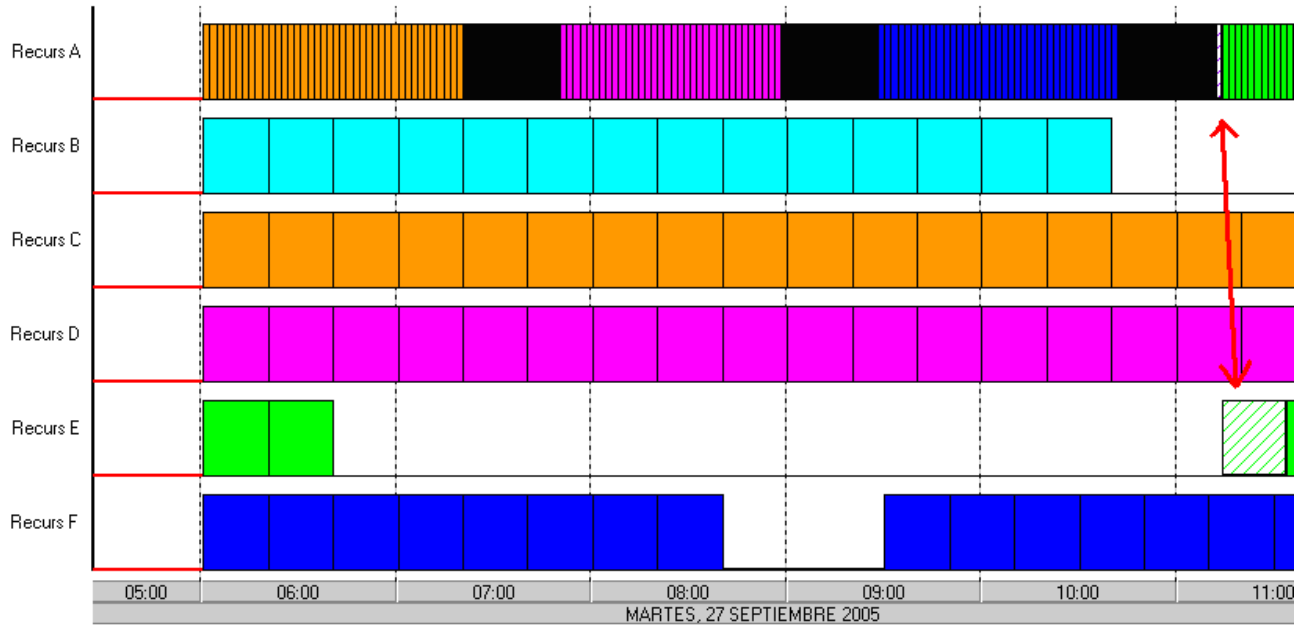
El siguiente paso es decidir cómo **EXPLOTAR** las restricciones.

Como la situación que causa el cuello de botella es la falta de materia prima de un tipo determinado en un instante de tiempo, podríamos decidir entre comprar producto de este tipo o, si somos capaces, lanzar su producción en nuestro propio entorno. En el caso que estudiamos, la primera no es una opción disponible, por lo que trataremos siempre de producir la materia prima necesaria.

Como hemos comentado en numerosas ocasiones, nuestro poder de decisión se encuentra principalmente en determinar qué es lo que produce la máquina A. Es por ello que debemos relacionar el lote retrasado con el lote productor de su materia prima, con el fin de tratar de adelantarlo en el tiempo, si es posible.

El lote señalado en la siguiente imagen, es el que produce la materia prima necesaria para que se pueda llevar a cabo el lote seleccionado en el recurso E. Si adelantáramos la producción del lote proveedor, es seguro (pues no hay mas restricción que la falta de producto intermedio) que se adelantaría la producción del

lote que hemos señalado para tratar.



Posteriormente se ha de **SUBORDINAR** todo lo demás a la restricción.

Esto quiere decir, hacer todo lo posible para que el recurso con falta de materia prima deje de ser el recurso limitante. Como hemos explicado en el paso anterior, lo que haremos es seleccionar el lote proveedor y reubicarlo de forma que el producto intermedio esté disponible en el momento adecuado y la máquina consumidora no se detenga.

La idea general será adelantar la producción del lote lo máximo posible siempre y cuando, al hacerlo, no se produzca una falta de materia prima en alguno de los lotes que hemos adelantado en iteraciones previas, pues de producirse esta situación estaríamos deshaciendo los pasos que hemos avanzado en anteriores acciones.

Algo importante a tener en cuenta, es que producto del cambio de secuencia en el recurso A, es muy posible que se hayan generado tiempos de cambio nuevos. Con el objetivo de regular el número de estas operaciones, a la hora de adelantar un lote proveedor, se ha de comprobar si en las inmediaciones del lugar donde lo deseamos colocar, existe algún grupo de lotes de su mismo tipo de producto, de forma que si

agregamos el lote a este grupo, evitemos una operación de cambio extra.

El cuarto paso, se denomina **Elevar** las restricciones.

> En este caso, elevar la restricción supondría aumentar la capacidad de producción del recurso A, o disponer de otro recurso capaz de producir la materia prima necesaria. Como esto se trata de una decisión que no atañe al planificador de tareas, este paso se omitirá en el flujo del algoritmo.

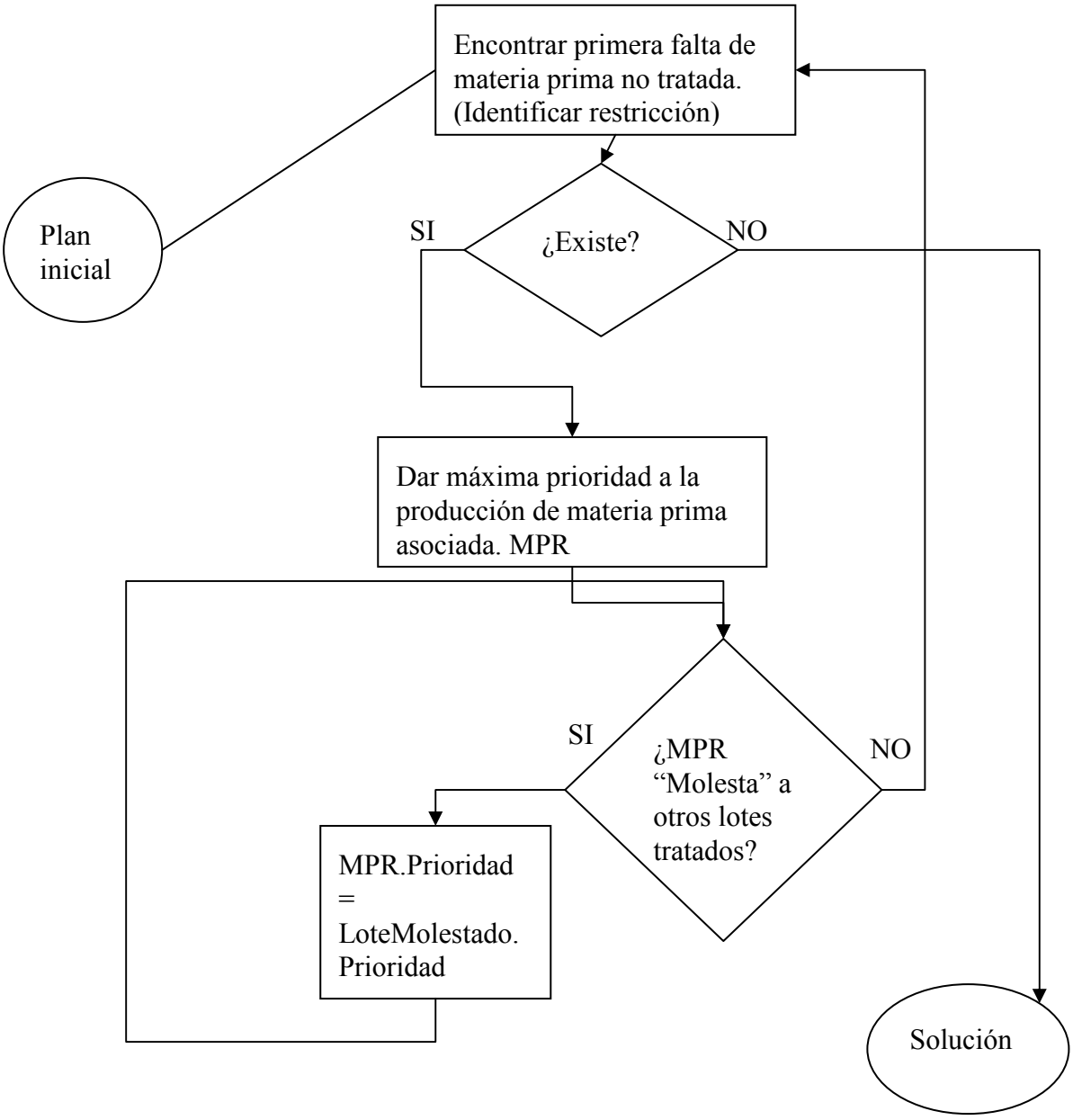
Si se logró eliminar la restricción **regresar al Paso 1.**

> Si fruto de las acciones realizadas (adelantar la producción de materia prima) hemos logrado eliminar el problema identificado en el paso 1, entonces se buscará de nuevo para encontrar el siguiente punto del plan donde se produce una falta de productos intermedios, para ello, volveremos al punto 1. Si el problema no se ha podido solventar, marcaremos ese lote de una forma especial, de forma que, a pesar de ser el factor limitante, lo omitamos en futuras iteraciones para tratar de mejorar la situación del plan a pesar de contar con un lote que no hemos podido adelantar lo suficiente.

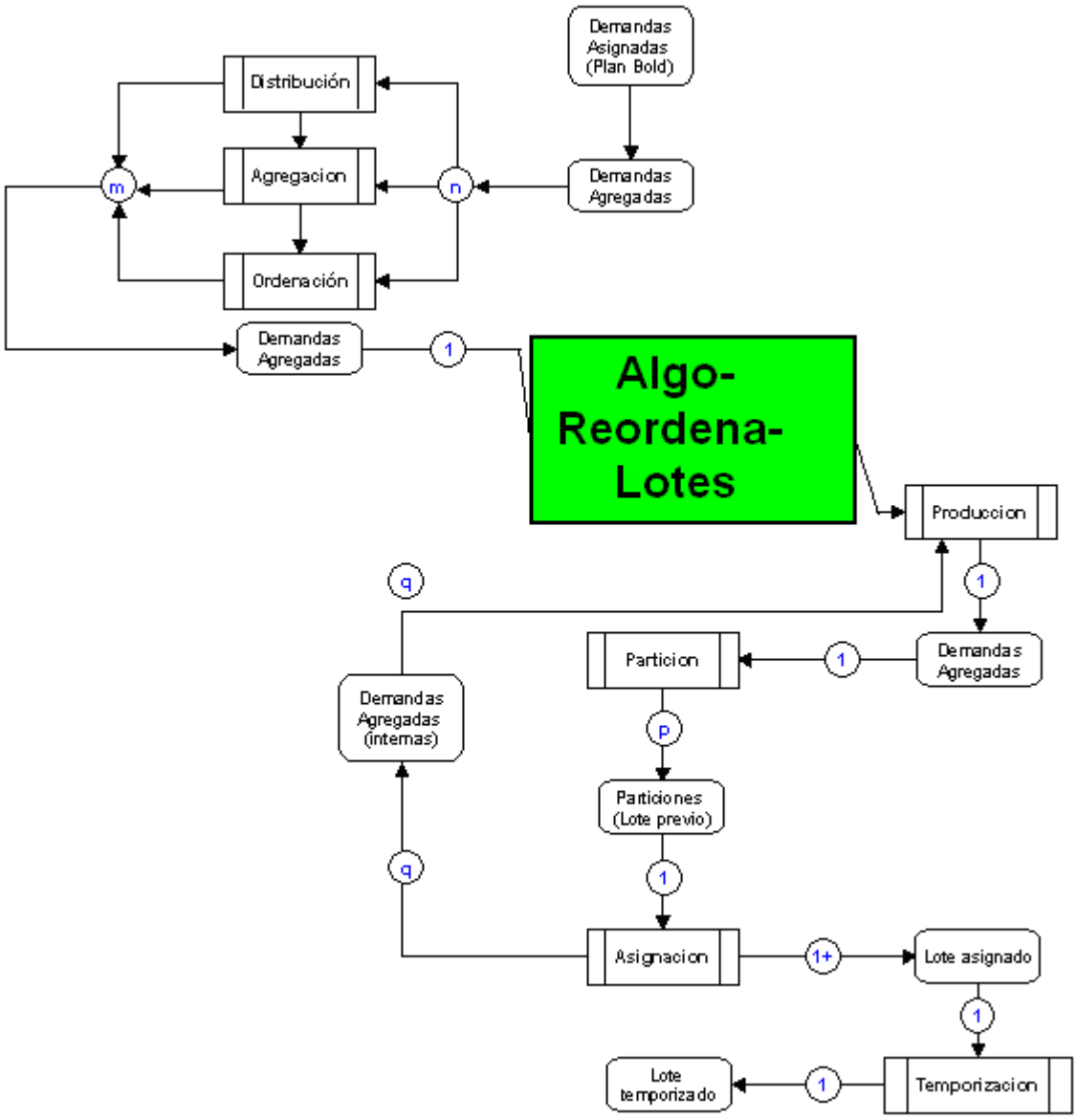
Con cada iteración que llevemos a cabo, mejoraremos la situación, pues cada vez, el problema se encontrará más alejado del inicio del plan, hasta que llegue un momento en que todos los problemas han sido tratados, tanto para marcarlos como “no mejorables siguiendo este procedimiento”, como para solventarlos adelantando la producción de sus lotes proveedores.

Cuando se llegue a un punto en el que no existan lotes retrasados por falta de materia prima, o en caso de existir, se encuentren marcados como insolventables, se detendrán las iteraciones y podemos dar por finalizada la búsqueda de la solución.

Esquema del algoritmo:



El algoritmo comentado anteriormente, complementa al existente en BOLD APS, pasando a formar parte del mismo como un nuevo elemento:



El sub-sistema AlgoReordenaLotes recibe un agregado de demandas lotificadas que son reordenadas en su seno de forma adecuada para conseguir mejoras en las soluciones finales. Su funcionamiento detallado se ha explicado en el presente capítulo.

Estado del plan en diferentes iteraciones:

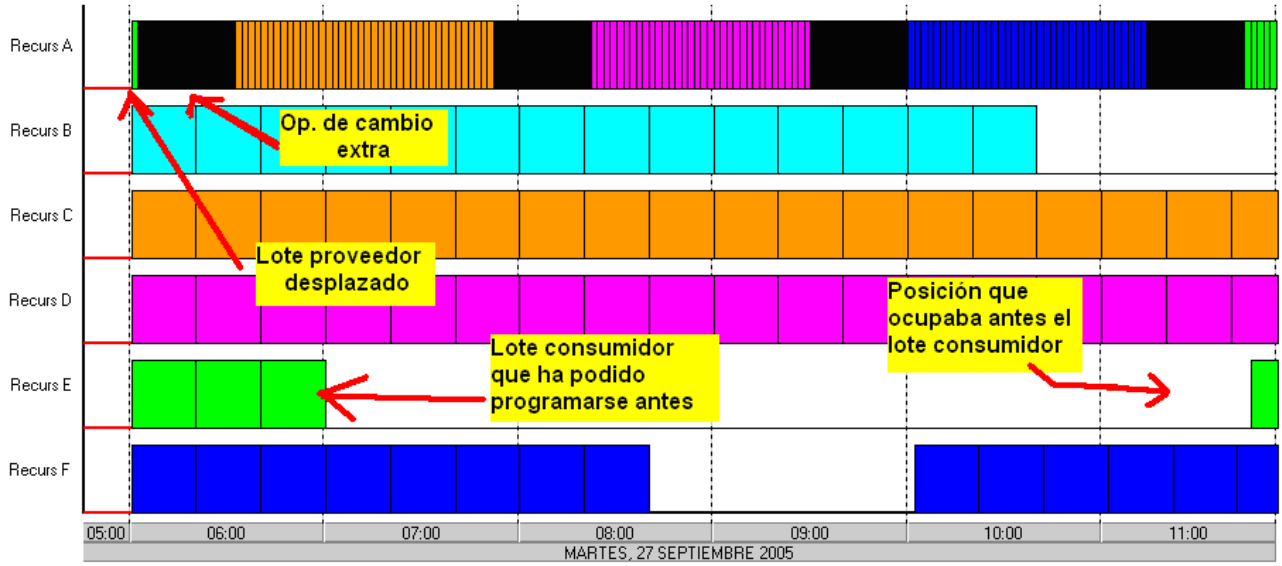
En este apartado, se muestran los diferentes estados por los que pasa el plan antes de llegar a la solución final. Para simplificar, no pondremos el estado tras cada una de las

iteraciones, sino sólo de aquellas que puedan aportar algo interesante, pues el resto pueden ser demasiado obvias.

Con el fin de facilitar la compresión de lo que sucede en cada momento, se han añadido comentarios a las imágenes. Además, de forma general, las ilustraciones no muestran el estado global del plan, sino solo el de aquella parte del mismo sobre el que se están produciendo cambios en cada iteración.

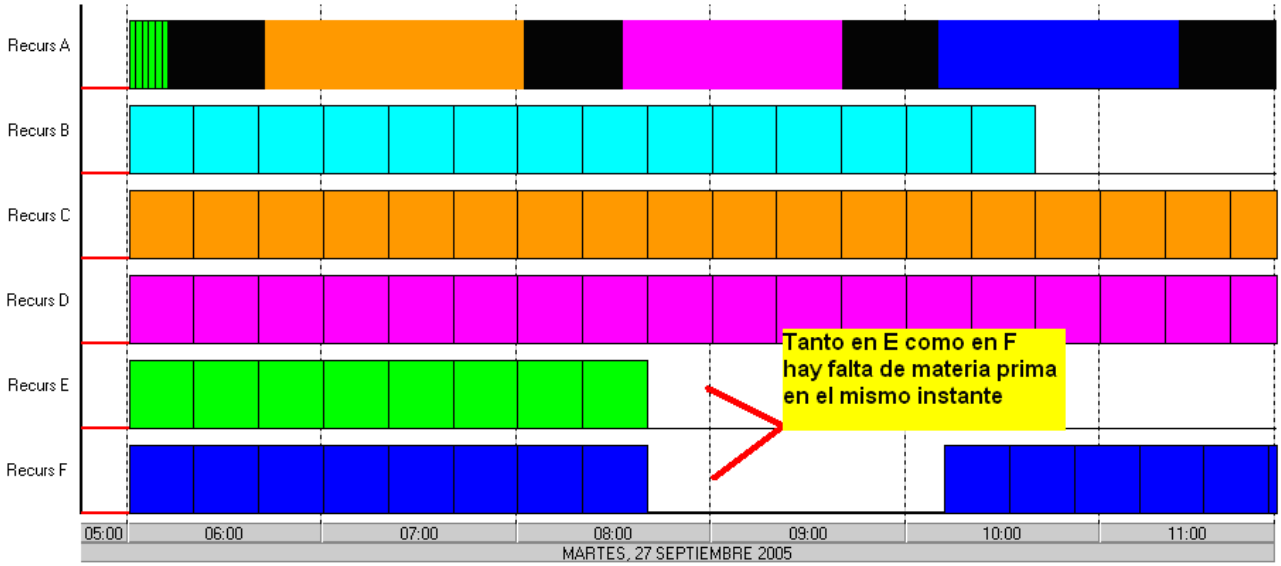
En la primera iteración se traslada un lote proveedor del recurso E, situándolo el primero de la lista. Como consecuencia, se ha podido fabricar un lote adicional de producto E en el turno de mañana.

Iteración 1:



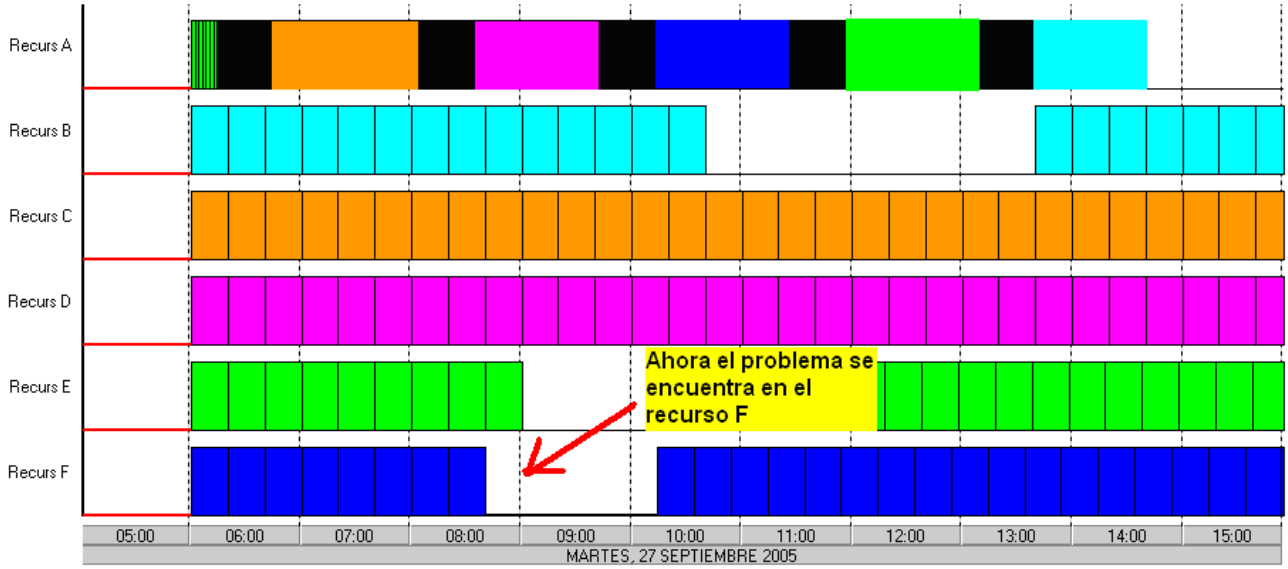
Tras el cambio, se vuelve a buscar cual es el factor limitante en la nueva situación. En este caso, vuelve a ser un lote del recurso E. Aplicando el procedimiento 5 veces más, se llega a la siguiente situación:

Iteración 6:

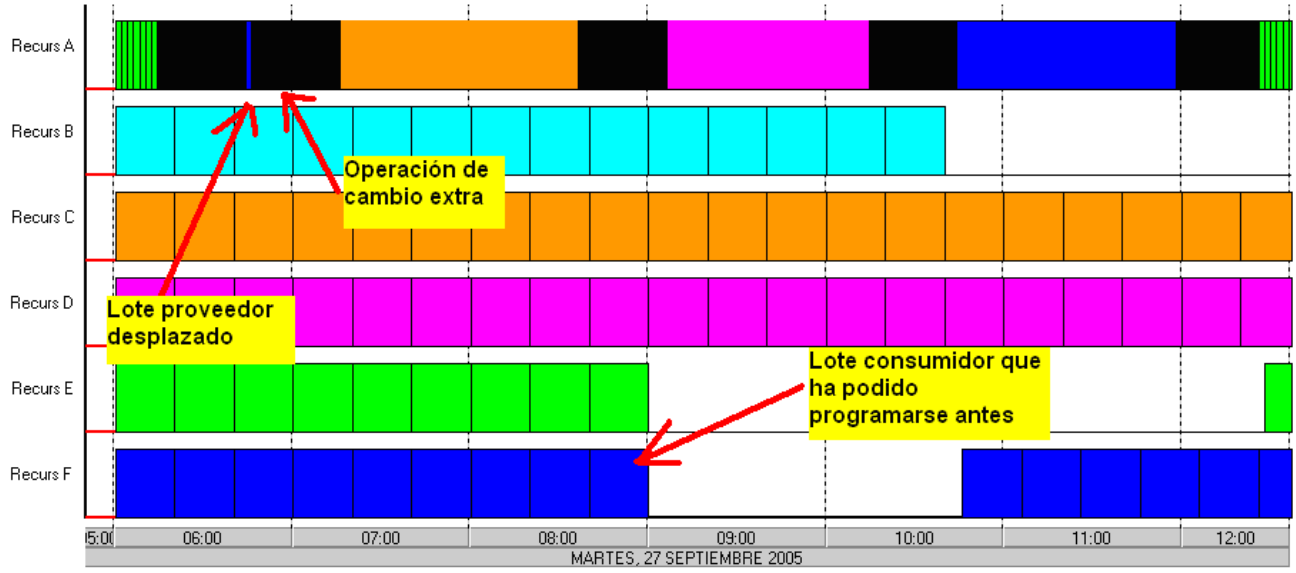


En este punto, nos encontramos que hay dos recursos donde falta materia prima en el mismo instante. Como en este caso no disponemos de ningún elemento diferenciador como prioridad del cliente o producto, trataremos primero el caso del recurso E.

Iteración 7:



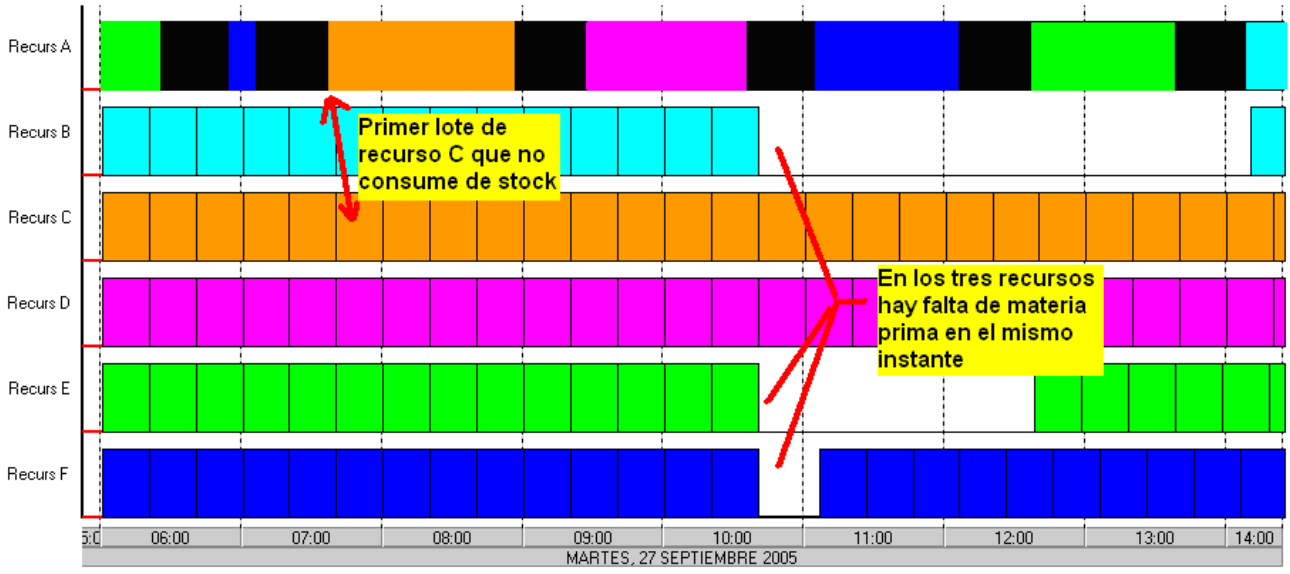
Iteración 8:



Tras efectuar la octava iteración, vemos como se ha generado otra operación de cambio extra que ha permitido programar la producción de un lote proveedor con anterioridad. Como consecuencia, tenemos que el lote consumidor del recurso F ha podido adelantarse lo suficiente como para dejar de ser el recurso limitante.

Iteración 9…18:

Lo sucedido en las anteriores iteraciones se va repitiendo entre las iteraciones 9 y 18. Se adelantan las producciones de más lotes proveedores del recurso E y del recurso F hasta que se llega a lo siguiente:



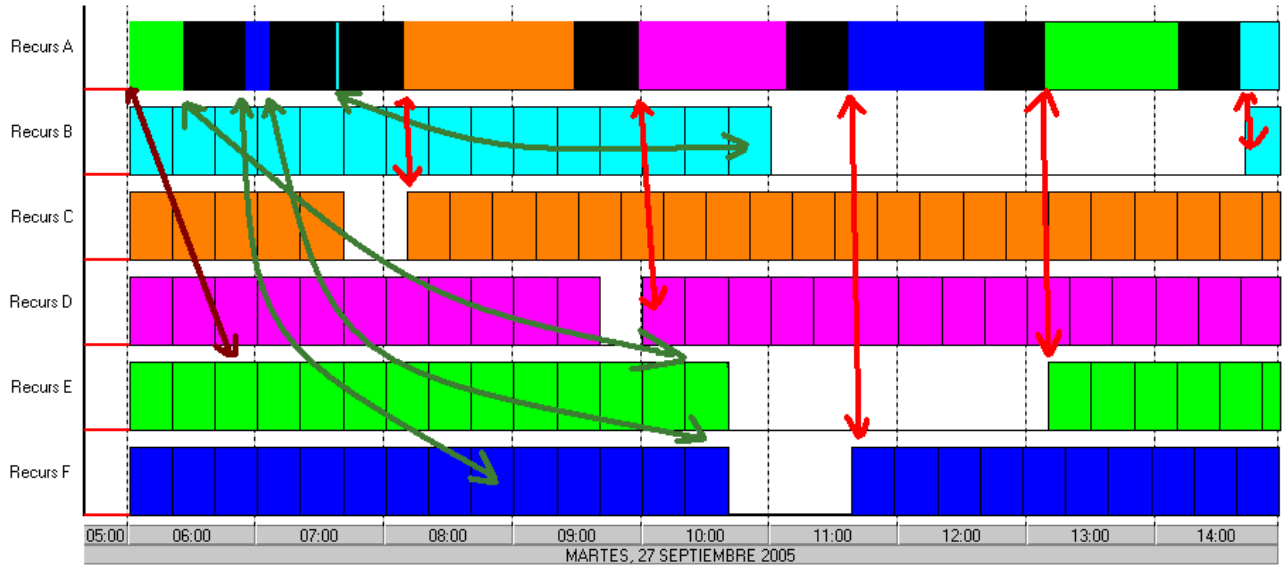
Tal y como sucedió en la iteración 6, se tratará de resolver primero el problema del recurso superior, en este caso el recurso B.

Como según el procedimiento fijado, lo que se hará es adelantar un lote de producto intermedio B tanto como podamos sin “molestar” a los lotes que ya hemos movido y teniendo en cuenta que la producción de producto intermedio C se encuentra muy próxima a su consumo, para solucionar el problema que tenemos en B, generaremos una falta de materia prima en el recurso C en un instante muy anterior al de B que estamos intentando solventar.

Esto parece un empeoramiento de la solución, pero tal y como está diseñado el algoritmo, al generar este “hueco” en C en fecha tan anterior al hueco que tratamos de solventar, es casi seguro que en la próxima iteración, sea este déficit el que se identifique como siguiente limitación.

Iteración 19:

Aunque la solución parezca peor, si mostramos de alguna forma las diferencias existentes entre que los lotes proveedores se fabrican y se consumen (**buffers temporales**), veremos que los cambios que hemos llevado a cabo hasta ahora se encuentran bastante holgados, permitiendo introducir lotes anteriores a ellos sin problema. Esta propiedad es la que nos permitirá solventar los huecos que se produzcan fruto de adelantar otros lotes.



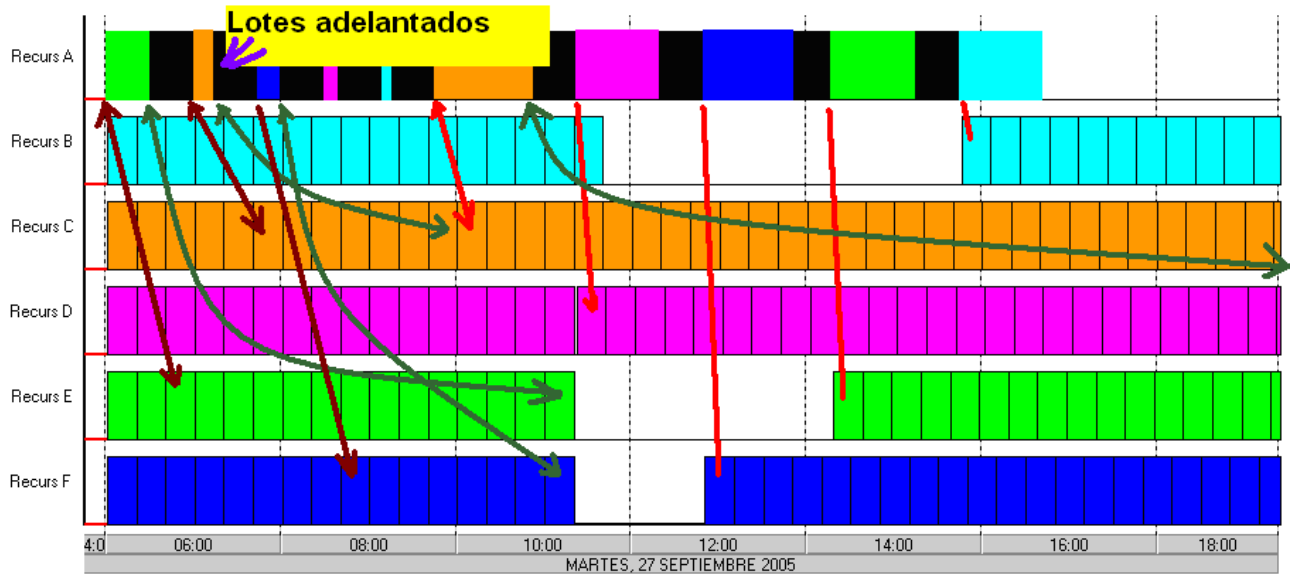
En esta ilustración, las flechas indican la relación entre los lotes consumidores y proveedores. Se han dibujado en color verde las flechas relacionadas con aquellos productos intermedios que se fabrican bastante antes de ser necesarios, y en rojo las que señalan las situaciones de rigidez en las que los productos son consumidos

inmediatamente después de ser producidos. Caso especial es la primera producción de producto intermedio C en el recurso A, pues los primeros lotes se encuentran bastante cercanos temporalmente a sus consumidores, mientras que los últimos del grupo se consumen pasado un tiempo considerable, por ello hemos dibujado una flecha de cada color.

Lo que veremos en las próximas iteraciones, es como el tiempo de antelación con que se fabrican los productos intermedios se va homogeneizando para todos los productos. No es adecuado tener la materia prima de un producto con mucha antelación mientras que la de otra, ni siquiera llega a tiempo.

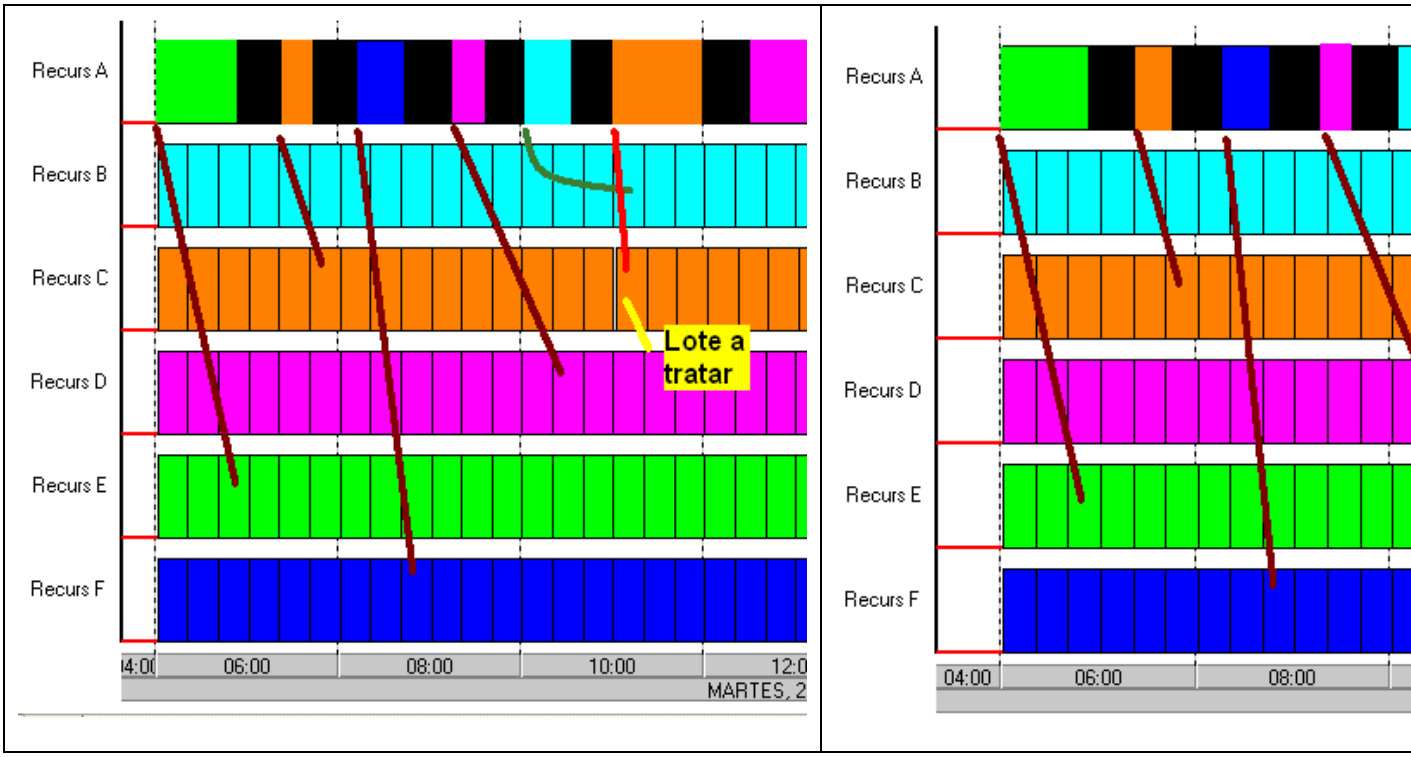
Iteración 20…37:

Como el problema se encontraba en este caso en el recurso C, se emplean 4 iteraciones en adelantar producto intermedio de este tipo. Dados los tamaños de los **buffers** existentes en los lotes que ya hemos adelantado, se nos permitirá situar los lotes proveedores de C en segunda posición. Este proceso, de detectar problemas y adelantar lotes continuará sin mayores sorpresas ajustando las holguras temporales existentes entre los lotes proveedores y consumidores, tendiendo a ser un valor lo más homogéneo posible entre todos los productos.



Iteración 38…75:

En el siguiente gráfico se muestra un pequeño hueco en el recurso C, que es el foco de mejora de la siguiente iteración. La manera de tratarlo será adelantando un lote de producción de producto intermedio C. Para ello se traspasará un lote desde la segunda agrupación hasta la primera. Este cambio será factible pues todos los lotes que se producen entre ambos grupos (productos intermedios F, D y B) tienen un buffer temporal suficiente como para permitir un retraso adicional equivalente al tiempo de producción de un lote de producto intermedio C sin incurrir en retrasos.



A costa de reducir ligeramente los tiempos de holgura con que contaban los productos intermedios F, D y B hemos conseguido servir a tiempo el producto intermedio C.

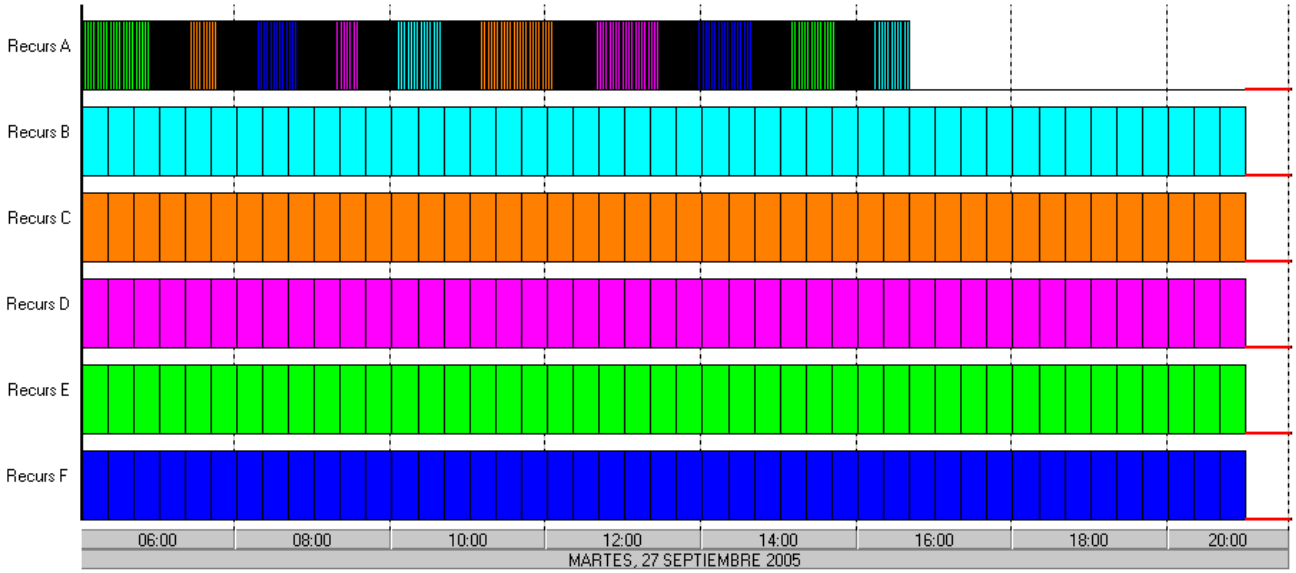
Resultado final:

5 iteraciones más, dejan el plan en un estado en el que no se pueden identificar nuevas limitaciones en el sistema, y por tanto se considera como plan de salida.

A pesar de que la solución teórica del problema proponía la secuencia:

4 - 2 - 5 – 3- 1- *4 – 2 – 5 – 3 – 1 –* **4 – 2 – 5 – 3 - 1**

El algoritmo desarrollado propone una secuencia algo diferente (**4-2-5-3-1**-2-3-5-4-1-, pero que aporta un resultado idéntico en cuanto a fechas de entrega.



La solución propuesta cuenta con la ventaja de disminuir el número de operaciones de cambio con respecto a la solución teórica. Esto se consigue a costa de producir únicamente las cantidades necesarias de productos intermedios, sin preocuparnos de mantener ningún tipo de margen entre la materia prima necesaria y la que producimos. El actuar de esta manera, como contrapartida, hace que el recurso A se encuentre libre desde las 16:45 hasta las 21:00, utilizando casi un turno menos de trabajo. Este tiempo puede ser aprovechable para dejar los stocks de materia prima en márgenes de seguridad.

Plan de pruebas:

Test0

	Demandas	Algoritmo modificado	
Pieza	**Cantidad**	**Fecha final entrega**	**Diferencia***
A	450	27/09/2005 21:00	11:40
B	450	27/09/2005 21:00	0:00
C	450	27/09/2005 21:00	0:00
D	450	27/09/2005 21:00	13:40
E	450	27/09/2005 21:00	10:00

Test1

	Demandas	Algoritmo modificado	
Pieza	**Cantidad**	**Fecha final entrega**	**Diferencia***
A	300	27/09/2005 16:00	0:00
B	450	27/09/2005 21:00	13:00
C	390	27/09/2005 19:00	0:00
D	450	27/09/2005 21:00	12:00
E	275	27/09/2005 15:20	0:00

Test2

	Demandas	Algoritmo modificado	
Pieza	**Cantidad**	**Fecha final entrega**	**Diferencia***
A	450	27/09/2005 21:00	0:00
B	450	28/09/2005 07:00	-10:00
C	450	27/09/2005 21:00	0:00
D	900	28/09/2005 21:00	16:40
E	900	28/09/2005 21:00	11:40

La columna “Diferencia” hace referencia a la resta entre la fecha final de entrega obtenida mediante el algoritmo original y la obtenida mediante el algoritmo modificado.

Al enfrentar al algoritmo modificado al plan de pruebas propuesto, vemos como el utilizar esta forma de proceder lleva a planes donde las demandas se entregan con bastante antelación en la mayoría de los casos, aún modificando las cantidades demandadas. Esto nos muestra, en parte la capacidad de la que dispone BOLD APS y su algoritmo de planificación para amoldarse a diferentes situaciones sin necesidad de efectuar cambios drásticos en la manera de funcionar.

Al no tratar de forma general de maximizar o minimizar el numero de operaciones no productivas (tiempos de cambio), podemos adaptarnos a cada situación pues el propio flujo de código será el encargado de encontrar una medida adecuada para los mismos, pues a través de sus iteraciones se irán creando nuevos tiempos de cambio cuando sea necesario y eliminándolos cuando sea posible. Así, nuestro objetivo principal deja de ser el gestionar los tiempos de cambios y pasa a ser el entregar los productos cuando son necesarios, dejando que este tipo de operaciones aparezcan de forma natural como producto de las necesarias transiciones entre tipos de productos.

Capítulo 7: Conclusiones y posibles ampliaciones

Los objetivos del proyecto, han sido cumplidos en su totalidad.

Este proyecto se encuentra dividido en dos fases. Los resultados de la primera se comentan en el apartado “Fase 1” del capítulo 6. Éstos son positivos pues se ha logrado aumentar la accesibilidad del código fuente del algoritmo para programadores ajenos al grupo de trabajo que lo diseñó. Además tal y como se comenta en el capítulo correspondiente, se han podido eliminar fragmentos de código que por no utilizarse, solo causaban confusión. Las métricas generales del subsistema del algoritmo de BOLD APS han sido mejoradas tras la conclusión de este proyecto.

La fase 2 y principal del presente proyecto se ha finalizado con excelentes resultados, pues las soluciones ofrecidas por el algoritmo modificado, no solo son de mejor calidad que las del algoritmo original, sino que son comparables en calidad a las soluciones teóricas del problema planteado.

Además, el tiempo de calculo extra que se desprende de la fase que hemos agregado (AlgoReordenaLotes) es bastante menor del que nos habíamos marcado como umbral. Siendo dicho límite del orden de los pocos minutos (2-4 minutos), se consigue la solución final en un tiempo cercano a los 15 segundos.

También se ha de destacar como positiva la similitud conseguida entre el flujo general del algoritmo y las reglas definidas en la teoría de las limitaciones.

Como en cualquier otro campo, en el de la planificación de la producción siempre quedarán múltiples líneas de desarrollo por explorar. Especialmente interesantes consideramos que son:

- La posibilidad de incorporar nociones de **algoritmos genéticos** pues durante el desarrollo de la nueva fase del algoritmo se hizo un intento que no dio muchos frutos y creemos que el potencial existente en esta rama es muy grande. A pesar de que el espacio de soluciones sea enorme, parece que la naturaleza del problema, y el hecho de que las diferentes soluciones puedan compartir fácilmente información común (tanto positiva como negativa) hacen de los algoritmos genéticos un pozo donde investigar bastante

profundo. Aunque es necesaria una aportación temporal mucho mayor que la disponible para la realización de este proyecto, puede ser una inversión muy interesante.

- Desarrollar un motor de **inteligencia artificial** que sea capaz de aprender de las modificaciones manuales realizadas por el usuario y que mejoran las soluciones inicialmente presentadas por el algoritmo.

  Muchas veces ocurre que las soluciones propuestas por un sistema de planificación han de ser alteradas manualmente con el fin de mejorarlas, pues normalmente existen situaciones que los procesos automáticos no tienen en cuenta, por ser demasiado específicos de un caso concreto o simplemente porque no se había previsto según que casuística. El hecho de que el proceso de planificación no sea capaz de aprender de las modificaciones manuales llevadas a cabo por el usuario hace que éste, se vea obligado a repetir las mismas correcciones cada vez que necesita planificar la producción de su fábrica.

  Un camino interesantísimo a recorrer sería otorgar la capacidad de aprendizaje al motor del algoritmo de forma que pueda aplicar algunas reglas inferidas de la forma de actuar del usuario.

Desviaciones

A pesar de que el desarrollo del proyecto se ha ajustado bastante a la planificación, tareas como las de documentación e implementación han sufrido un aumento en el número de horas que se les ha dedicado.

Las posibilidades que se abrían al comienzo de este proyecto eran tantas que el simple hecho de conocer los rudimentos de algunas de ellas ha supuesto un trabajo de aprendizaje bastante grande. Además, éste se trata de un tema lo suficientemente profundo y estudiado como para existir tanto material de estudio como se desee encontrar. Ha sido necesario efectuar una lectura superficial de diferentes puntos de vista, así como de información básica acerca de conceptos imprescindibles sobre los que se

edifica todo lo demás. Se ha profundizado la documentación en dos temas principalmente:

El primero, la teoría de las limitaciones, bastante comentada a lo largo de este proyecto, pues se ha utilizado tanto en la definición del problema como en la forma de conseguir su solución. Documentos extraídos del Instituto Goldratt y bibliografía de E. Goldratt han sido las principales fuentes sobre este tema.

El segundo, el método de optimización denominado swarm optimization (optimización por enjambre de partículas). Hasta conocer las verdaderas dimensiones de un proyecto donde se aplique esta técnica para solucionar el problema de la planificación de la producción se invirtió un gran tiempo de documentación y pruebas. Realmente esta opción de desarrollo, resulta especialmente interesante, pero la dedicación necesaria resulta excesiva para abordarla, al menos de momento. Este sistema pude ser prometedor, pero requiere de un tiempo no disponible en este proyecto. El descartar este camino aunque ha resultado ser enriquecedor, ha conllevado una cantidad de días considerable.

A pesar de que el tiempo previsto para la fase de documentación era inicialmente de 52 horas, situadas como primera etapa del proyecto, este número se ha visto ampliado en 20 horas. Además, el conjunto total de estas horas, ha sido distribuido durante la primera mitad del desarrollo del proyecto, concluyendo casi cuando se decide abordar el problema utilizando un algoritmo especializado.

Por otra parte los procesos de test se han llevado a término con menores problemas de los previstos, lo que ha supuesto una reducción en el tiempo dedicado a ellos. Aunque el someter al algoritmo a los diferentes conjuntos de pruebas ha revelado en ocasiones errores de codificación, en la mayoría de los casos todo ha funcionado de forma satisfactoria. Además el generar diferentes escenarios de pruebas que ofrecer al algoritmo ha sido bastante sencillo de conseguir, pues el software sobre el que se trabaja se encuentra orientado a admitir variaciones en cuanto a la demanda con facilidad.

Donde se ha producido un cambio significativo en cuanto a la planificación ha sido en la distribución de horas dedicadas a “implementación”, “pruebas y modificaciones”, y “análisis de modificaciones”. Pues en lugar de encontrarse tal y como figuran en la

planificación, se han intercalado periodos de implementación con periodos de análisis y pruebas. Éstos últimos, dan lugar a nuevas modificaciones, implementación y nuevos análisis y modificaciones… Produciéndose varios ciclos de alternancia en lugar de uno solo como se tenía previsto.

A pesar de todas estas desviaciones y siguiendo el principio de albañilería del uno con otro, unos desajustes se han compensado con otros, y el total de horas consumidas ha sido aproximadamente el previsto inicialmente.

Capítulo 8: Bibliografía

La meta: un proceso de mejora continua:(con la entrevista realizada por David Whitford, ...) / Eliyahu M. Goldratt, Jeff Cox ; [traducido y revisado por Enrique Rey Arufe y Mª Consuelo Núñez Fernández]. B 25337-2001

El síndrome del pajar. Goldratt, Eliyahu M. B 45866-2006

Programación Matemática I Felipe, P. y otros. Ciudad de la Habana, 1983.

Álgebra y matemática discreta Aledo Sánchez, Juan Ángel. AB 63-2000

Las hormigas Werber, Bernard. B 40666-1997

Documentación APS (diversos documentos acerca del uso de APS y del funcionamiento del algoritmo)

Particle Swarm Optimization James Kennedy, Russell Eberhart . (http://www.engr.iupui.edu/~shi/Coference/psopap4.html)