UNIVERSIDAD DE SEVILLA

ESCUELA SUPERIOR DE INGENIEROS

INGENIERÍA DE TELECOMUNICACIÓN

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

ÁREA DE TEORÍA DE LA SEÑAL Y COMUNICACIONES

PROYECTO FIN DE CARRERA

DESARROLLO DE UNA HERRAMIENTA DE AYUDA A LA PLANIFICACIÓN. GENERACIÓN DE LA ESCALERA DE AVIONES

AUTOR: DAVID GALVÁN MUÑOZ

TUTOR : RAFAEL BOLOIX TORTOSA

FEBRERO 2003

Proyecto fin de carrera: **DESARROLLO DE UNA HERRAMIENTA DE AYUDA A LA PLANIFICACIÓN. GENERACIÓN DE LA ESCALERA DE AVIONES**

Autor: David Galván Muñoz

Tutor: Rafael Boloix Tortosa

El tribunal nombrado para juzgar el proyecto fin de carrera arriba citado, compuesto por:

Presidente:

Vocal:

Vocal secretario:

Acuerda otorgarle la calificación de:

Sevilla, a de de 2003

AGRADECIMIENTOS

Quiero dar las gracias a todas las personas que han compartido conmigo los “duros” años de carrera y me han apoyado siempre, especialmente a mi familia y a los buenos amigos que he hecho durante estos años.

En segundo lugar, me gustaría agradecer la colaboración de quienes trabajaron conmigo en EADS CASA, especialmente a Ramón Manresa, Santiago López y Teresa Talavera, ya que sus aportaciones fueron decisivas en este proyecto. Tampoco quiero olvidarme de las aportaciones de amigos, que a pesar de mi insistencia, me han ayudado siempre con gran entusiasmo a elaborar esta memoria, especialmente de Noelia.

Por último, quiero mostrar mi agradecimiento a Rafael Boloix Tortosa, por haberse ofrecido a ser mi tutor durante el desarrollo de este proyecto y el apoyo mostrado en la “larga” realización del mismo.

ÍNDICE

ÍNDICE

1.INTRODUCCIÓN

1.1. OBJETIVO DEL PROYECTO

En la actualidad la búsqueda de técnicas y herramientas que nos permitan optimizar el trabajo es una cuestión fundamental para las empresas. En ingeniería esta necesidad se acentúa debido a la importancia del tiempo en los costes asociados al desarrollo y al diseño.

Con este proyecto se intenta, mediante el desarrollo de una herramienta software, facilitar la creación de lo que se conoce en aeronáutica como la escalera de aviones. Esta escalera no es más que un diagrama de Gantt en el que se puede visualizar el comienzo y fin de los distintos aviones a fabricar en un periodo de tiempo concreto. La principal utilidad de la aplicación desarrollada es la de automatizar un proceso antes desarrollado de forma manual. Se engloba dentro del ámbito de la planificación y su principal objetivo es facilitar una tarea antes ardua y repetitiva, aunque esta no sea la única ventaja conseguida, como se demostrará a lo largo de esta memoria.

Este proyecto surge de una necesidad real en un entorno laboral concreto, por lo que el enfoque es eminentemente práctico. Esto no es óbice para tratar cuestiones que a mi entender son importantes en el ámbito profesional y que están relacionados con la génesis de este proyecto, como la necesidad de dotar a las empresas de soluciones globales informatizadas y las crecientes necesidades de infraestructuras informáticas y de telecomunicaciones como soporte a las políticas de ingeniería concurrente. Aún siendo su problemática muy concreta, los resultados, conclusiones y desarrollo de este proyecto son muy generalizables y útiles en diversos campos.

La utilidad práctica del proyecto nos permite abordar un tema cada vez más importante en las empresas, la necesidad de disponer de herramientas software que aun siendo en su origen genéricas estén adaptadas a sus necesidades para una mejor eficiencia en tiempo y calidad en el trabajo.

Por último, aclarar que no es el objetivo de este proyecto el estudio de la fase final del ensamblaje de un avión, por lo que no se entrará en detalle en esta materia. Por motivos de seguridad industrial y militar se va a exponer una información muy limitada de las características del avión y de su ensamblaje, siendo siempre información muy genérica, en ningún caso comprometida y buscando siempre hacer la explicación lo más real posible pero tomando datos no reales en la operación de ensamblaje.

1.2. ESTRUCTURA DE LA MEMORIA

La siguiente memoria se puede dividir, conceptualmente, en dos bloques. El primero está formado por los cinco primeros capítulos, el segundo esta formado por el resto.

El objetivo del primer bloque es ubicar el proyecto (*capítulo II*), y realizar un estudio teórico del entorno laboral en el que se desarrolla (*capítulos III y IV*). También se estudiarán aquellos aspectos teóricos de planificación y aeronáuticos que son necesarios para una comprensión profunda de la aplicación (*capítulo V*).

En el segundo bloque de capítulos se presentará y se profundizará en la aplicación desarrollada, para ello se analizará cómo se realizaba la planificación antes de la utilización de esta herramienta automática (*capítulos VI*). También se explicará el funcionamiento de la aplicación y las fases de desarrollo (*capítulos VII y VIII*). Por último, se sacarán conclusiones de las ventajan obtenidas con el uso de la aplicación desarrollada y se analizarán posibles líneas de avance (*capítulo IX*).

2.ORIGEN DE LA PROBLEMÁTICA

2.1. INTRODUCCIÓN

Este proyecto es un proyecto "real", cuya función principal no es teórica sino práctica, es pues importante para la comprensión del mismo ubicar el entorno donde se originó la necesidad y al cual está destinada su utilidad.

2.2 EADS

La empresa para la que se hizo la aplicación es EADS (*European Aeronautic Defence and Space Company*). Las actividades de EADS se extienden a los segmentos de aeronáutica civil y militar, navegación espacial, sistemas de defensa y servicios. La empresa se creó por fusión de la compañía alemana *DaimlerChrysler Aerospace AG*, la francesa *Aerospatiale Matra* y la española CASA el 10 de julio de 2000. La EADS ocupa la primera posición entre las compañías del sector aerospacial de Europa y la segunda de las del mundo.

EADS está organizada en cinco Divisiones:

- Airbus
- División de Transporte Militar (MTA)
- Aeronáutica
- Espacio
- Sistemas Civiles y de Defensa

Dentro de EADS el proyecto se ha realizado para EADS CASA. Construcciones Aeronáuticas, S.A. (CASA), es la primera compañía del sector aeronáutico español. Como resultado de su unión en 1999 a EADS, pasa a ser EADS CASA y se estructura en las cuatro divisiones en que desarrolla su actividad industrial: Aviones de Transporte Militar, Airbus, Aeronáutica y Espacio. A su vez las dos factorías ubicadas en Sevilla (San Pablo y Tablada) pertenecen a la MTAD (*Military Tansport Aircraft División*), es decir, a la División de aviones de transporte militar.

Para EADS uno de los proyectos más importantes es el nuevo avión de transporte militar, A400M (*Airbus 400 Military*). Airbus se dedica exclusivamente a la aviación civil, por lo que para este nuevo avión se creó AMC (*Airbus Military Company*), que a su vez subcontrata a MTAD, por su experiencia en la fabricación de aviones militares de transporte.

Este proyecto es un claro ejemplo de la nueva tendencia de la ingeniería: la ingeniería concurrente. Son muchos los países que intervienen en la fabricación del A400M dividiéndose la carga de trabajo, ocupándose de ellas las respectivas compañías aeronáuticas de los países. Todos los países son europeos puesto que este avión pretende ser la referencia de la aviación de transporte militar sustituyendo al Hercules C-130 americano, al igual que ocurre en la aviación civil con el Airbus 380, y así acabar con la dependencia en el área militar de EEUU. Esta política europea se completa con el proyecto Galileo, el nuevo sistema GPS europeo, el helicóptero Tigre y el nuevo avión de combate Eurofighter. De entre estos países cabe destacar Alemania, Francia, Reino Unido, Turquía, Italia (todavía no esta definida su inclusión), Portugal y España. Los ministerios de defensa de los países anteriormente citados son los principales clientes de este proyecto.

Llegado a este punto, conviene aclarar que este tipo de proyectos no son realizables a menos que sean rentables de antemano, es decir, se debe empezar con un volumen mínimo de pedidos. En este caso, el volumen afecta al precio de cada avión y este a su vez a la rentabilidad del proyecto. Este "umbral" se ha fijado (realmente ha variado en estos dos últimos años) en torno los 200 aparatos.

En Sevilla se realizará lo que se conoce en aeronáutica como FAL (*Final Assembly Line*), es decir, la línea de montaje final. Las distintas partes del avión llegarán a Sevilla para su posterior ensamblaje y entrega al cliente. También se harán en Sevilla los cursos de aprendizaje para pilotos.

2.3. PROGRAMA DEL A400M

Una vez explicada básicamente la estructura empresarial de EADS es importante entender cómo se organiza el proyecto del A400M. Al ser un proyecto en el que intervienen muchos países existe una jefatura central con misiones de control y gestión que se conoce como oficina de gestión del programa del A400M o PMO (*Program Management Office*). Actualmente está en Madrid por ser la MTAD (EADS CASA) la empresa especializada en aviación de transporte militar pero todavía no se ha fijado la sede definitiva de la dirección al existir la posibilidad de un traslado a Toulouse, sede central de Airbus. A partir de ahora a la dirección del proyecto del A400M la denominaremos como Programa del A400M. Programa tiene presencia en todas las factorías dónde se realizan tareas vinculadas con la fabricación y ensamblaje del avión.

Antes de seguir conviene aclarar las diferencias existentes entre programa y proyecto. Los términos gestión de proyectos (*Project Management*) y gestión de programas (*Program Management*) son habitualmente tratados como sinónimos, en otros casos, la gestión de proyectos es una subdivisión de la gestión de programas, por lo que un programa está compuesto de varios proyectos. También puede considerarse al revés. Por lo tanto, llegamos a la conclusión de que al referirnos a programa y proyecto es conveniente un consenso en la definición de cada término. En aeronáutica se habla del Programa del avión XXX e incluye a los proyectos de diseño y desarrollo del avión y también a los de manufacturación y soporte al cliente. Es por ello que utilizamos el término de programa, aunque no sea incorrecto hablar de proyecto.

En Sevilla, en la factoría de San Pablo, para el estudio del proceso de ensamblaje existe un departamento de ingeniería, Ingeniería de Desarrollo del A400M, encargado del estudio del proceso industrial propio del ensamblaje y también está la PMO que

controla el ensamblaje del A400M, la PMO realiza funciones de supervisión, validación y gestión económicas sobre el trabajo desarrollado por el departamento de ingeniería.

2.4. A400M

En la figura 2.1 vemos una imagen realizada por ordenador del futuro avión de transporte militar A400M. Esta es una imagen que muestra la polivalencia de este avión diseñado para el transporte de todo tipo de cargas, ya sea para misiones humanitarias como las propias tareas militares.

Figura 2.1. Avión de transporte militar A400M

Es interesante conocer datos relevantes sobre el avión que da origen a la necesidad de este proyecto. Los datos que se exponen a continuación son totalmente públicos.

El A400M es un cuatrimotor turbohélice que puede cumplir una amplia gama de misiones: transportar una carga útil máxima de 37 toneladas (120 soldados / paracaidistas, nueve palés de 108" x 88" (dos en rampa), 66 camillas y 10 asistentes médicos) o 25 toneladas de carga útil a una distancia de 5.500 Km. Todo ello a una velocidad de crucero de 0,72 Mach. La cabina de pilotos, similar a la del Airbus, con pantallas LCD, incorpora lo más avanzado en aviónica integrada, como HUD, SATCOM, etcétera.

El A400M se ha diseñado para satisfacer las necesidades de siete fuerzas aéreas: un transporte militar moderno, rápido, práctico y económico. Nueve países contratantes han pedido 229 aparatos, el lanzamiento del programa fue en el 2001.

Tras la fabricación de 6 aviones prototipo, en el año 2011 se comenzará en San Pablo (Sevilla) la fabricación en serie a un ritmo de 3 aviones al mes.

Podemos dividir las partes que componen un avión en estructurales y en los distintos sistemas o grupos.

- Dentro de las estructuras del avión tenemos:

Fuselaje: es la parte principal o cuerpo de una avión. Es la parte donde se aprovisiona el espacio para la carga de pago, controles, accesorios y otros equipos.

Las alas: son las partes diseñadas para proporcionar sustentación

Estabilizador horizontal: contribuye en gran medida a la estabilidad longitudinal del avión. Se trata de una superficie aerodinámica simétrica, ya que debe tener posibilidad de generar cargas verticales tanto hacia arriba como hacia abajo.

Estabilizador vertical: supone un importante factor de estabilización en la estabilidad direccional, al igual que el estabilizador horizontal está constituido por un perfil simétrico para que pueda desarrollar fuerzas en ambos sentidos, derecha e izquierda.

El A400M es un avión de ala alta porque sus estabilizadores verticales y horizontales están en un plano horizontal más elevado que las alas como es típico de un avión de carga.

Superficies de control de vuelo: como superficies de mando primario tenemos alerones, timones de profundidad y dirección. Son similares en construcción y varían sólo en tamaño, forma y métodos de unión a las superficies fijas. En su construcción son similares a las alas.

Tren de aterrizaje: el tren de aterrizaje es el conjunto que soporta el avión durante el aterrizaje o mientras el avión opera en está parado en tierra. Es un elemento diseñado con gran robustez.

- Dentro de los sistemas del avión tenemos:

El grupo motopropulsor: es un avión turbohélice. Un motor turbohélice es básicamente una turbina de gas diseñada para mover un eje al que está unida la hélice mediante un reductor de velocidad.

El sistema eléctrico: el sistema eléctrico está formado por las unidades y componentes eléctricos que generan, controlan y distribuyen la energía a todos los elementos y sistemas que la necesitan. En los aviones se utiliza corriente alterna y corriente continua. La corriente alterna constituye la fuente primaria, en tanto que la corriente continua constituye la fuente secundaria.

Los aviones, para atender a sus demandas de energía eléctrica, utilizan básicamente los siguientes tipos de corriente:

1. Corriente alterna trifásica de 115 voltios a 400 cps.
2. Corriente alterna de 28 voltios a 400 cps.
3. Corriente continua de 28 voltios.

El sistema hidráulico: el sistema hidráulico es un sistema auxiliar que sirve para accionar los distintos elementos que son operados hidráulicamente en el avión. Tiene dos partes, el primario que acciona, el timón de dirección, y el secundario que acciona el circuito de retracción y extensión del tren de aterrizaje.

El sistema neumático: básicamente, el sistema neumático funciona con los mismos principios de trabajo que el sistema hidráulico y se utiliza para lograr fines similares a los previstos con éste, pudiendo operar, por ejemplo, frenos y dirección, compuertas, ciertos elementos en situación de emergencia, mover bombas hidráulicas, alternadores, puestas en marcha, bombas de inyección, sistema antihielo, presurización y aire acondicionado del avión.

El sistema de presurización: se encarga de mantener el aire en la cabina a una presión superior a la exterior, sin olvidar que al aterrizar y al despegar ambas deben ser iguales. El régimen de cambio de la altitud debe ser suficientemente confortable y lento como para asegurar que los pasajeros no sufrirán consecuencias de rápidos cambios de presión.

El sistema de aire acondicionado: este sistema está íntimamente relacionado con los de presurización, ya que el aire acondicionado que entra en el avión para presurizar deberá primero sufrir el proceso de acondicionamiento antes de ser introducido en la cabina.

El sistema de oxígeno: el sistema de oxígeno es un sistema previsto para ser utilizado en situaciones de emergencia y normalmente no se utiliza para proporcionar oxígeno a la cabina, ya que esta se mantendrá en los niveles requeridos por el sistema de presurización.

El sistema antihielo: el sistema neumático desprende el hielo formado en los bordes de ataque de las alas, estabilizadores y hélices mediante el inflado alternativo de unas gomas (botas antihielo) colocadas en los borde de ataque.

El Sistema de combustible: el sistema de combustible tiene como función el suministrar combustible líquido y limpio de vapores al motor a las presiones adecuadas y en la cantidad requerida para que sea quemado en las cámaras de combustión.

El sistema contra incendios: el sistema contra incendios está compuesto de un sistema de detección, un sistema de aviso de fuego y un sistema de extinción

Sistemas de instrumentos del avión: básicamente esta formado por todos los instrumentos de medición:

- **Instrumentos de vuelo:** anemómetro, altímetro, bastón y bola, y coordinador de virajes, variómetro, horizonte artificial, indicador de ángulo de ataque.

- Instrumentos de motor: tacómetros, manómetro de admisión, indicadores de presión de combustible y aceite, indicadores de cantidad de combustible, indicadores de temperatura, fluxómetros de combustible.

- Instrumentos de navegación: brújula magnética, el girodireccional, sistema de piloto automático, sistema director de vuelo.

- Sistemas de comunicación y navegación: los sistemas de navegación más utilizados son:

- ✓ Sistema VOR (radiofaro omnidireccional en VHF)
- ✓ Radiocompás o sistema ADF (buscador automático de dirección)
- ✓ Equipo DME o equipo medidor de distancias de ultra alta frecuencia
- ✓ Transpondedor
- ✓ Altímetro codificador
- ✓ Transmisor localizador de emergencia (ELT)
- ✓ Sistemas de aterrizaje por instrumentos (ILS)

2.5. FAL

FAL es el acrónimo de *Final Assembly Line* (línea de montaje final). Como ya se ha comentado la fase final del programa del A400M se hará en Sevilla. Esta fase final constará de la FAL y entrega al cliente de los aparatos así como la preparación técnica de pilotos y mecánicos. La FAL no es más que una línea de montaje en el que se realizarán las operaciones necesarias para el ensamblaje del avión.

Por cuestiones de seguridad industrial no entraremos en detalle en el estado actual de en el que se encuentra el estudio de la FAL pero es interesante mencionar las operaciones básicas de las que consta.

Podemos definir el proceso de ensamblaje como una sucesión de operaciones, por lo que su grado de definición está ligado al de las operaciones que lo componen, estas se pueden agrupar en función de su nivel de detalle, siendo las operaciones más

importantes y generales las que nos permiten tener una visión global del proceso, a estas "macro operaciones" se las conoce como estaciones. Usando esta terminología podemos referirnos al ensamblaje como una sucesión temporal de estaciones. Es frecuente dividir las estaciones en operaciones básicas de gran importancia ligadas a la misma, a estas operaciones se las denominas subestaciones. En este caso, la división no implica una sucesión temporal, ya que varias subestaciones pueden realizarse simultáneamente.

Ejemplos de estaciones básicas en una Fal son: [1]

1. Unión de fuselaje.
2. Integración del avión.
3. Equipado.
4. Pruebas funcionales.
5. Instalación de motores.
6. Decorar, amueblar.[2]
7. Pintura.
8. Línea de vuelo.
9. Entrega y reparto.

Unión de fuselaje: en esta estación tenemos tres subestaciones en paralelo:

1. **Unión de fuselaje central y cabina**
2. **Unión de estabilizadores vertical y horizontal**
3. **Unión del ala**

De estas tres macro operaciones la más delicada es la de unión del ala teniéndose que utilizar técnicas de best-fit (mejor ajuste), siendo un proceso complicado y costoso en tiempo y recursos. La unión del ala es la que más fuerzas (tensiones en vuelo) soporta, de ahí su importancia.

[1] El orden no es significativo.

[2] Traducción del término inglés furnishing.

Integración del avión: básicamente consiste en ubicar las grandes piezas formadas en la estación anterior (fuselaje central-cabina, estabilizadores, alas) en gradas para su unión estructural.

Equipado: se dota al avión de la instalación eléctrica (mazos de cables) y de los sistemas de navegación y electrónicos, así como todo el equipamiento instrumental y de sistemas neumáticos e hidráulicos.

Pruebas Funcionales: es la estación más larga en duración y de las que proporciona más complejidad. Se utilizan gran cantidad de sistemas de medidas automatizados y simuladores. El objetivo es probar todos los sistemas del avión. Se divide en dos subestaciones:

1. **Pruebas funcionales interiores**
2. **Pruebas funcionales exteriores**

En las pruebas funcionales exteriores se realizan todas las pruebas relacionadas con los sistemas de combustible y en general pruebas con grandes medidas de seguridad, y las pruebas de comunicaciones que no se pueden realizar en el interior de un hangar.

Equipado de motores: el comienzo de esta estación es importante porque determinará la fecha para el pedido de motores al fabricante. Esto es importante puesto que supone un gran coste de inversión y afecta a la viabilidad del proyecto.

Decorar, amueblar: con estos términos tan genéricos nos referimos tanto al equipado interior del avión como a la infraestructura necesaria para dotar al avión de toda su funcionalidad.

Pintura: este es una de las operaciones de mayor duración y necesita de instalaciones muy preparadas y adecuadas para ello. Se utilizan hangares especializados con mascaras para las distintas partes del avión y con un gran control de presión y temperatura.

Línea de vuelo: es una de las estaciones más importantes y es la última donde los sistemas se ponen a prueba antes de la entrega al cliente. En esta estación se realizan los ensayos de vuelo, simulaciones de vuelo y puesta a punto de sistemas electrónicos de navegación y comunicación. También se realiza en esta estación el rodaje de motores.

Entrega y reparto: es la última estación y requiere como mínimo una semana de plazo, pues no sólo se realiza la entrega sino que es en esta estación cuando los pilotos del cliente prueban el avión y sugieren ciertas modificaciones si fuesen necesarias, hasta que se da el visto bueno. También es cuando tienen lugar todos los trámites de papeleo del avión como obtención de certificaciones, etcétera.

2.6. LA ESCALERA DE AVIONES

El término escalera es usado en aeronáutica para referirse a la planificación temporal del comienzo y finalización del proceso de ensamblaje de los aviones, así como su entrega a los clientes. El uso del termino "escalera" es obvio si nos fijamos en la figura 2.2. Podemos apreciar también una disminución en la duración de los aviones, la explicación de este fenómeno, en el que se profundizará en el capítulo V de esta memoria, se debe a la reducción de tiempos debido a las curvas de aprendizaje.

Podemos observar que una escalera es un diagrama de Gantt al más alto nivel de detalle del proceso y del cual obtenemos como información el comienzo del proceso y la entrega de los distintos aviones a lo largo de un plazo de tiempo.

El documento de planificación obtenido con el diagrama de Gantt se puede completar con información adicional relativa a los recursos y costes asociados a los mismos. Este tipo de documentos ofrece información muy importante de las consecuencias que tienen en las entregas (fase final) posibles retrasos en las primeras etapas del proyecto. Se observa en la figura 2.2 como las fechas están referidas a una fecha de comienzo del proyecto (T_0).

La escalera de aviones también es una fuente de información de las operaciones básicas y la secuenciación de las estaciones, así como de los planes de entrega y el

destino por clientes (generalmente países) de los aviones. Podemos observar que los primeros aviones ensamblados no se entregan inmediatamente a los clientes, se utilizan para la realización de pruebas y finalmente vuelven a entrar en el proceso de ensamblaje (no en todas las operaciones), para su entrega, en función de las necesidades de la planificación.

Antes de comenzar labores de ingeniería más concretas y detalladas es necesario, en general, ver la viabilidad de las grandes decisiones a adoptar, ya sea para cumplir criterios económicos como para cumplir plazos de entregas fijados con los clientes. De estas decisiones dependerán cuestiones de vital importancia como son el comienzo y fin de los aviones, el número de hangares para el ensamblaje, la distribución de contrataciones previstas a lo largo de la vida del proyecto y los gastos asociados a futuras inversiones.

Este es un proceso con realimentación, es decir, el departamento de ingeniería encargado del estudio de la fase de ensamblaje propone unos datos preliminares sujetos a posibles modificaciones, se genera una planificación previa, un cálculo de costes y con las conclusiones obtenidas tras un estudio de la viabilidad se vuelven a recalcular los datos previos de los que partimos. Estos datos son entre otros las duraciones previstas de las distintas tareas de las que consta el ensamblaje, los recursos asociados a esas tareas y el coste de dichos recursos. Todos estos cambios influyen necesariamente en la planificación del proyecto en su conjunto, por lo que es necesario modificarla continuamente.

La escalera de aviones es por lo tanto un documento necesario desde el comienzo de vida del proyecto del avión, es además en este primer momento cuando más necesidad hay de reajustes de planificaciones, puesto que hay muchas cuestiones por definir. La planificación no es exclusivamente una guía “del plan a seguir” sino que es una herramienta de vital importancia para definir el proceso de ensamblaje.

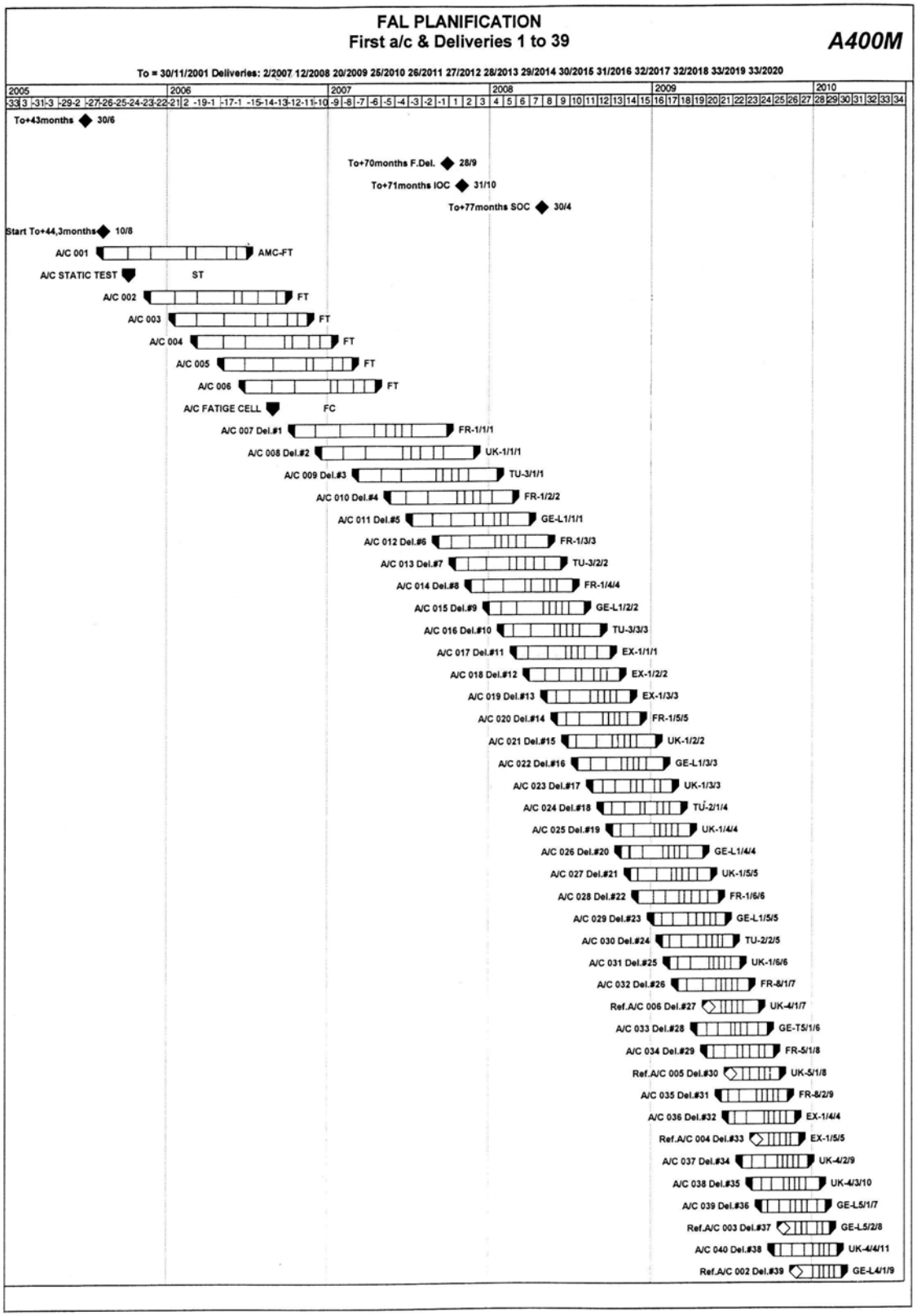


Figura 2.2. Ejemplo de escalera de aviones elaborada para el A400M.

3.GESTIÓN DE PROYECTOS

3.1. INTRODUCCIÓN

El objetivo de este apartado es profundizar en todo lo que conlleva la gestión de proyectos y particularmente en aquellos aspectos relacionados con la planificación.

La aplicación desarrollada tiene entre otras funciones la validación de tiempos. Esto es necesario para asegurar que las operaciones y la secuenciación definidas en el ensamblaje no sean incompatibles con los plazos de entrega de los aviones. Esta validación es parte del trabajo de la oficina del programa (PMO), que es la que se encarga de la gestión del programa del A400M.

La PMO tiene, además de la validación de fechas, otras misiones propias de la gestión y que estan relacionadas con la coordinación de un proyecto en el que intervienen distintos paises y en el que las decisiones que hay que tomar están marcadas por muchos factores. Es por tanto, conveniente estudiar las tareas propias que se derivan de la gestión de proyectos y más concretamente de la planificación.

Como hemos dicho, el uso del término programa en vez de proyecto es típico en aeronáutica y se debe a la globalización de proyectos que se presupone en un programa aeronáutico, ya que un programa está formado por los proyectos de diseño, manufacturación y soporte post-venta del avión.

3.2.CONCEPTOS GENERALES DE LA GESTIÓN DE PROYECTOS

3.2.1. Definición y características generales

Podemos definir la gestión de proyectos como la aplicación de conocimientos, habilidades, herramientas y técnicas para proyectar actividades con el fin de cumplir con las necesidades y expectativas de los involucrados en un proyecto.

La mayoría de los proyectos comparten muchas características:

- Los costes y las contrataciones son pequeños al principio, crecen a medida que se avanza en la realización y decrecen rápidamente cuando finaliza (figura 3.1).

- La probabilidad de completar con éxito el proyecto es más baja (y por lo tanto el riesgo y la incertidumbre más altos) al comienzo del proyecto. La probabilidad de éxito generalmente aumenta cuando el proyecto avanza en el tiempo.

- La capacidad de influir de los accionistas del proyecto en el producto y coste final es superior en la fase inicial y progresivamente va disminuyendo cuando el proyecto avanza. La mayor contribución a este fenómeno se debe al incremento del coste debido a correcciones cuando el proyecto está más avanzado.

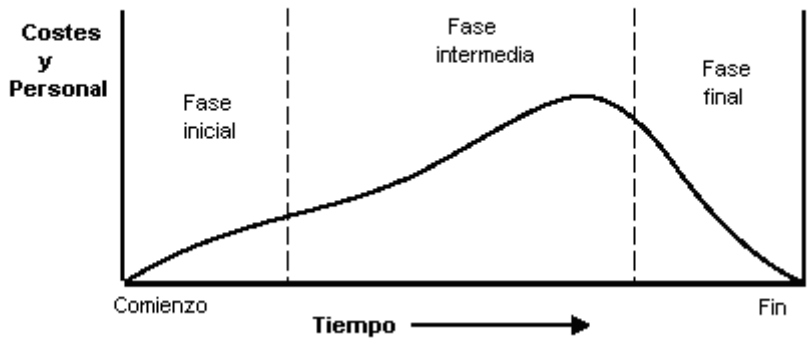


Figura 3.1. Ejemplo de ciclo de vida de un proyecto.

Las etapas que intervienen en el ciclo de vida de un proyecto (figura 3.2) son:

Fase I: Viabilidad. Formulación del proyecto, estudios de viabilidad, estrategias de diseño y aprobaciones.

Fase II: Planificación y diseño. Diseño base, costes y planificación temporal, condiciones y términos de contratos, y planificación detallada.

Fase III: Producción. Manufacturación, reparto, obras civiles, instalaciones, pruebas.

Fase IV: Facturación y puesta en marcha. Pruebas finales, mantenimiento.

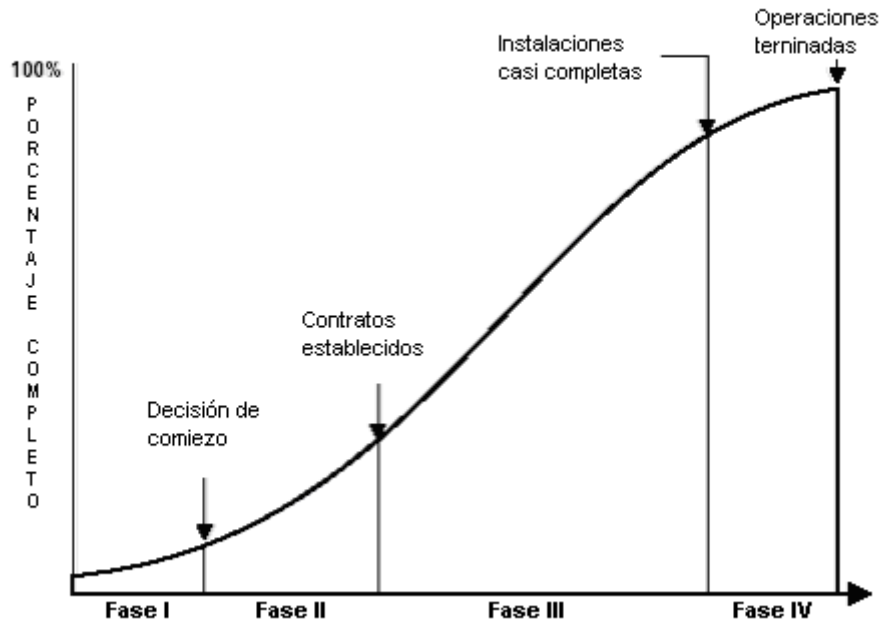


Figura 3.2. Construcción representativa del ciclo de vida de un proyecto.

3.2.2. Grupos de actividades

Tenemos cinco grupos de actividades o procesos de un proyecto, cada uno con una o más actividades:

INICIO: encargado de la identificación de cuando un proyecto o fase de un proyecto debe comenzar.

PLANIFICACIÓN:[1] definición y mantenimiento de un esquema de trabajo que permita alcanzar las metas previstas.

EJECUCIÓN*:* coordinación de personal y recursos para levar a cabo el plan de trabajo.

CONTROL: asegurar que los objetivos del proyecto se consiguen, para ello se monitoriza y se evalúa el progreso del proyecto y se toman medidas correctivas si son necesarias.

FINALIZACIÓN: formalizar que un proyecto o fase de un proyecto ha concluido.

En la figura 3.3 podemos observar la relación entre los distintos grupos.

Figura 3.3. Relación entre grupos de actividades.

[1] Se ha traducido el término Planning, como planificación pero también es válido y en algunos casos más adecuado el significado "Proyectar algo".

Entre los distintos grupos de actividades existe un solapamiento temporal (figura 3.4). La importancia o nivel de actividad de los procesos varía en función del desarrollo del proyecto. Actividades como la planificación tienen una mayor importancia al principio de la vida de un proyecto, para posteriormente perder "peso" frente a actividades como control o ejecución. Es importante planificar y coordinar el nivel de actividad de los distintos procesos para que el solapamiento (siempre existente) entre actividades no suponga un inconveniente en el desarrollo del proyecto.

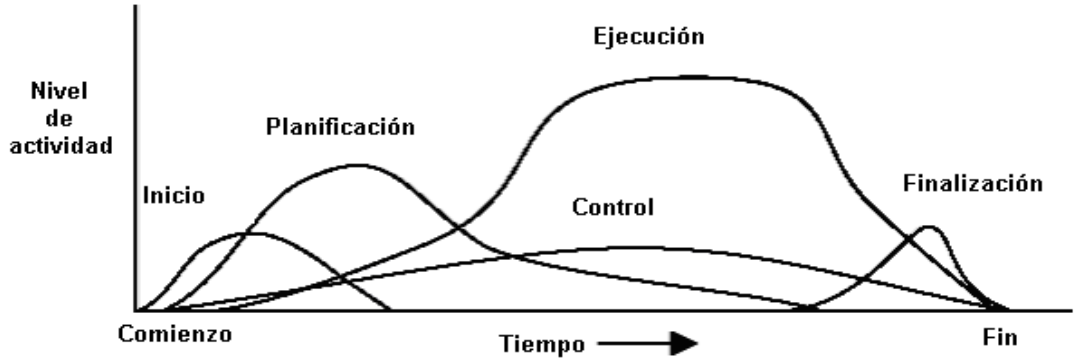


Figura 3.4. Solapamiento entre las distintas actividades.

Vamos a estudiar con más detalle las actividades relacionadas con la planificación, que debido a la naturaleza de nuestro proyecto son las que tienen más interés.

Las actividades que intervienen en la planificación son de las más importantes de un proyecto, puesto que generalmente implican hacer algo que previamente no se ha hecho. Comparándola con las otras actividades principales, la planificación es la que más cantidad de actividades o procesos requiere. Las principales actividades que intervienen en la planificación son:

- Planificación general o de alcance: se encarga de desarrollar un documento donde se fijan las bases para futuras decisiones del proyecto.

- Definición general: se encarga de dividir el proyecto en componentes o subproyectos más manejables.

- Definición de actividades: se encarga de identificar las actividades especificas que se tienen que realizar para la consecución de los distintos subproyectos.

- Secuenciación de actividades: identifica y documentar las dependencias entre actividades.

- Estimación de la duración de actividades: estimación de número de periodos de trabajo que son necesarios para completar las actividades individuales.

- Realización de un cronograma o planificación temporal: analizar la secuenciación de actividades, la duración de los recursos requeridos para la creación de una distribución temporal del proyecto.

- Planificación de los recursos: determinar que recursos (personas, equipos, materiales) y en que cantidad deben ser usadas para la realización de las actividades programadas.

- Estimación de costes: realización de una aproximación de los costes asociados a los recursos que se necesitan

- Identificación de costes: localización, dentro del coste total estimado, del coste debido a los trabajos individuales.

- Realización de la planificación del proyecto: obtener los resultados de las demás actividades derivadas de la planificación y realizar un documento coherente.

3.3. PLANIFICACIÓN

3.3.1. Introducción

En este punto abordaremos uno de los puntos más importantes de la gestión de proyectos, la gestión temporal o planificación. La planificación engloba todas las

actividades que se requieren para asegurar la finalización en el tiempo previsto del proyecto. Los procesos más importantes que intervienen son:

1. Definición de las actividades.
2. Secuenciación de las actividades.
3. Estimación de la duración de las actividades.
4. Elaboración de una planificación temporal.
5. Control de la planificación.

Para el estudio de cada uno de los procesos se definirán las datos necesarios (entradas), las técnicas y herramientas necesarias para obtener los resultados y un análisis de estos resultados (salidas).

En el siguiente apartado vemos una visión global de las los procesos que se derivan de la planificación.

3.3.2. Definición de las actividades

La definición de las actividades requiere identificar y documentar las actividades específicas a realizar para obtener los componentes y objetivos identificados en la división estructurada del trabajo.

3.3.2.1. Entradas en la definición de actividades

1. División estructurada del trabajo. Una división estructurada del trabajo es una agrupación orientada y divisible en componentes de los elementos del proyecto, que organiza y define la totalidad del proyecto.

2. Informe General. La justificación y objetivos del proyecto contenidos en el informe general deben ser considerados explícitamente durante la definición de las actividades.

3. Información histórica. La información histórica consiste en la información obtenida de proyectos anteriores o similares.

4. Restricciones. Las restricciones son factores que limitan las opciones del equipo de gestión de proyectos.

5. Supuestos. Los supuestos son factores, que para la planificación, son considerados como ciertos. Generalmente supone cierto riesgo para el proyecto que hay que evaluar.

3.3.2.2. Técnicas y herramientas para la definición de actividades

1. Descomposición. La descomposición implica subdividir los elementos del proyecto en otros más pequeños y más manejables para tener una mejor gestión de control.

2. Plantillas. Una lista o parte de una lista de proyectos precedentes, es usada a menudo como una plantilla para un nuevo proyecto.

3.3.2.3. Salidas de la definición de actividades

1. Lista de actividades. La lista de actividades debe incluir todas las actividades que se desarrollan en el proyecto.

2. Detalle de soporte. La lista de actividades debe estar documentada y organizada para facilitar las tareas a otras áreas de trabajo.

3. Estructura de trabajo actualizada. Partiendo de la división estructurada del trabajo, se deben identificar aquellos componentes (materiales de trabajo) que siendo importantes, no se dispone de ellos o se necesita una mayor clarificación o corrección de la descripción.

3.3.3. Secuenciación de actividades

La secuenciación de actividades implica identificar y documentar la dependencia entre actividades. La secuencia debe realizarse con el objetivo de posteriormente lograr un programa de trabajo realista y factible.

3.3.3.1. Entrada a la secuenciación de actividades

1. Lista de actividades.

2. Descripción del producto. Las características del producto afectan a la secuenciación de la actividad, por ejemplo, el layout de una planta a construir o las interfaces de los subsistemas en un proyecto software.

3. Dependencias preceptivas. Son las dependencias que son inherentes a la naturaleza del trabajo que se va a realizar. Habitualmente implican limitaciones físicas, por ejemplo, en un proyecto de electrónica, se debe realizar un prototipo antes de ser testado. Este tipo de dependencias también son conocidas como lógica dura (*Hard Logic*).

4. Dependencias discrecionales. Son las dependencias que son definidas por el equipo de gestión. Deben ser usadas con cuidado puesto que puede limitar posteriormente opciones de planificación. También son conocidas como lógica suave (*Soft Logic*).

5. Dependencias externas. Son las dependencias que implican una relación entre las actividades del proyecto y las actividades que no son del proyecto. Por ejemplo, dependencias con suministradores.

6. Restricciones.

7. Supuestos.

3.3.3.2. Herramientas y técnicas para la secuenciación de actividades

1. Método de diagrama por precedencia o **método PDM** (*Precedence diagramming Method*). Este es un método de construcción de diagramas donde las actividades son nodos unidos por flechas que muestran las dependencias entre las actividades.

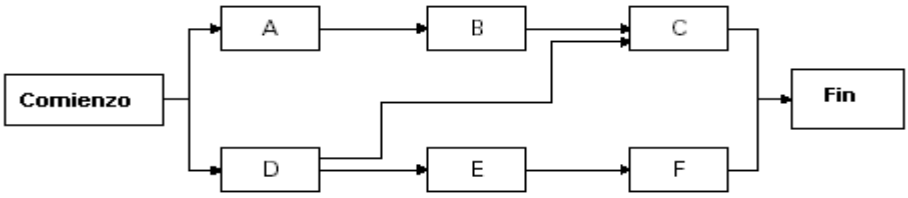


Figura 3.5. Ejemplo de método PDM.

2. Método de diagrama por flechas o **método ADM** (*Arrow Diagramming Method*). Este método usa flechas para representar las actividades y conectarlas a los nodos que muestran las dependencias.

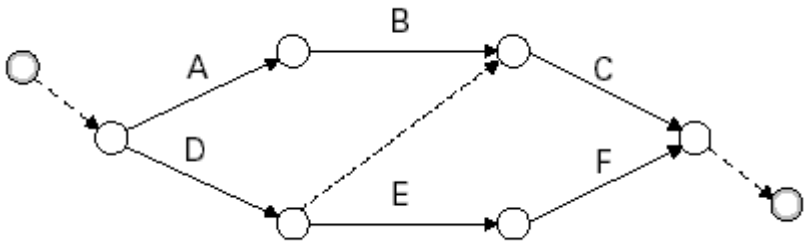


Figura 3.6. Ejemplo de método ADM.

3. Método de diagrama condicional. Técnica de construcción de diagramas, como por ejemplo GERT (*Graphical Evaulation and Review Technique*) y modelos de dinámicas de sistemas, permiten el uso de actividades no secuenciales, como lazos de realimentación o ramas condicionales. Los métodos PDM y ADM no permiten este tipo de actividades.

4. Plantillas de redes. Tipos de redes estandarizadas que pueden ser usadas para la preparación de diagramas para el proyecto.

3.3.3.3. Salidas a la secuenciación de actividades

1. Diagrama en red del proyecto. Consiste en un esquemático de la actividades del proyecto y las relaciones (dependencias) entre ellas.

2. Lista de actividades actualizadas.

3.3.4. Estimación de la duración de las actividades

La estimación de la duración de las actividades implica evaluar el número aproximado de periodos de trabajo que se necesitan para completar cada una de las actividades identificadas. Las personas o grupo de personas que están más familiarizadas con la naturaleza de las actividades específicas deberían ser las que hacen, o al menos aprueban, las estimaciones.

3.3.4.1 Entradas a la estimación de la duración de las actividades

1. Lista de actividades.

2. Restricciones.

3. Supuestos.

4. Recursos requeridos. La duración de la mayoría de las actividades se ve influenciada por sus recursos asignados.

5. Capacidad de los recursos. La duración de una actividades generalmente está influenciada por la capacidad de los recursos asignados, por ejemplo, un trabajador experto a media jornada pede equivaler a un trabajador novel a jornada completa.

6. Información histórica. La información tomada de precedentes, generalmente tomada de las siguientes fuentes:

- Ficheros de proyectos. Una o más de las organizaciones involucradas en el proyecto deben mantener registros de resultados de proyectos precedentes, que estén lo suficientemente detallados como para sean de ayuda en la estimación de la duración de actividades.

- Bases de datos comerciales de estimación de duraciones. Es frecuente la disposición comercial del uso de información histórica.

- Conocimientos y experiencia del equipo de trabajo.

3.3.4.2. Herramientas y técnicas para la estimación de la duración de actividades

1. Juicio experto. Las Duraciones son frecuentemente muy complicadas de estimar debido al gran número de factores que pueden influir. El juicio de un experto guiado por la información histórica se usa cuando no hay otra posibilidad de estimación. Un factor a tener en cuenta es que la estimación es una fuente de riesgo.

2. Estimación por analogía. Usa la duración actual de una actividad previa o parecida como base para el cálculo de una actividad futura.

3. Simulaciones. La simulación implica el cálculo de múltiples duraciones con diferentes supuestos. El más común es el análisis de Monte Carlo en el cual una distribución de resultados probables es definida para cada actividad y usada para calcular una distribución de resultados posibles para la totalidad del proyecto.

3.3.4.3. Salidas para la estimación de duraciones de actividades

1. Estimación de la duración de actividades. Se deben incluir indicaciones del rango de posibles resultados, por ejemplo, valor ± Δ variación ó % de la probabilidad de variación.

2. Bases de las estimaciones. Los supuestos que se han utilizado tienen que documentarse.

3. Lista de actividades actualizada.

3.3.5. Desarrollo de un calendario de trabajo

El objetivo de este proceso es determinar las fechas de comienzo y fin de las actividades. El desarrollo habitual es un proceso iterativo con aquellos procesos que son origen de datos, especialmente la duración.

3.3.5.1. Entradas para el desarrollo de un calendario de trabajo

1. Diagrama en red del proyecto.

2. Estimación de la duración de las actividades.

3. Recursos requeridos.

4. Descripción de los recursos compartidos. Se tiene que saber cuántos recursos hay disponibles, en qué momento y para qué socio del proyecto o grupo de trabajo.

5. Calendarios.

6. Restricciones.

7. Supuestos.

8. Adelantos y retrasos. Algunas de las dependencias requieren de una especificación de adelanto o retraso, para conseguir una interdependencia correcta.

3.3.5.2. Herramientas y técnicas para el desarrollo de un calendario de trabajo

1. Análisis matemático. El análisis matemático implica el cálculo teórico de comienzo y fin de todas las actividades sin incurrir en las limitaciones impuestas por la utilización de recursos compartidos. Los datos obtenidos no son el calendario, pero indican los periodos de tiempo en los que las actividades deberían ser ubicadas dada la limitación de recursos y otras restricciones. Los análisis matemáticos mas conocidos son:

   - CPM (*Critical Path Method*). Calcula una única y determinada fecha de comienzo y fin para cada actividad, basándose en una red lógica especifica y secuencial y una única duración estimada. El objetivo del CPM es determinar que actividades tienen menos flexibilidad para ser ubicadas temporalmente.

   - GERT (*Graphical Evaluation and Review Technique*). Permite tratamiento de probabilidades tanto de redes como de estimación de duración de actividades.

   - PERT (*Program Evaluation and review Technique*).Usa lógica secuencial de redes y una duración estimada por media ponderada para calcular la duración del proyecto. PERT en sí mismo es poco usado aunque PERT como estimación es frecuentemente usado en cálculos del camino crítico.

2. Compresión de duración. La compresión de duración es un caso especial de análisis matemático que busca formas de acortar la duración del proyecto sin modificar las características generales del proyecto (para cumplir con las fechas impuestas en los objetivos del calendario). Para ello se utiliza técnicas basada en comprimir el tiempo sin aumentar el coste del proyecto y la realización de tareas en paralelo que normalmente se realizan en serie.

3. Simulación.

4. Medición. Nivelación de forma heurística de recursos. Los análisis matemáticos frecuentemente dan como resultado cronogramas que asocian más recursos de los disponibles a ciertos periodos de tiempo. Para solventar eso es frecuente el

uso de técnicas heurísticas como asociar primero los recursos a las tareas consideradas como críticas. La nivelación de recursos puede llegar a ser una tarea más ardua que la realización de un calendario preliminar.

5. Software para la gestión de proyectos.

3.3.5.3. Salidas al desarrollo de un calendario de trabajo

1. Calendario del proyecto. Incluye la fecha planeada de comienzo y la fecha esperada de finalización para cada actividad. El calendario del proyecto o cronograma de tareas puede ser presentado de diversas formas. Las más frecuentes son las siguientes:

- Diagrama en red con información de fechas añadidas. Estos diagramas muestran información tanto de la lógica del proyecto como del camino crítico.

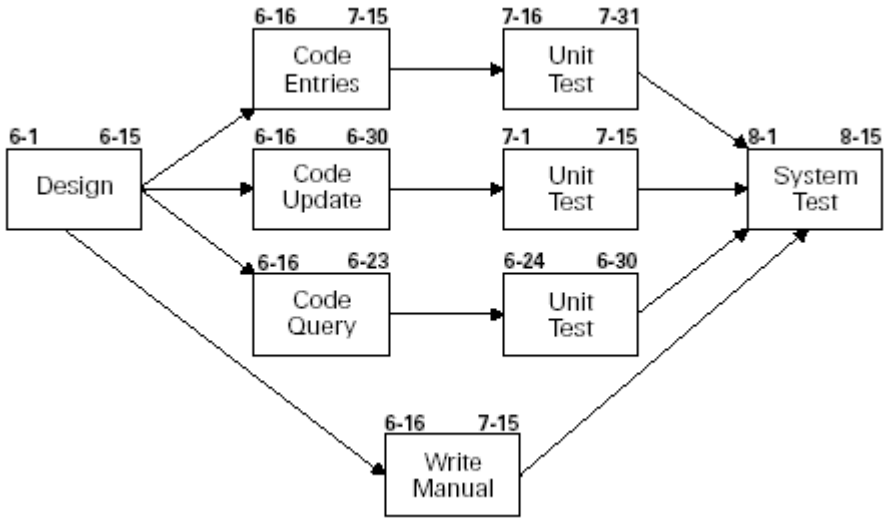


Figura 3.7. Ejemplo de diagrama con fechas.

- Diagrama de barras o diagrama de Gantt. Muestra la fecha de comienzo y fin de las actividades y la duración esperada, pero normalmente no muestra las dependencias. Es muy fácil de leer y se usa frecuentemente en las presentaciones de gestión.

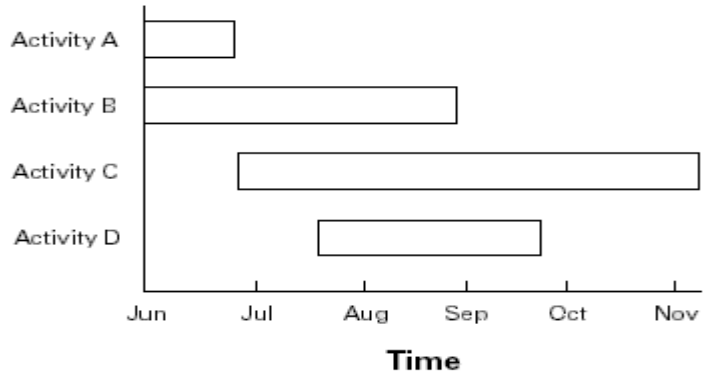


Figura 3.8. Ejemplo de diagrama de Gantt.

- Gráfico de hitos. Parecido al diagrama de Gantt, pero identificando el comienzo o la finalización de las distintas partes del proyecto así como las interfaces externas del proyecto.

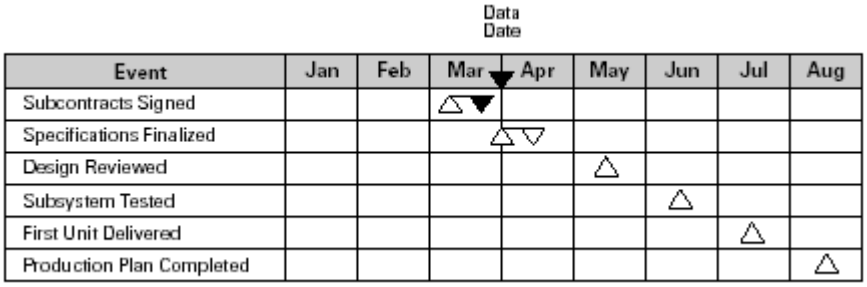


Figura 3.9. Ejemplo de diagrama de hitos.

- Diagrama de red escalado en tiempo. Es una mezcla entre diagramas de red y diagrama de Gantt en el que se muestra la lógica del proyecto, la duración de las actividades e información del calendario.

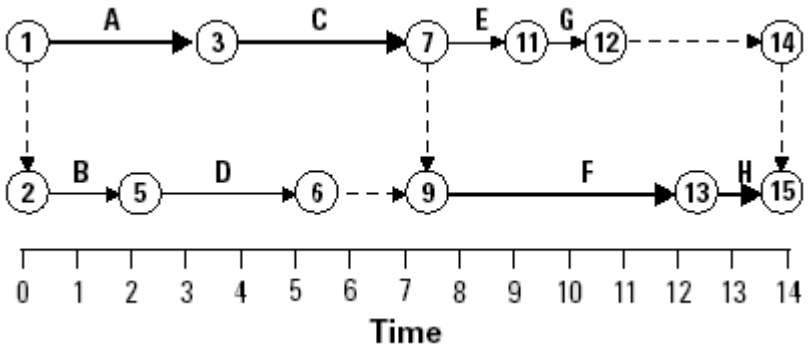


Figura 3.10. Ejemplo de diagrama escalado en tiempo.

2. Documentación de ayuda o soporte. Al menos se debe documentar todos las suposiciones hechas y las restricciones que se han tenido a la hora de realizar la

planificación temporal. La cantidad de información varia en función de las necesidades, pero entre ellas puede estar la elaboración de una planificación alternativa o un asesoramiento de riesgos.

3. Plan de gestión de la planificación. Define cómo deben ser tratados los cambios que sufra la planificación. Puede ser de forma informal o formal, o de forma rigurosa o teniendo en cuentas alternativas generales.

4. Lista de necesidades de recursos actualizada.

3.3.6. Control de la planificación

Se encarga de controlar los cambios que se puedan producir en el calendario o programa de trabajo para conseguir que esos cambios resulten beneficiosos.

3.3.6.1. Entradas al control de la planificación

1. Calendario de trabajo.

2. Creación de informes. Los informes de las actuaciones seguidas en la realización del proyecto nos permite saber si se han cumplido las fechas previstas.

3. Solicitud de cambios. Esta solicitud puede presentarse de diversas formas, oral o escrita, de forma directa o indirecta, como iniciativa interna o externa, obligatoria u opcional. Estos cambios pueden dilatar el proyecto o acelerarlo.

4. Plan de gestión de la planificación.

3.3.6.2. Herramientas y técnicas para el control de la planificación

1. Sistema de control de cambios al programa. Define los procedimientos por los cuales se van a realizar los cambios a la planificación. Incluye todas las autorizaciones para el cambio.

2. Medición de actuación. Una parte importante del control es decidir cuando o que magnitud de variación requiere una intervención correctiva. Existen técnicas de medición que ayudan a la toma de estas decisiones.

3. Planificación adicional. Algunos proyectos deben ejecutarse exactamente de acuerdo a la planificación. Cambios futuros implican revisar la duración estimada de las actividades, la secuencia de actividades o el análisis de cronogramas alternativos.

4. Software de gestión.

3.6.3.3. Salidas al control de la planificación

1. Actualización de la planificación.

2. Acciones correctivas. Generalmente se realizan para asegurar que una actividad finaliza a tiempo o con el menor retraso posible.

3. Aprendizaje. Las causas de las variaciones, los razonamientos que condujeron a la toma de medidas correctivas, y otro tipo de aprendizaje derivado del control de la planificación debe ser documentado y por lo tanto pasar a ser parte de la base de datos de información histórica del proyecto a realizar y para otros proyectos.

4. EL SISTEMA INFORMÁTICO EN LA EMPRESA

4.1. INTRODUCCIÓN

El conocimiento del sistema informático de una empresa de ingeniería nos permitirá entender mejor las necesidades de la empresa y cúales son las tendencias actuales en soluciones software para la definición y gestión de procesos industriales. La realización de este apartado será genérica, pero se indicarán cuales son esas necesidades en EADS CASA, esto nos permitirá saber la ubicación, dentro de la infraestructura de CASA, de la aplicación de planificación desarrollada.

Para entender el soporte informático necesario, se hará un estudio de las herramientas informáticas de las que se disponen y su utilidad. Puesto que las necesidades de comunicación e interoperabilidad son crecientes también hablaremos de la migración de aplicaciones locales a aplicaciones residentes en un servidor web, y por lo tanto accesibles desde una intranet.

No es el objetivo de este proyecto realizar un análisis profundo de la infraestructura informática de la empresa, sino tener una visión global.

4.2. APLICACIONES Y SISTEMAS

En una empresa de grandes dimensiones, con muchos departamentos y usuarios del sistema informático, es necesario tener un control sobre las aplicaciones instaladas en los ordenadores. Es por ello que se utilizan sistemas operativos de red. Los plataformas más utilizados son las de entorno UNIX y WINDOWS NT. En el caso de CASA el sistema operativo para la mayoría de los PC's es WINDOWS NT y sólo en las maquinas utilizadas para aplicaciones de diseño grafico se trabaja en entorno UNIX. Los permisos para los usuarios son controlados por el departamento informático de la empresa que son las personas que tienen acceso como administradores de red.

Existen aplicaciones comunes, utilizadas por todos los departamentos, como son todas las aplicaciones ofimáticas. En el caso de CASA son las aplicaciones del paquete MS-Office (Word, Excel, Access, PowerPoint). También se dispone de un servicio de mensajería interno y acceso a correo externo, para los que tengan autorización, como es la aplicación Notes de Lotus. Para cualquier otra aplicación considerada de ámbito más característico, como MS-Project, se necesita una autorización del administrador de red, y una instalación, realizada de forma remota por el administrador, en el PC del usuario.

Todas las aplicaciones utilizadas se encuentran en el servidor de aplicaciones. Es también muy importante la velocidad y los medios de transmisión de la intranet y red de área local para no provocar cuellos de botella en la ejecución de las aplicaciones. En el caso de CASA la red es Fast-Ethernet a 100 Mbps.

4.2.1. Lenguajes de programación

Existen multitud de soluciones software en el mercado, pero por su coste o porque la necesidad es muy especifica, es muy frecuente que en las empresas se realicen programas propios o adaptaciones de otros. Es por ello que para trabajar en una empresa, especialmente si esta dedicada a la ingeniería, es importante el conocimiento de los lenguajes de programación.

De las características propias de cada lenguaje de programación se derivan sus potencialidades de uso. Es por ello que el uso que se da a los lenguajes secuenciales (C, Fortram, Pascal,...) sea distinto a los orientado a objetos (C++, Java,...) o lenguaje de objetos (Visual Basic).

Es muy común el uso del lenguaje C en sistemas de control o cuando se requiere mucho uso de librerías que profundizan en comandos de bajo nivel. En el ensamblaje del A400M, es típico este uso para la creación de simuladores y muchas aplicaciones específicas para pruebas funcionales.

La mayoría de las aplicaciones que existen en el mercado están realizadas con una programación orientada a objetos, pues son muy importantes la característica que proporcionan este tipo de lenguajes: herencia, sobrecarga, polimorfismo, etcétera.

Es por lo tanto importante, para entender el funcionamiento de una aplicación a un nivel de detalle mayor, el conocimiento del lenguaje. Existe otra cualidad también fundamental. Es frecuente que las aplicaciones ofrezcan sus librerías de objetos para posibles modificaciones y mejoras a cargo del cliente, o como labores de consultoría para la empresa creadora del software, esto es lo que se conoce como customización[1] de una aplicación, es decir, una aplicación en su origen muy genérica, tiene la posibilidad de adaptarse a las necesidades concretas del cliente. Esto se traduce en interfaces gráficas personalizadas, presentaciones de resultados siguiendo el estándar de la empresa, cálculos con fórmulas concretas, etcétera.

Lenguajes como Java están en expansión por facilitar la tarea de programación con entornos de red, Internet y la creación de aplicaciones cliente-servidor. Esto es debido a que es muy fácil la utilización de librerías diseñadas específicamente para este uso. Otros como Visual Basic son muy utilizados por la facilidad de desarrollar interfaces gráficos y control por eventos, esto permite hacer de forma sencilla programas que requieran comunicación con el usuario. Otro factor a importante es que Visual Basic es el lenguaje que utilizan las aplicaciones de Microsoft, que además pone a disposición del usuario librerías con objetos de Visual Basic para desarrollar aplicaciones.

[1] Término que proviene del verbo inglés *customize*, adaptar a las necesidades de algo o alguien.

4.2.2. Aplicaciones CAD

CAD es el acrónimo de *Computer Aid Design* (Diseño Asistido por Computador). Son aplicaciones que en general requieren de máquinas que tengan una tarjeta gráfica potente, un mínimo de memoria RAM de 256MB y un procesador rápido. En EADS CASA las aplicaciones CAD utilizadas son Catia para entornos 3D y sólidos, Autocad para entorno 2D y planos, y Delmia para simulaciones en entornos 3D. La intención es migrar en un futuro próximo de Delmia a Windchild. Hemos incluido herramientas que si bien no son de diseño propiamente dicho, como por ejemplo Delmia, su uso por los departamentos de diseño y de desarrollo es muy importante, puesto que son las herramientas que nos permiten la visualización de un proceso con toda la potencia de su interconectabilidad con herramientas propiamente CAD como es CATIA.

En la definición de un proceso, las herramientas CAD y de simulación juegan un papel fundamental puesto que nos van a poder visualizar cómo es el proceso y definir las operaciones. Herramientas como Catia y Delmia necesitan conectarse a una base de datos, sistema PDM (*Product Data Management*), donde están todos los materiales (realmente son ficheros) que definen un modelo. Esta base de datos se conoce como Vault (Baúl), por ser el lugar donde se accede para buscar y salvar los modelos.

4.2.3. Bases de datos

Cada día es más importante el concepto de dato único. Debido al cambio de filosofía de trabajo en las empresas, las empresas tienden hacia la ingeniería concurrente, por lo que es importante el intercambio de información actualizada y el fácil acceso al trabajo desarrollado por otros.

Las bases de datos son el motor del soporte informático de la empresa, puesto que son las herramientas que proporcionan un fácil acceso a la información, con todas las herramientas de gestión y supervisión asociadas. Es esta necesidad la que lleva a utilizar progresivamente bases des datos con mayor velocidad de acceso a la información, capacidades multiusuario y gran capacidad de memoria. Las bases de datos son también la pieza fundamental de las aplicaciones que presentan un acceso en red, vía Web u otra

aplicación cliente – servidor, puesto que permiten fácil consulta y modificación de datos con las nuevas tecnologías y lenguajes, como SQL.

Son las necesidades crecientes de gestión, capacidad y adecuación a las nuevas tecnologías lo que lleva al cambio de bases de datos que ofrecen menos servicios a bases de datos más potentes. En CASA esto se traduce en el cambio de RDB (Real Data Base) a ORACLE, base de datos que se esta imponiendo en el ámbito empresarial.

4.2.4. Intranet

En una empresa con una gran distribución geográfica, son lógicas las necesidades de tener un sistema de comunicación fiable, rápido y fácil de gestionar. Una Intranet es una red de comunicaciones de ámbito corporativo. Existe la tendencia de migrar desde aplicaciones de ámbito local a aplicaciones residentes en un servidor y accesibles mediante aplicaciones clientes, como por ejemplo un navegador tipo Netscape o Internet Explorer. Por lo tanto es frecuente el desarrollo de nuevas aplicaciones usando lenguajes como ASP, JavaScript, VisualBasicScript y XML. En ellas los ordenadores clientes se usan como terminales de acceso a las aplicaciones que corren en ordenadores más potentes y que tienen un fácil y controlado acceso a las bases de datos, por lo que es un acceso a la información siempre actualizado y controlado por el servidor.

4.2.5. Sistemas PDM

PDM es el acrónimo de *Product Data Management* (Gestión de datos del producto). Las Oficinas Técnicas suelen realizar bien la tarea de almacenar sistemáticamente planos y diseños de componentes, piezas y ensamblajes. Pero generalmente no tienen unos datos fáciles de comprender e identificar referentes a atributos tales como 'tamaño', 'peso', 'dónde se usan', etcétera.

Los sistemas *Data Management* son capaces de gestionar tanto los atributos de los datos de los productos ('material', 'fecha de diseño',...), como los datos en sí (planos, diseños 3D ó documentos Word), e incluso la relación que existen entre todos ellos, gracias a un sistema de Base de Datos Relacional.

La clasificación debe ser una característica fundamental de los sistemas PDM. Toda la información de datos con características similares es agrupada dentro de 'clases'. Una clasificación más detallada es posible gracias al uso de 'atributos' para describir las características fundamentales de cada componente dentro de una clase.

Los componentes son introducidos en la Base de Datos bajo una gran variedad de clases que se ajustan a las necesidades del negocio. Esto permite que todos los componentes de trabajo de la empresa puedan ser organizados en una estructura jerárquica fácilmente trazable. A cada parte del producto se le pueden asignar una serie de atributos. Esto posibilita la creación de Listas de Materiales fácilmente a partir de los diseños. Si seleccionamos un determinado producto, las relaciones entre sus ensamblajes y entre las partes que forman estos ensamblajes se mantienen. Esto quiere decir que se puede abrir una completa Lista de Materiales, incluyendo los documentos y las partes, para el producto completo o para determinados ensamblajes dentro de ese producto.

El reto es minimizar el tiempo de lanzamiento al mercado mediante el uso de la ingeniería concurrente, pero manteniendo el control de los datos y distribuyéndolos automáticamente a todos aquellos que los necesitan, cuando los necesitan. El modo en que los sistemas PDM afrontan este reto es guardando los datos maestros en una 'cámara acorazada' dentro del sistema, y solamente una vez. De esta manera, su integridad está asegurada, y todos los cambios que se realicen serán monitorizados, controlados y grabados.

Las copias duplicadas de los datos maestros, por otra parte, pueden ser distribuidas libremente a usuarios de varios departamentos para diseño, análisis o aprobación. Los nuevos datos son entonces devueltos a la 'cámara acorazada'. Cuando se realiza un cambio en los datos, lo que realmente ocurre es que una copia modificada de esos datos, firmada y fechada, se almacena en la 'cámara acorazada', junto a los datos anteriores sin modificar, que permanecen en su antigua forma como un registro permanente. Este el principio básico de los más avanzados sistemas PDM.

En resumen las ventajas principales de usar un sistema PDM son las siguientes:

1. Reducción de tiempo: se acorta el tiempo de acceso a los datos de diseño, ya que se pueden visualizar instantáneamente cuando se quiera, lo que reduce drásticamente el tiempo destinado a cualquier tarea. Soporta eficazmente la gestión de las tareas de la Ingeniería Concurrente. Permite a los miembros autorizados acceder a los datos más relevantes, permanentemente, asegurando que se trata de las últimas versiones de los diseños.

2. Mejora de la productividad del diseño

3. Seguridad 100% de los datos: el concepto de 'cámara de seguridad' de los sistemas PDM facilita que, mientras los datos son accesibles instantáneamente para aquellos que lo necesitan, los datos y documentos maestros permanecen fiables y seguros. A su vez, se crea un registro con todas las modificaciones y versiones de los datos.

4.2.6. Sistema ERP

Por sistema ERP, acrónimo de *Enterprise Resource Planning* (planificación de los recursos de la empresa), se conoce a las aplicación de gestión empresarial que integra el flujo de información, consiguiendo así mejorar los procesos en distintas áreas (financiera, de producción, logística, comercial, recursos humanos).

El sistema ERP más implantado a nivel mundial es SAP R/3. SAP es el acrónimo y nombre de la empresa creadora de R/3 (Sistemas, Aplicaciones y Productos para el proceso de datos). CASA utiliza un software propio que controla las ordenes de trabajo, el inventario, etcétera, denominado SPRINT, pero en un futuro es intención de la empresa sustituirlo por SAP R/3. Es por ello que voy a hablar de SAP R/3, para entender cuál es la utilidad y potencialidad de estos sistemas ERP.

SAP es un sistema multi-idioma, multi-sociedad, multi-organización, multi-divisa, multi-ejercicio fiscal, multi-plan de cuentas, multi–plataforma basado en sistemas abiertos y en el concepto de cliente-servidor. Es pues, un software especialmente

pensado para grandes empresas, con multitud de sedes en distintos países y con un afán integrador. El objetivo es la satisfacción de los requerimientos funcionales de la empresa. Para ello es fundamental la integración en tiempo real y el principio de dato único.

Los beneficios esperados de la utilización de un sistema ERP son:

1. Ayudar a la mejora en la gestión de la empresa

2. Facilitar la optimización de recursos necesarios para las distintas áreas de la empresa

3. Simplificar la programación de las actividades

4. Ayudar a la procedimentación de las acciones

5. Suministrar información de apoyo para las decisiones de los gestores

6. Integración de sistemas

Para dotar a SAP de todas las funcionalidades, SAP se divide en módulos (figura 4.1) que se conectan a una base de datos, generalmente ORACLE, con peticiones del tipo cliente-servidor.

El lenguaje en el que se creó SAP, es un lenguaje propio con orientación a objetos denominado ABAP4. SAP dispone de facilidades para construir aplicaciones en ABAP4 y que sustentadas en los objetos que manipula SAP, permiten una customización del sistema.

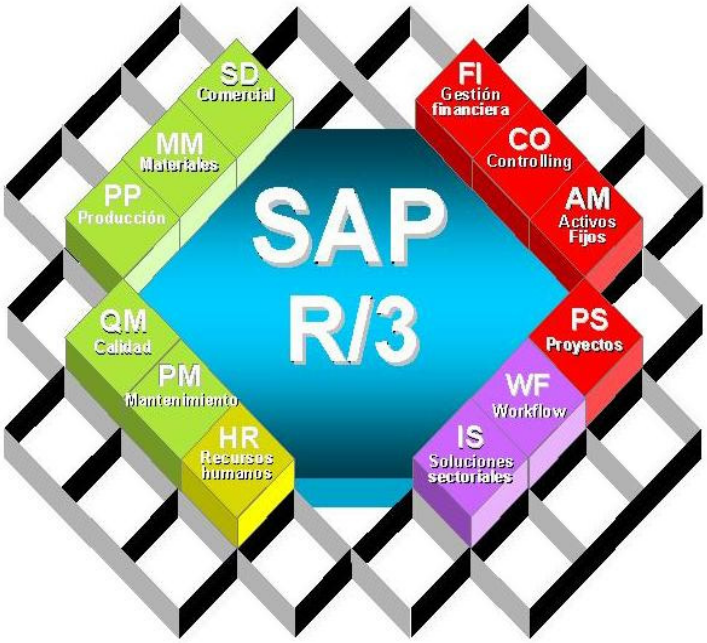


Figura 4.1. Módulos de SAP R/3

4.3. SOLUCIONES GLOBALES

Por soluciones globales nos referimos a aquellas aplicaciones que integrándose en la estructura informática existente en la empresa dan solución a todos los aspectos de una problemática, esa solución da cobertura a todos los procesos que se realizan y esta totalmente informatizada.

No se pretende excluir otro tipo de soluciones ni aseverar que las que se propongan en este punto son las mejores. Para la realización de este apartado se ha hecho un análisis de las necesidades que se tienen y en función de las mismas cuales son las características que debe disponer un software que pretende resolverlas. El análisis de la arquitectura de las aplicaciones se ha realizado en función de las más utilizadas en este tipo de aplicaciones.

Estudiaremos las características que deben tener las soluciones de este tipo para dos problemáticas distintas, la gestión de proyectos y el estudio del ensamblaje.

Para la gestión de proyectos debemos tener una solución que centrándose en los datos existentes proporcione herramientas de información y control (figura 4.2).

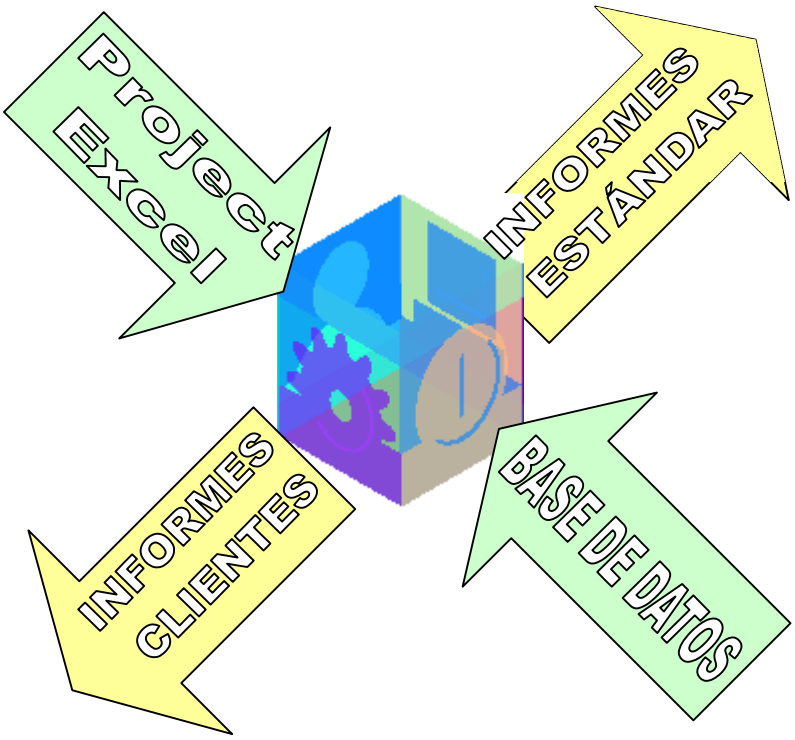


Figura 4.2. Ejemplo de Entradas-Salidas a un sistema de Gestión

Las soluciones para la definición de los procesos de ensamblaje o manufacturación se conocen como MPM (*Manufacturing Proces Management*). Para proporcionar una solución global se deben presentar interfaces (figura 4.3) con las herramientas informáticas origen de datos (PDM) y una vez definidos los procesos deben volcarse en el sistema de gestión de producción (ERP).

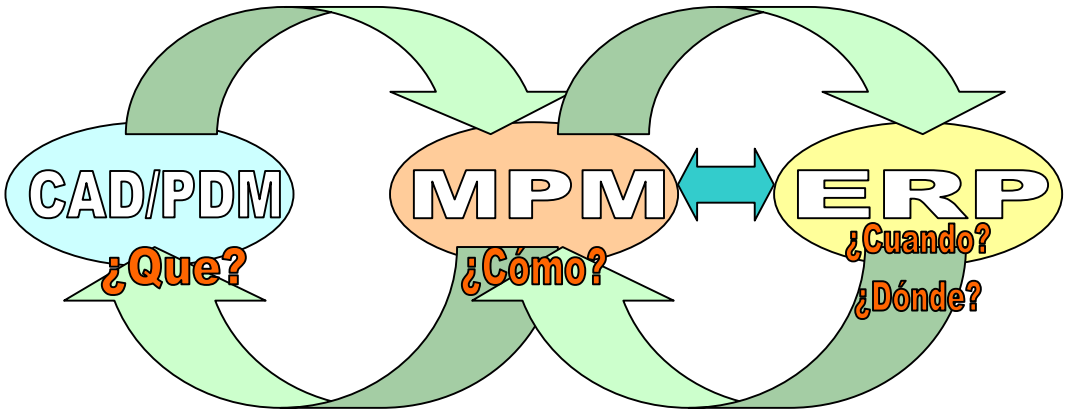


Figura 4.3. Esquema de flujo de información en una solución global.

4.3.1. Solución para la gestión de proyectos

Estudiamos anteriormente en que consistía la gestión de proyectos. Para gestionar proyectos utilizando un software y que este se pueda comunicar con el resto del sistema informático, es necesario que el software disponga de interfaces con las aplicaciones que son origen de datos, como por ejemplo MS Project, y con las que son posibles destinos, sistema ERP o MPM. Además los datos deben ser tratados como únicos y modificados en tiempo real, por lo que se hace necesario el uso de una base de datos. Para poder disponer de facilidades de gestión desde una red, ya sea vía Internet o intranet, es necesario, además de una base de datos disponer de una arquitectura en la que tengamos un servidor Web.

La gestión de proyectos a distancia es una facilidad importante, entre otros motivos porque permite la consulta a una base de datos sin tener que tener instalada la aplicación de gestión. La mayoría de soluciones globales tienen un módulo de consulta, informes y visualización mediante navegador Web. El núcleo de la aplicación es generalmente un servidor de peticiones que comunica las peticiones con los módulos que las gestionan, ya que para el desarrollo de estas aplicaciones es muy importante el concepto de modularidad (figura 4.4).

La aplicación de gestión debe ser capaz de proporcionar herramientas para la gestión de varios proyectos organizados de forma jerárquica y ser multiusuario. Debe ser una herramienta de planificación o en su defecto tener interfaces con herramientas de planificación como MS Project. Debe facilitar la gestión y distribución de recursos, dotar al sistema de control de gastos y de riesgos y tener facilidades para la realización de informes con capacidades gráficas. Además del módulo de acceso Web, es importante dotar a la aplicación de un módulo de seguimiento de proyectos con capacidad de acceso remoto. Todo estas facilidades se deben proporcionar basándonos en una arquitectura cliente-servidor.

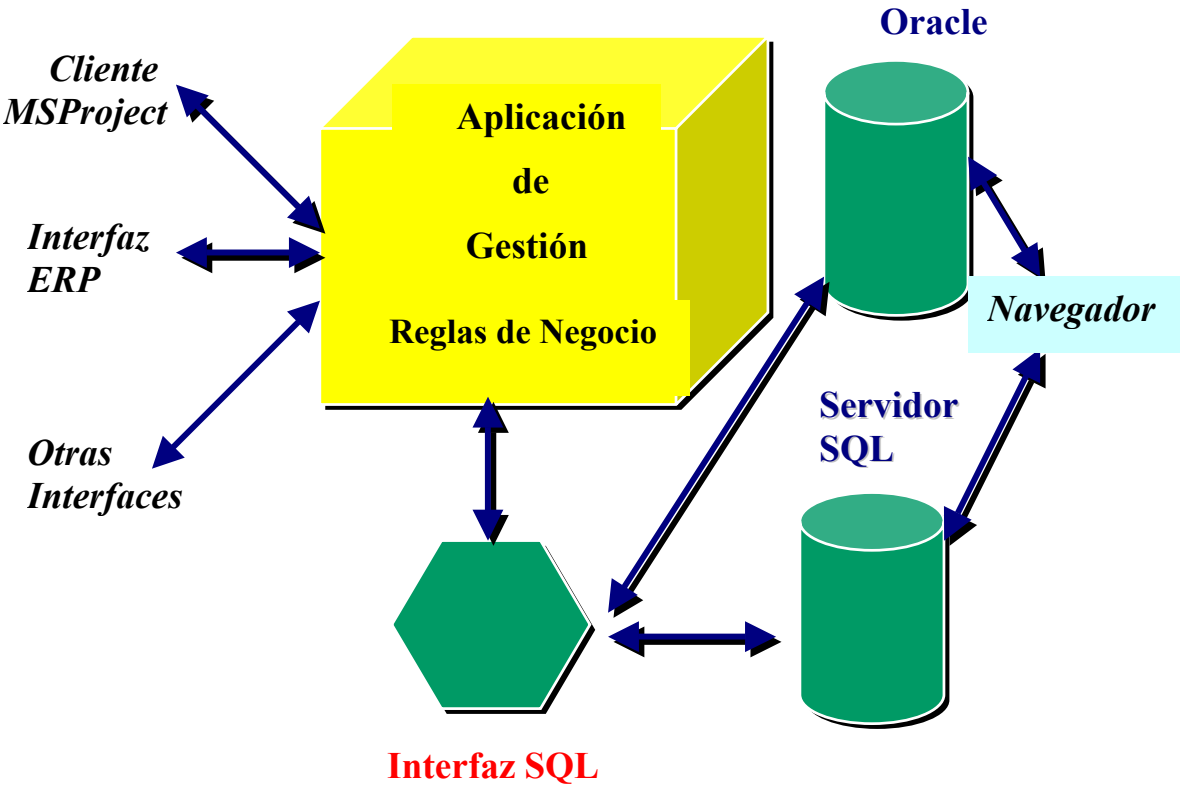


Figura 4.4. Esquema de interfaces y acceso a una base de datos vía web.

4.3.2. Solución para la elaboración de procesos de ensamblaje

Vamos a estudiar la arquitectura que debe proporcionar una solución MPM. No basta con disponer de una aplicación potente, es necesaria una ínter conectividad con las otras herramientas software de las que se dispone. También es necesario que permita un fácil acceso a la información por los distintos departamentos que intervienen en un proyecto. Para la definición de un proceso es muy importante la conectividad con las herramientas gráficas de diseño (Catia, Autocad), con las bases de datos con información de materiales y recursos (PDM) y con el sistema de gestión de la producción (ERP).

Para dotar a la solución MPM de interacción con el cliente a través de un navegador Web y de acceso a información, es conveniente una arquitectura de tres capas (figura 4.5).

La aplicación debe ser customizable, por lo que debe tener herramientas para ello. Es típico disponer de un administrador de objetos, lo que permite la creación de objetos nuevos, generalmente creados en C++ o Visual Basic. Esto permite también la creación de interfaces con otras herramientas software como pueden ser MS Excel o MS Project.

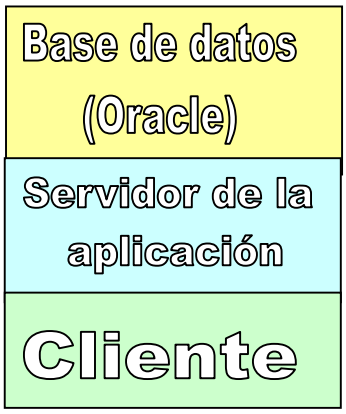


Figura 4.5. Arquitectura en capas para acceso a una base de datos.

Para disponer de acceso mediante aplicaciones Web se debe disponer de servidores Web. Es muy frecuente la utilización de tecnología Java en este tipo de aplicaciones.

Figura 4.6. Arquitectura para acceso a una base de datos vía web.

Una solución global MPM debe dar respuesta a todas las necesidades que surgen a la hora de definir un proceso. Por lo tanto debe tener módulos o pequeña aplicaciones que se encarguen de las siguientes tareas:

- Secuenciar Procesos. Una forma típica de realizar esta secuenciación es mediante el uso de diagramas de PERT o mediante diagramas de Gantt.

- Asignar tiempos a las tareas u operaciones. También es importante la posibilidad de dotar a la aplicación de la opción de utilizar técnicas de balanceo de tiempos. Para la asignación de tiempos es conveniente el uso de una base de datos donde se almacenan los tiempos tipo

- Asignar costes a las operaciones. El módulo de costes es una facilidad a la gestión del proyecto. El incluirlo en la definición de procesos nos permite disponer de un dato más para la elección de un proceso frente otro.

- Generar secuencias de ensamblaje, estudio de interferencias, colisiones. Módulo con gran conectividad con las aplicaciones CAD. Son simulaciones en 3D.

- Simulación de flujos. Generalmente se utilizan técnicas de simulación de eventos discretos. La funcionalidad de este módulo es muy importante puesto que es la herramienta fundamental para el análisis de las líneas de producción. Es interesante la inclusión de la posibilidad de dotar a la aplicación de técnicas de optimización mediante el uso, por ejemplo, de algoritmos genéticos.

Para la integración de la aplicación con las herramientas CAD, PDM y ERP, además de otras posibles aplicaciones de interés, son muy importantes los módulos que se podrían denominar de interfaces. Estos módulos van muy ligados a la posibilidad de customización y de dotar a la aplicación de herramientas de modularidad y acceso a bases de datos. Como grupos importantes tenemos los de importación de datos y los de exportación. Las principales fuentes de importación de una aplicación MPM son las aplicaciones CAD y los sistemas PDM. Como datos a importar principalmente tendríamos, la lista de materiales, los Layout 2D y los modelos 3D. De la lista de materiales se obtiene de forma automática los elementos que van a formar parte en una operación, de los layout en 2D se obtienen parte de los recursos y los modelos 3D se utilizan principalmente como origen de datos del módulo encargado del estudio del ensamblaje (interferencia, colisiones, simulaciones, etcétera). Una vez definido el proceso (identificación de operaciones, tiempos, secuenciación) los datos son

exportados al sistema ERP. Es muy importante para la valía de un sistema MPM la posibilidad de que esta exportación se haga de forma automática.

En la figura 4.7 mostramos una posible configuración (módulos e interfaces) de una solución MPM que cumple con los requisitos anteriormente descritos.

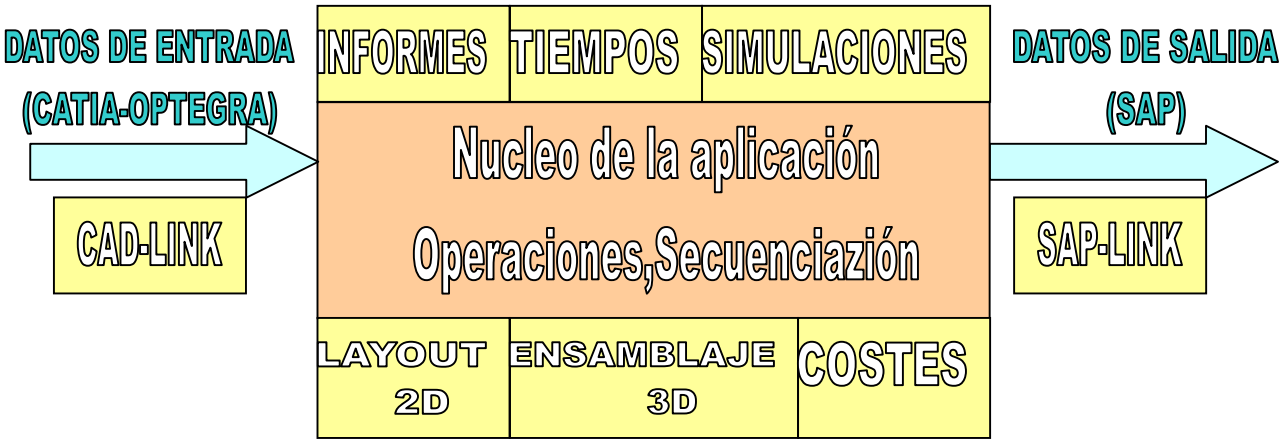


Figura 4.7. Módulos e interfaces de un sistema MPM.

5.CONCEPTOS DE PLANIFICACIÓN EN AERONÁUTICA

5.1. INTRODUCCIÓN

Es importante para entender el correcto funcionamiento de la aplicación y también el desarrollo en su creación, estudiar algunos conceptos relacionados con la planificación. No pretendemos un estudio riguroso, sino entender qué buscamos al hacer una planificación y qué tenemos que tener en cuenta, desde un punto de vista intuitivo. Podríamos decir que la planificación básicamente es una cuestión de sentido común, es decir, dados unos objetivos y unas realidades se deben adecuar las cosas para que funcionen lo mejor posible. Trataremos pues de centrarnos en la problemática de elaborar una escalera de aviones.

A la hora de realizar una FAL (Línea de Ensamblaje Final) podemos tener dos filosofías de trabajo, una FAL ligera o una FAL pesada. La ligera se caracteriza por ser más rápida y "sencilla", esto repercute en una mayor elaboración previa a la FAL de las partes que intervienen en el ensamblaje. Por lo tanto, se pretende con este tipo de política de trabajo no saturar el proceso de ensamblaje a costa de hacer más completas y consensuadas las interfaces. Por interfaces se entienden a los puntos de unión de las distintas estructuras. Tener mejor definidos los interfaces de unión requiere una mayor colaboración entre fabricantes y un mayor coste de tiempo en políticas de concurrencia. La FAL pesada da más trabajo y elaboración al proceso de ensamblaje y menos en la

elaboración. Actualmente las empresas se están decantando por la realización de procesos ligeros, este es el caso de la FAL del A400M.

Recordemos básicamente los conceptos de una FAL ligera. Tenemos las estaciones, que son las grandes operaciones a realizar y que en la realidad se traducen a recintos adecuados para las tareas propias de cada operación. La duplicación de estaciones se conoce como posiciones. Una posición es la duplicación exacta de una estación e irá en paralelo a ella. Como estaciones principales a realizar en el ensamblaje de un avión tenemos:

1. Montajes estructurales: fuselaje, ala, cola,...
2. Montajes de grandes conjuntos (grupos)
3. Equipado
4. Pruebas
5. Línea de vuelo
6. Pintura
7. Entrega

5.2. CONCEPTOS BÁSICOS DE PLANIFICACIÓN

Para elaborar una planificación tenemos que tener en cuenta los siguientes factores: tareas, recursos y calendarios. En el problema que nos incumbe las tareas serán las estaciones y operaciones, los recursos son los operarios y máquinas, y el calendario será el fijado por la empresa. Como factores adicionales a tener en cuenta nos encontramos con las curvas de aprendizaje y con la fecha de arranque del proyecto.

5.2.1. Tareas

En nuestro caso las tareas de más alto nivel son las estaciones. Estas siempre irán en serie, es decir, se sucederán temporalmente. Esta no es más que una imposición que nosotros haremos en nuestra planificación y que más tarde veremos que será importante en la aplicación. Las estaciones se pueden duplicar, triplicar, etcétera con lo que tendremos dos posiciones, tres posiciones y así respectivamente. Las estaciones se

pueden dividir en subestaciones que pueden estar en paralelo o en serie. Siempre se puede descomponer las tareas generales que forman una estación en operaciones más sencillas.

La división sería pues la siguiente:

Estación -> Tareas principales-> Hojas de operaciones-> Operaciones

Las estaciones se dividen en tareas generales que la describen y estas a su vez están compuestas de operaciones. La división de una estación en subestaciones suele hacerse cuando las tareas fundamentales que la forman tienen gran importancia por si solas y es necesario para su realización dotarlas de recursos específicos.

Un proceso visto al más alto nivel se corresponde con una sucesión de estaciones y subestaciones.

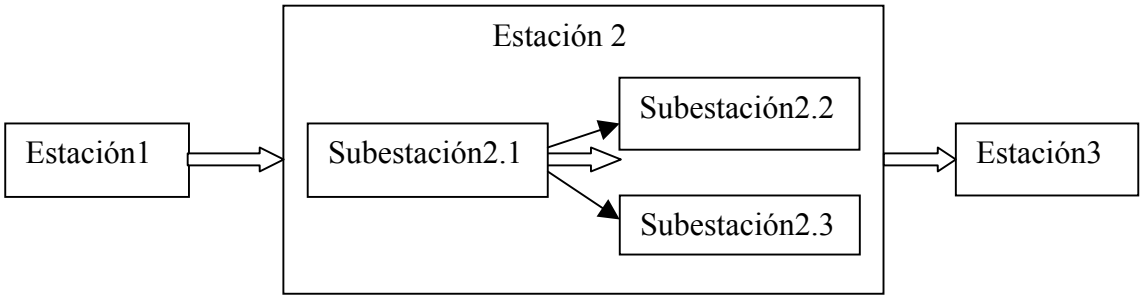


Figura 5.1. Ejemplo de estaciones y subestaciones en serie y paralelo.

Para definir las tareas tenemos:

- Horas
- Máximo número de operarios

Duración mínima del proceso

La duración de un proceso viene fijada por las horas de trabajo que implica. Existen tablas para el cálculo de tiempos de determinadas operaciones como remachado, escariado, taladrado, etcétera. La mayoría de las operaciones se ven afectadas en su duración por el número o el tipo de recursos asociados, por lo que podemos definir la duración mínima como la relación entre horas mínimas del proceso y el máximo

número de operarios. La duración mínima del proceso se conoce como **L/T** (***Lead Time***).

Supongamos que tenemos el esquema de la figura 5.2, en el que representamos el ensamblaje de los aviones frente al tiempo.

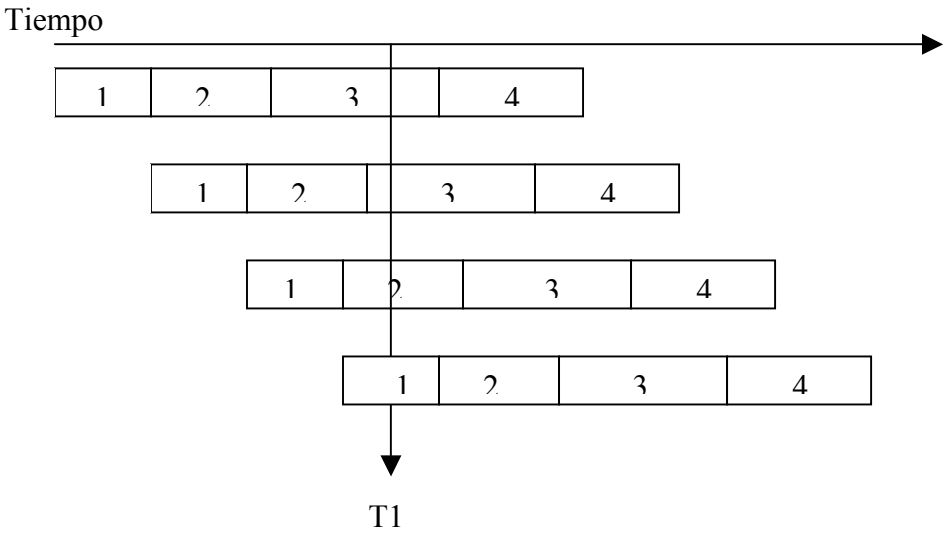


Figura 5.2. Diagrama temporal (Gantt) de sucesión de estaciones.

Las barras horizontales representan la sucesión de estaciones para la fabricación de un avión (A/C: "Aircraft"). Finalizada la estación 1 del primer avión se comienza la estación 1 del segundo avión y así para todas las estaciones y todos los aviones.

En T1 tenemos que hacer frente a dos estaciones 3 por lo que tendríamos que duplicar esa estación (figura 5.3).

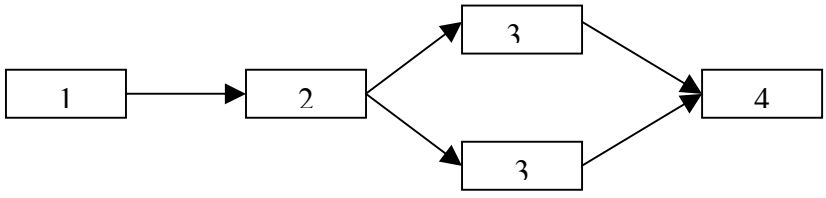


Figura 5.3. Diagrama de flechas de las estaciones.

Tenemos pues en la figura 5.2, T1 = (3(2), 2, 1). Si no se quiere o puede duplicar la estación 3 el escalonamiento de la figura 5.2 tiene que ser distinto, aumentando la separación temporal entre la finalización de los aviones.

El escalonamiento está regido por el plan de ventas, este fija los plazos de entrega y las cantidades. Las cantidades fijan lo que se conoce como plan de producción.

Si tenemos el siguiente ejemplo:

Año	1	2	3	4	5
Número de entregas	2	5	20	22	18
Entregas acumuladas	2	7	27	49	67

En el avión 28 tenemos que estar a un ritmo de 22 Aviones / año (cadencia de entrega / año). Para el cálculo de días entre entregas tenemos que tener en cuenta la disponibilidad de la fábrica, por lo que tenemos la siguiente relación:

$$\frac{\text{Número días laborales} / \text{año}}{\text{Cadencia entrega} / \text{año}} = \text{Días entre entregas} \tag{5.1}$$

5.2.2. Enlace entre estaciones

Es muy importante entender como afecta la duración de las estaciones en el enlace entre ellas y esto a su vez al plan de producción. Para ello se haremos una serie de suposiciones de las que sacaremos conclusiones.

Los enlaces entre estaciones son siempre del tipo “fin a comienzo” y a priori sin ningún retraso o adelanto. Esto se traduce en la imposibilidad de comenzar la estación x de un avión hasta que no hayamos finalizado su estación x-1 y la estación x del avión anterior. Los enlaces implican restricciones en el comienzo de las estaciones, lo que afecta a un parámetro muy importante de la planificación: el tiempo entre entregas.

1. **Estaciones iguales.**

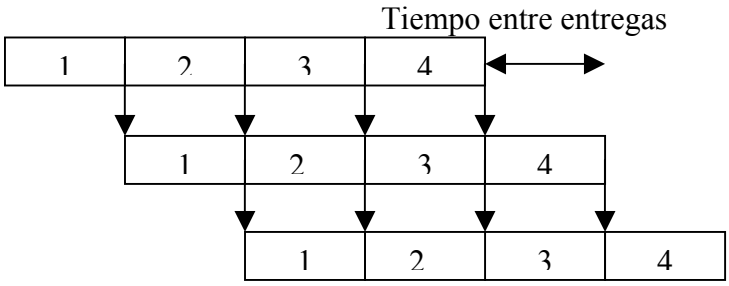


Figura 5.4. Estaciones de igual duración.

Todos los recursos están aprovechados en todo momento y la cadencia de entrega es igual al tiempo de las estaciones y constante. Es la situación ideal.

2. **Una estación más corta.**

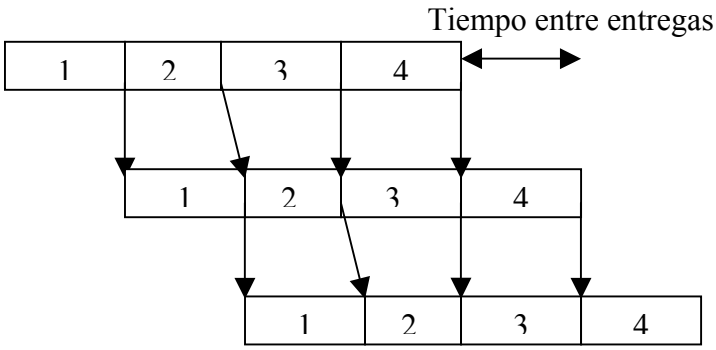


Figura 5.5. Estación intermedia más corta y el resto iguales.

Se produce un tiempo muerto en la estación más corta. Cadencia constante e igual a la del caso anterior.

3. **Una estación más larga.**

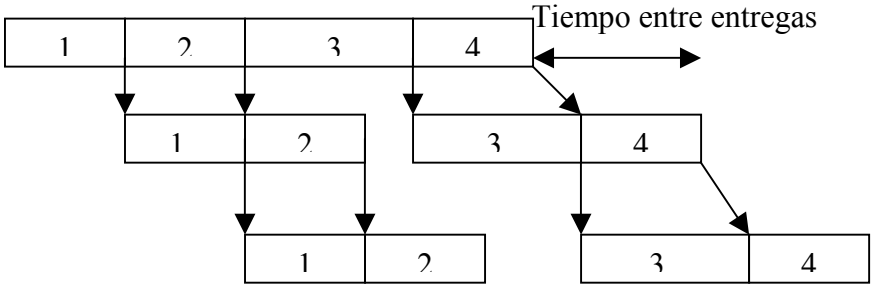


Figura 5.6. Estación intermedia más larga y el resto iguales.

Se producen dos consecuencias importantes. Tenemos un tiempo de espera para poder empezar la estación más larga y a partir de ahí las estaciones tienen tiempos muertos, esto implica recursos sin actividad con el consiguiente perjuicio. Otro factor importante es la limitación que se nos produce en la cadencia de entrega. El tiempo entre entrega mínimo es igual al tiempo de la estación más larga.

Si consideramos la duración del proceso de ensamblaje de un avión como el intervalo de tiempo que va desde el comienzo de la primera estación hasta la finalización de la última estación, observamos en la figura 5.6 que la duración de cada avión va aumentando a medida que van entrando aviones nuevos. Este efecto será de vital importancia para el desarrollo posterior de la aplicación.

Para solucionar este efecto perjudicial y que no exista ruptura en el proceso debemos retrasar el comienzo de la primera estación (figura 5.7), de esta forma ahorramos costes de almacenaje y de inmovilizado de capital

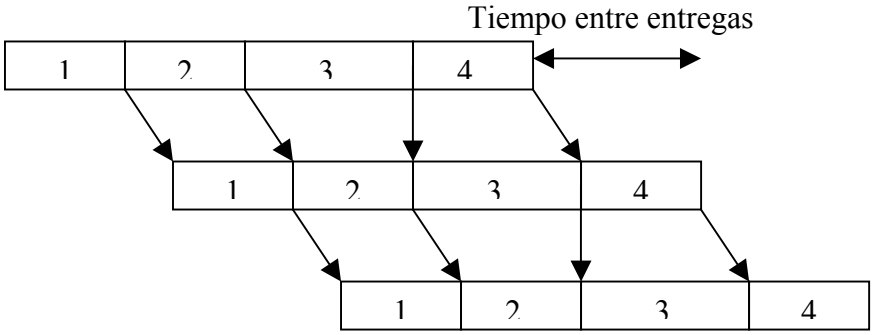


Figura 5.7. Retraso en la primeras estacion para evitar esperas.

El retraso es igual a la diferencia de tiempo entre la estación más larga y el resto de las estaciones. Se observa que la estación que no tiene retraso en su comienzo es la más larga.

4. Una estación más larga y otra más corta.

Se observa en la figura 5.8 como se producen muchos más tiempos muertos y la cadencia de entregas viene limitada por la estación más larga, en este caso la primera estación.

Es importante analizar como la posición de la estación más larga y más corta afecta a los retrasos acumulados, así se observa que estos son mayores si la estación más larga se encuentra en las primeras estaciones del proceso.

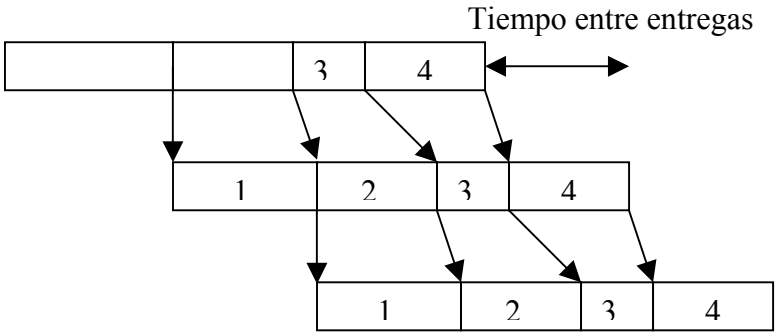


Figura 5.8. Estación más larga al principio.

Podemos concluir que la ubicación de la estación más larga tiene un efecto más significativo que el de la ubicación de la estación más corta.

5. Estaciones de distintas duraciones.

En la figura 5.9 vemos el efecto de que la estación más corta esté al comienzo.

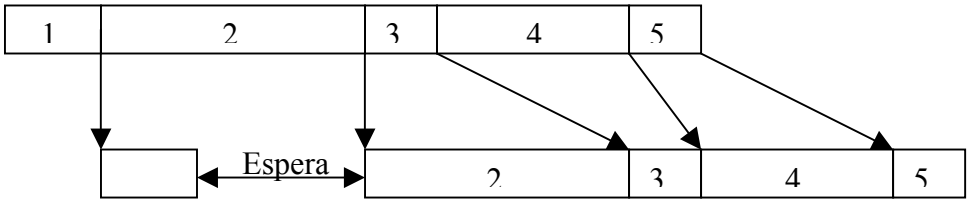


Figura 5.9. Tiempo de espera provocado por una estación más corta.

Se corrige la ruptura como se hizo anteriormente retrasando el comienzo de la primera estación

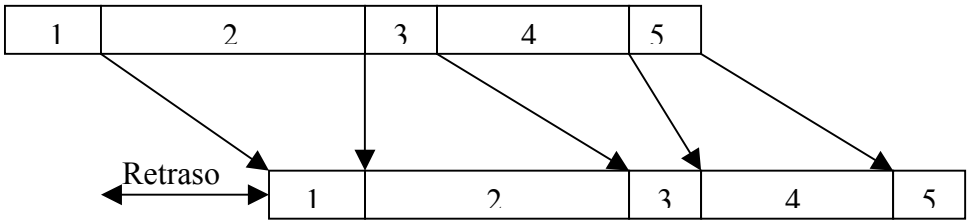


Figura 5.10. Retrasos en la primera estación para evitar esperas.

El retraso es igual a la duración de la estación 2 menos la duración de la estación1 (D2 - D1). Siendo la segunda estación la más larga.

$$\text{Retraso} = D2 - D1$$

La estación más larga es ahora la estación 4 y la segunda más larga es la 2. Por lo tanto tenemos un proceso en el que la estación más larga sucede más cerca de su finalización.

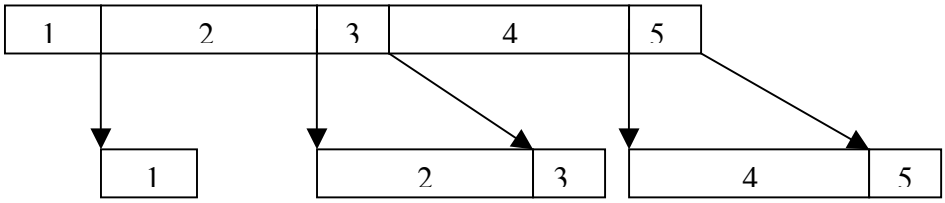


Figura 5.11. Varias zonas de espera provocadas por estaciones intermedias.

Observamos en la figura 5.11 dos puntos de ruptura. Se solucionan estas rupturas, al igual que antes, retrasando el comienzo de la primera estación hasta conseguir que el ensamblaje de un avión no tenga zonas de ruptura.

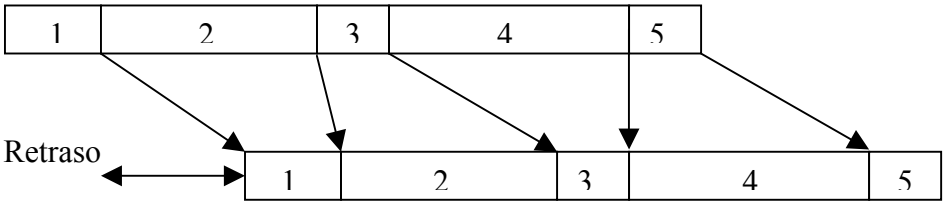


Figura 5.12. Retrasos acumulados para evitar esperas.

En este caso el retraso se obtendría de la siguiente forma:

$$\text{Retraso} = D2 + D3 + D4 - (D1 + D2 + D3) = D4 - D1$$

Obtenemos que el retraso que tenemos que aplicar en el comienzo del ensamblaje de un avión si no queremos zonas de ruptura es igual a la duración de la estación más larga menos la duración de la primera estación.

En todos los casos el tiempo entre entregas es igual al de la estación más larga.

5.2.3. Recursos

La descripción y cuantificación de los recursos utilizados es una parte fundamental en la definición de un proceso. La disponibilidad, la calidad y el número de recursos afectan de forma primordial al cálculo de las duraciones de las tareas.

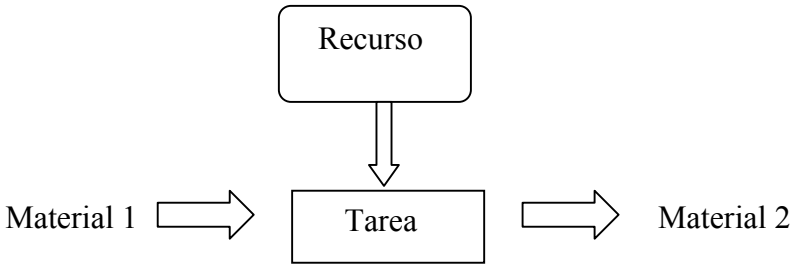


Figura 5.13. Elementos que intervienen en una operación.

La idea básica de la figura 5.2 es que cada tarea tiene asociada un conjunto de recursos. A su vez cada material tiene asociado un conjunto de tareas y por asociación un conjunto de recursos. El concepto de material es muy amplio y vago, así para nuestro caso puede ser un elemento físico como un ala o un motor, o también es válido conceptualmente el diagrama anterior considerando, por ejemplo, como material una prueba funcional.

Para una planificación de nivel global como es la escalera de aviones, los recursos principalmente son los operarios. Por tanto, los parámetros fundamentales a tener en cuenta son el número de operarios y los turnos de trabajo.

Teniendo en cuenta un horario laboral de 8 horas, podemos tener uno, dos o tres turnos por día. El número de turnos y de operarios para una operación son los factores fundamentales para reducir el tiempo de duración de una tarea. Otras formas de rebajarlo afectan al proceso en sí, como por ejemplo sería la utilización de maquinaria que redujera el tiempo y el número de operaciones manuales. Estas son otro tipo de consideraciones que no vamos a tratar, sólo indicar que suelen ser cuestiones de grandes análisis por parte de las empresas y que intervienen otros factores de calificación como son las inversiones, amortizaciones y rentabilidad en las compras de la maquinaria y utillaje en general.

5.2.4. Calendario

El calendario es fundamental en cualquier planificación. Hay muchos factores que hay que tener en cuenta a la hora de realizar un calendario laboral.

Los días festivos son un primer factor. Estos pueden ser nacionales, regionales (autonómicos) y / o locales. Estos días, por lo general, resultan de convenios entre la empresa y los sindicatos. Tenemos además las vacaciones, en las que tenemos connotaciones de tipo legal, los días libres y los días no laborales. Todos estos días suelen depender del convenio fijado en cada empresa.

Como resultado de la diferencia entre los 365 días, excepto años bisiestos, y los días festivos tenemos la jornada anual por trabajador. Por ejemplo, en EADS CASA está fijada para el año 2002 en 215 días con lo cual y considerando 8 horas de trabajo diario por trabajador se fija una jornada anual de 1719 horas / año.

Otro concepto a tener en cuenta es el de días con la factoría abierta. Esto implica rotaciones en el trabajo, más contrataciones, más turnos, etcétera. Como resultado tenemos un número distinto al de la jornada anual, en el caso de EADS CASA este número se fijó en 220 días con la factoría abierta.

La cadencia de salida al año se obtiene pues de la siguiente forma:

Cadencia de salida = aviones / Días con la factoría abierta.

En un proyecto cuya duración se prevé para muchos años como es el del A400M, el calendario total del proyecto es difícil de obtener. Es obvio que es imposible conocer a priori los convenios que tendremos en el futuro e inclusive los festivos fijados por el estado. Por lo tanto, es importante realizar una planificación que no dependa en exceso de este tipo de factores, o que sea fácilmente modificable ante cualquier cambio.

Otro punto fundamental en un proyecto donde intervienen distintos países y empresas como este es el de la compatibilidad de calendarios.

Una vez realizada la planificación del proyecto esta es sensible a posibles modificaciones, la principal es la de la fecha de arranque del mismo. El punto de partida T_0 es fundamental, puesto que cualquier atraso o adelanto en él, repercute en toda la planificación. Por lo que la planificación se suele hacer en función de esta fecha, generalmente sin especificar, y es una vez fijada cuando los días se marcan en un calendario concreto.

5.2.5. Curvas de aprendizaje

Este es un parámetro muy importante en la planificación de una línea de ensamblaje con muchos procesos manuales como es el caso del ensamblaje de un avión. El concepto de curva de aprendizaje tiene una explicación muy intuitiva. Los trabajos manuales o que requieren de intervención humana no tienen siempre la misma velocidad de ejecución. Su tiempo de ejecución depende de factores como la fatiga, la concentración, conocimientos previos, instalaciones adecuadas, etcétera. En nuestro caso el factor que regulan este tipo de curvas es el aprendizaje, es decir, hacemos más rápido las cosas cuando estamos más habituados a hacerlas y sabemos mejor cómo se hacen. Esto se ve reflejado en que se tarda menos en ensamblar lo últimos aviones que los primeros, puesto que los operarios conocen mejor las tareas a realizar.

Las curvas de aprendizaje siguen la siguiente ecuación y gráfica.:

$$R_x = R_n \left(\frac{x}{n}\right)^{\frac{\lg(c) - 2}{0,301030}} \tag{5.2}$$

R_x = valor de la repetición x que se quiere calcular

R_n = valor de la repetición n asociada (dato conocido)

x = Número de la repetición del valor que se quiere calcular (R_x)

n = número de la repetición de dato conocido (R_n)

c = curva de mejora, en porcentaje (%)

Ejemplo de curva del 85%

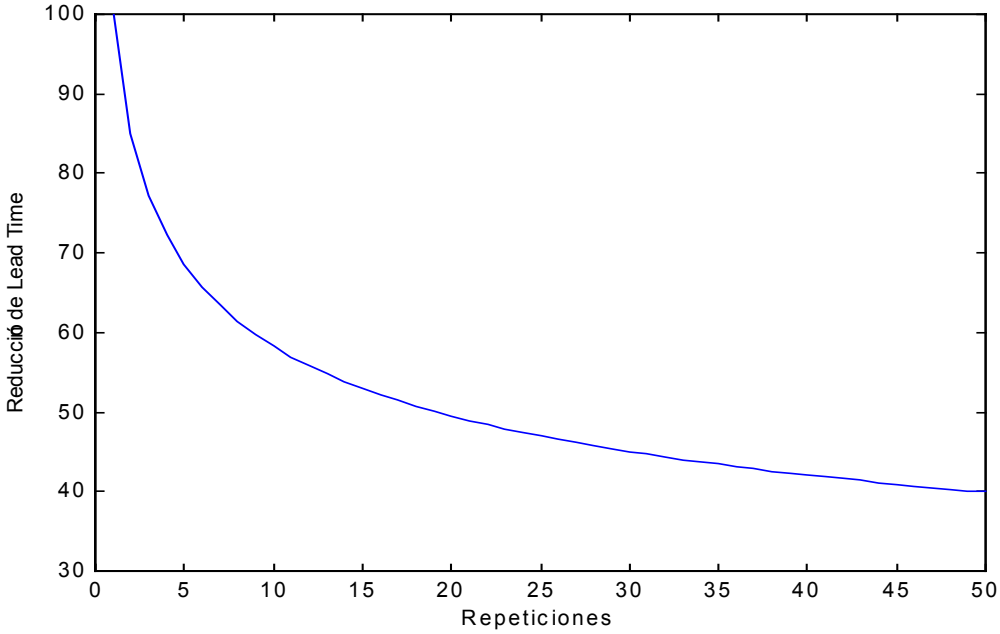


Figura 5.14. Curva de aprendizaje con factor de mejora del 85%.

Para la obtención de esta curva los valores de la ecuación 5.2 son los siguientes:

R_x = valores del eje Y de la gráfica

$R_n = 100$

x = valores del eje X de la gráfica

n = 1

c = 85

Las curvas de aprendizaje están tabuladas. Evidentemente, la reducción de tiempo depende del tipo de tarea, no es la misma la reducción de tiempo por aprendizaje en un taladrado que un escariado o en la aplicación de pintura a una chapa. La operaciones realizadas por maquinaria sin intervención humana no tienen reducción de tiempo, puesto que no necesitan aprender ni se cansan, realizarán la misma tarea en el mismo tiempo la primera vez y la última.

Cuando no queremos aplicar reducción de tiempo se le aplica un factor a la curva (5.2) del 100%, basta observar la ecuación para determinar que no tenemos reducción, este es el caso por ejemplo de una tarea totalmente automatizada. Por otro lado nunca se puede tener un factor del 0% puesto que tenemos que aplicarle un logaritmo al factor de corrección de tiempos. Cuanto más cercano a 100% sea el factor menor reducción de tiempo tendremos, igualmente cuando menos sea el factor de corrección mayor reducción de tiempo se obtendrá.

Volviendo a los diagramas utilizados para explicar las relaciones entre estaciones, observamos el efecto provocado por la reducción de tiempos por aprendizaje

Si observamos el caso ideal en el que todas las estaciones son iguales (figura 5.4) y la reducción es igual en todas las estaciones tenemos la figura 5.15.

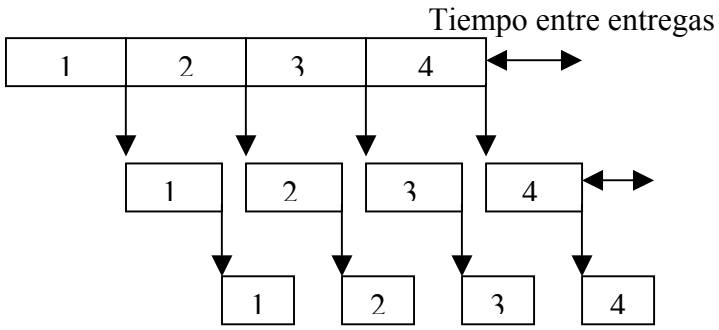


Figura 5.15. Efectos de la reducción de tiempos en los enlaces entre estaciones.

Observamos que si comenzamos las estaciones justo cuando terminan las del avión precedente tenemos tiempos muertos en el ensamblaje de un avión, es decir, un avión se podría ensamblar en un tiempo menor de lo que se hace.

Como los trabajos de una estación para un avión no pueden comenzar mientras no terminen los trabajos en esa estación con el avión anterior en la secuencia de montaje, a menos que tengamos duplicada esa estación, es obvio que la única solución es retrasar el comienzo de las estaciones para no “trocear” los aviones.

Otra posible solución es la de intentar conseguir la situación óptima, que no es otra que la de tener todas las estaciones iguales. Para ello lo que hacemos es una distribución

de los recursos, de forma que, por ejemplo, cuando la reducción de tiempo sea significativa en una estación a esa estación se le asignen menos recursos.

En ambas situaciones podemos observar que el tiempo entre entregas va disminuyendo con el tiempo debido a la reducción por aplicación de curvas de aprendizaje, esto es lo que proporciona al diagrama de Gantt resultante su forma de escalera.

Es importante indicar que es conveniente realizar el estudio del proceso de ensamblaje, desde el punto de vista de ingeniería del proceso, buscando tener estaciones con *Lead Time* (L/T) iguales o en su defecto parecidos.

Estudiemos con un ejemplo de como afectan las curvas de aprendizaje a la cadencia de entrega de aviones y en definitiva, al plan de producción.

ESTACIONES	CURVA	Nº POSICIONES	Nº TURNOS	L/T(DÍAS)
Estación 1	85 %	1	1	10
Estación 2	85 %	1	1	30
Estación 3	85 %	1	1	20
Estación 4	85 %	1	1	10
Estación 5	85 %	1	1	10
Estación 6	85 %	1	1	20

Si analizamos este supuesto sin tener en cuenta la reducción de tiempos por curva de aprendizaje tendríamos la siguiente cadencia de entrega:

Estación con Lead Time máximo : Estación 2 = 30 días
Días con la factoría abierta = 220 días
Cadencia de entrega = 220/30 = 7 aviones /año

La cadencia de entrega es constante porque esta viene fijada por la duración de la estación más larga, valor que no se modifica en toda la escalera.

Si tenemos en cuenta la curva de aprendizaje los resultados son:

Aviones	L/T máximo	Subtotal días	Aviones	L/T máximo	Subtotal días
A/C 1	30	30	A/C 7	19	168,14
A/C 2	25,5	55,5	A/C 8	18,42	186,56
A/C 3	23,19	87,69	A/C 9	17,92	204,48
A/C 4	21,17	108,86	A/C 10	17,48	221,96
A/C 5	20,57	129,43	A/C 11	17,1	239,06
A/C 6	19,71	149,14	A/C 12	16,75	255,81

En este caso tenemos una cadencia en el primer año de 10 aviones / año. Cadencia variable a lo largo de los años.

Otro aspecto que hay que tener en cuenta es lo que se conoce como “fin de curva”. Observamos que la curva es una exponencial, por lo tanto asintótica. Generalmente la curva se ajusta bien a la realidad en las primeras repeticiones, pero llega un momento en el que no existen reducciones significativas por lo que llegado ese punto se deja de aplicar la curva de aprendizaje. Existe bibliografía que basada en la experiencia recogen las tabulaciones aplicables a las curvas en función de las tareas típicas en procesos industriales.

5.2.6. Estación crítica

En los puntos anteriores hemos visto el efecto que tiene la estación de mayor duración en el tiempo entre entregas, y como su posición afecta al comienzo de la primera estación si no queremos efectos indeseables como son tiempos muertos y “huecos” entre estaciones.

La estación de mayor duración se conoce como estación crítica. Es fundamental para una buena planificación identificar la estación crítica de un proceso. La estación crítica no tiene porque ser la de L/T máximo, tenemos que tener en cuenta otros parámetros como son el número de posiciones y el de turnos de operarios. Una estación con operaciones de mucha duración que en principio es la crítica de un proceso puede dejar

de serlo si la duplicamos o reducimos su tiempo a base de aumentar el número de operarios o turnos de trabajo.

Una forma de identificar la crítica es "ver" que estación provoca un cuello de botella en el proceso, esto se traduce en una limitación en el tiempo entre entregas. Existen multitud de algoritmos (CPM, PERT,...) que detectan los caminos o rutas críticas de un proceso. Siempre tendremos una estación o ruta crítica, porque aunque se consiga que una estación crítica deje de serlo mediante el aumento de posiciones, existirá una segunda más larga y así sucesivamente.

El criterio a seguir en la reducción de tiempo de una estación crítica depende de muchos factores, por ejemplo, podemos tener en cuenta criterios económicos para elegir entre aumentos de turnos (aumento de sueldos, contrataciones,...) o aumento de posiciones (compra de terrenos, nuevos hangares,...) o tener en cuenta criterios de utillaje, etcétera.

Ejemplo:

Duración de un avión	**Posiciones**	**Turnos**
20,4	1	1
10,2	1	2
6,8	1	3
5,1	2	2

Podemos observar la disminución de tiempo que se consigue en el ensamblaje de un avión mediante un aumento de turnos de operarios y de posiciones de la estación crítica. Como ya se ha comentado, en la opción a elegir influyen diversos criterios de la gestión de proyectos como son los propios de planificación, económicos, imposiciones externas, etcétera.

5.2. PARTICULARIDADES DE LA PLANIFICACIÓN AERONÁUTICA

Lo que se pretende con este apartado es conocer la problemática especifica que se nos plantea en la realización de una escalera de aviones. Tenemos que tener en cuenta los factores particulares y propios del hecho de ser una planificación del ensamblaje de aviones.

5.3.1. Aviones especiales

Denominamos aviones especiales a los aviones que tienen operaciones distintas al resto de aviones. Estos aviones suelen ser los primeros aviones y los denominados aviones de refurbishing.[1]

Los primeros aviones en la escalera son los más complicados puesto que son los que más ensayos necesitan y por supuesto, al ser los primeros podemos encontrarnos con problemas no previstos. Los operarios se enfrentan a tareas que realizan por primera vez y necesitan recurrir con mucha frecuencia a las ordenes y planos de trabajo. Son aviones que tienen un número de estaciones superiores a los normales. Por normales, nos referimos a los aviones que se ensamblan cuando el proceso está ya lanzado. Existen ensayos y certificaciones que se realizan en estos primeros aviones. Un intervalo típico de aviones que consideramos especiales por ser los primeros es de 7 aviones. Estos aviones especiales no se entregan de inmediato sino que se utilizan para la obtención de certificados y realización de pruebas (figura 5.16).

Los aviones de refurbishing son los primeros aviones fabricados (no entregados a los clientes) una vez que vuelven a la cadena de ensamblaje para su acondicionamiento y entrega al cliente. Como es obvio no necesitan de estaciones como la de ensamblaje de las principales estructuras o equipamiento de grandes sistemas, pero sí de otras como la de decoración interior, pruebas funcionales o pintura y claro está, entrega al cliente. El

[1] El término refurbishing proviene del verbo inglés *refurbish*: renovar, hacer reformas en.

hecho de que sean aviones ya fabricados y por lo tanto con un tiempo menor de "ensamblaje" nos da la posibilidad de tener un margen de maniobra para la planificación de los años con mayor demanda de aviones.

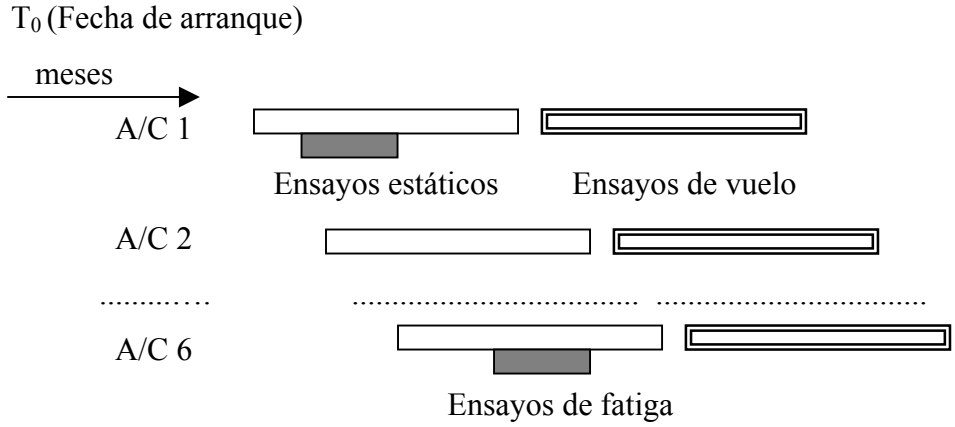


Figura 5.16. Esquema típico de operaciones en los primeros aviones.

5.3.2. Avión de régimen

Es muy importante para la planificación y para el departamento de Ingeniería de Desarrollo el concepto de avión de régimen. Se alcanza el avión de régimen cuando tenemos la máxima cadencia de entrega de aviones y a partir de ese momento esta es constante. Esto implica que hemos alcanzado los tiempos mínimos en la realización de operaciones y que las curvas de aprendizaje han dejado de tener validez.

Un dato muy importante en el estudio de la viabilidad del proyecto es el del número de entrega de aviones al año, para cumplir con el plan de entregas fijado ya por el departamento de ventas. Por lo tanto, es importante tener una cadencia máxima de entrega lo antes posible y constante. Es la duración del ensamblaje (*Lead Time* del avión) de este avión la que nos va a fijar esta cadencia. Un valor típico es alcanzar el avión de régimen en la repetición 100.

Llegado este punto es conveniente aclarar un concepto que es importante, y que es fácil que provoque confusión. No es lo mismo el avión 100 que la repetición 100. Cuando se habla de repetición es para indicar una entrada en la curva de aprendizaje, es

la variable "x" de la curva, sin embargo, no tiene porqué coincidir con el número de avión que se ensambla, ni con su posición en la escalera de aviones. Es muy típico no empezar a aplicar curvas de aprendizaje hasta que el proceso esté ya lanzado, puesto que los primeros aviones son especiales, y por lo tanto con operaciones distintas al resto, por lo que no se les aplica curva de reducción. A estos aviones no se los considera como repeticiones, por tanto si suponemos que el avión 7 es el primero que forma parte normalmente de la escalera, este sería la primera repetición para entrar en la ecuación de reducción de tiempos (5.2).

5.3.3. Avión de referencia

La realización de la escalera se hace a partir de datos, proporcionados por el departamento de ingeniería de desarrollo, del ensamblaje de un avión. Existen datos tabulados, que se encuentran en una base de datos de tiempo, de duración de las operaciones básicas. Es decir, están tabulados los tiempos en realizar un taladrado, un escariado, lijar, etcétera y también en función de la dificultad y utillaje que requiere cada operación. En un primer momento, el proceso no está definido, es más el trabajo es definir el proceso, por lo que no se pueden tener esos datos. Así intervienen otros factores como la experiencia, buscar información en procesos similares, etcétera. Como conclusión de un primer planteamiento en la realización del proceso se obtienen unos tiempos que se cree se asemejen a la realidad. Estos tiempos se obtienen para las distintas estaciones y para un avión de referencia. Apropiadamente tendríamos que decir repetición de referencia puesto que para el cálculo de tiempos hay que tener en cuenta el grado de aprendizaje de los operarios.

El dato de repetición de referencia es la entrada "n" en la fórmula de reducción de tiempos por aprendizaje (5.2). Los datos de tiempo proporcionados por ingeniería para el avión o repetición de referencia son la base para la construcción de la planificación.

Un valor típico para el que se obtienen estos tiempos es para el avión 10, en este caso no es repetición 10 por lo que si queremos introducirlo en la curva, y suponiendo que el avión 7 es la primera repetición, tendríamos los datos para la repetición 3.

6. PLANIFICACIÓN DE LA FAL CON MICROSOFT PROJECT

6.1. INTRODUCCIÓN

El objetivo de este capítulo es aprender cómo se hace la planificación de la línea de montaje final utilizando la herramienta software de planificación Microsoft Project 98. Se realiza con Microsoft Project 98[1] porque es el programa de planificación usado en EADS CASA y es el utilizado en el desarrollo de la aplicación PlanFal.[2]

Este proyecto surge ante la necesidad de mejorar un proceso costoso en tiempo. Es pues importante, conocer las limitaciones que nos encontramos a la hora de realizar una escalera de aviones usando Microsoft Project, limitaciones que posteriormente se intentarán solventar con el desarrollo de la aplicación PlanFal, que automatiza el proceso haciendo uso de la capacidad que tiene Project de admitir módulos de programación.

El capítulo se enfoca partiendo de los conceptos básicos de Project (ver anexo I), relacionandolos con los conceptos ya vistos en el capítulo V referentes a la planificación y aeronáutica. En el desarrollo del mismo se profundizará en aquellos aspectos de Project que son importantes para las necesidades concretas a la hora de planificar la línea de montaje final. Es necesario cierto conocimiento básico de Project, por lo que sería importante leer el manual básico que se incluye en la memoria como anexo. En

[1] Habitualmente nos referiremos a Microsoft Project 98 como Project.

[2] PlanFal es el nombre de la aplicación de planificación desarrollada en este proyecto.

todo caso, para profundizar en aquellos aspectos que no quedasen claros, recomiendo la utilización de la ayuda que ofrece Project.

Project dispone de distintos interfaces para presentar e introducir información, que se denominan vistas. La vista que se utilizará habitualmente es el diagrama de Gantt, ya que como se dijo en el capítulo II, una escalera de aviones es un diagrama de Gantt de la planificación de la FAL a alto nivel, en el que los resultados que se quieren obtener son el comienzo y fin del proceso de ensamblaje de los aviones. El diagrama de Gantt es además un diagrama muy intuitivo y fácil de manipular. Por todas estas razones es la vista más idónea para trabajar en Project.

Evidentemente, la labor de planificar requiere previamente de unos datos para trabajar. Estos datos son aportados por el departamento de ingeniería que estudia el proceso de ensamblaje, este departamento a su vez, necesita los datos aportados por la planificación para validar los procesos definidos.

Para la realización de la escalera de aviones los principales datos que se necesitan son: las operaciones básicas para el ensamblaje del avión, las estimaciones de la duración de las operaciones, la secuenciación de operaciones y el avión o repetición de referencia para el que se estimaron las duraciones.

Datos como los recursos asociados a las distintas operaciones, que para otros cálculos (por ejemplo costes) son muy importantes, en principio no se necesitan para visualizar el diagrama de Gantt, puesto que este viene fijado por las duraciones. El número de operarios influye en el cálculo de la duración de una operación, pero es la duración en sí la que determina las fechas de comienzo y fin de las operaciones y en consecuencia del ensamblaje del avión.

6.2. CALENDARIO

Como primer paso para definir un proyecto debemos fijar su fecha de comienzo. En el caso del programa del A400M, el ensamblaje es una fase más, concretamente la penúltima, del proyecto global. Fases del proyecto, como por ejemplo la fase de diseño,

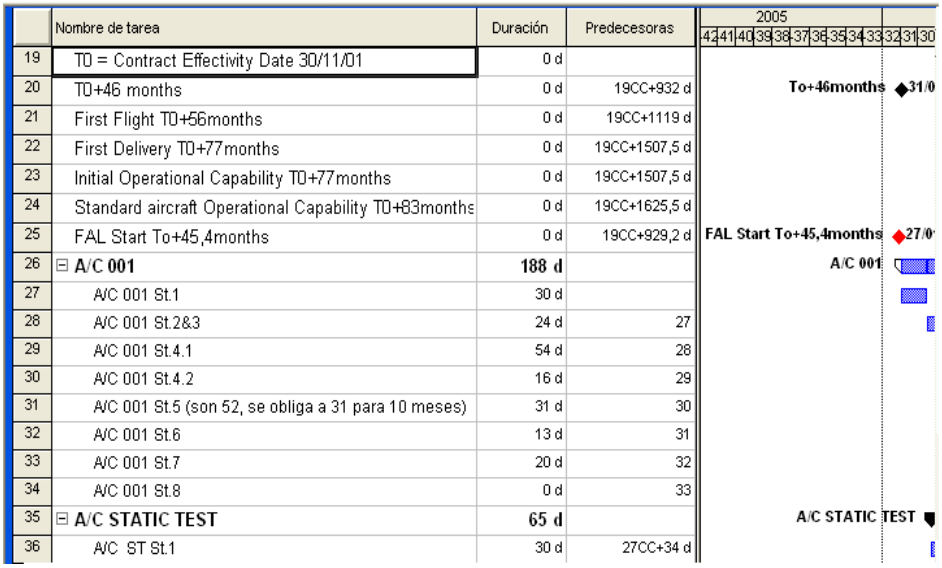
tienen necesariamente una fecha de comienzo anterior a la del ensamblaje. Esto conlleva a que el comienzo del ensamblaje esté condicionado por muchos factores, como son la finalización de fases de estudios previos, fases de diseño, cuestiones comerciales, firma de convenios, compras de terrenos, disposición de capital y recursos, y un largo etcétera. Es por ello por lo que no se puede hablar de una fecha establecida y fija de comienzo, sino que se habla de una fecha $\mathbf{T_o}$ de comienzo del proyecto. $\mathbf{T_o}$ se considera el punto de arranque del proyecto y afecta al cálculo de fechas del resto del proyecto.

Otro factor a tener en cuenta antes de comenzar con la planificación de las tareas es la definición de una calendario laboral. Como se comentó en el capítulo V el calendario laboral depende de cuestiones ajenas a la planificación y al proyecto. Existen otras limitaciones importantes para fijar el calendario, como es la imposibilidad de conocer el calendario a unos años vistas, puesto que no están firmados los convenios que lo regulan.

Al comenzar un proyecto debemos indicar una fecha de comienzo, así como el tipo de calendario que vamos a utilizar (ver anexo I). El calendario lo podemos modificar a través de la opción **cambiar calendario laboral.** Básicamente fijamos las horas de trabajo y los días festivos y laborables.

Figura 6.1. Modificaciones de Calendario en Project.

Es importante realizar la planificación de forma que posibles modificaciones en $\mathbf{T_o}$ afecten al resto de las tareas, puesto que un adelanto o retraso (posibilidad bastante factible) de $\mathbf{T_o}$ implica un adelanto o retraso del proyecto. Es por ello que las tareas deben estar vinculadas (ver anexo I) de forma que al menos el comienzo de la primera tarea este vinculado al hito[1] $\mathbf{T_o}$.



	Nombre de tarea	Duración	Predecesoras
19	T0 = Contract Effectivity Date 30/11/01	0 d	
20	T0+46 months	0 d	19CC+932 d
21	First Flight T0+56months	0 d	19CC+1119 d
22	First Delivery T0+77months	0 d	19CC+1507,5 d
23	Initial Operational Capability T0+77months	0 d	19CC+1507,5 d
24	Standard aircraft Operational Capability T0+83months	0 d	19CC+1625,5 d
25	FAL Start To+45,4months	0 d	19CC+929,2 d
26	**A/C 001**	**188 d**	
27	A/C 001 St.1	30 d	
28	A/C 001 St.2&3	24 d	27
29	A/C 001 St.4.1	54 d	28
30	A/C 001 St.4.2	16 d	29
31	A/C 001 St.5 (son 52, se obliga a 31 para 10 meses)	31 d	30
32	A/C 001 St.6	13 d	31
33	A/C 001 St.7	20 d	32
34	A/C 001 St.8	0 d	33
35	**A/C STATIC TEST**	**65 d**	
36	A/C ST St.1	30 d	27CC+34 d

Figura 6.2. Ejemplo de planificación basada en Fecha de arranque.

6.3. TAREAS

Fijado el calendario, el siguiente paso es introducir las tareas. Microsoft Project es una aplicación desarrollada bajo programación orientada a objetos. Las tareas son tratadas como objetos y sus características como propiedades de ese objeto. Los campos más importantes que podemos rellenar en una tarea son: Nombre, Precedencia, Duración, Fecha de comienzo y Fecha de Finalización. No todos los campos son independientes, por ejemplo, la fecha de finalización de una tarea viene fijada por la fecha de comienzo, la duración y el calendario. Si Project encuentra incongruencias en los datos introducidos, por ejemplo, introducción de una fecha anterior al comienzo del

[1] Un hito es una tarea de duración cero, cuya función es señalar momentos importantes o fechas que hay que tener en cuenta en el desarrollo del proyecto.

proyecto, nos advierte de ello y ofrece una alternativa, aunque la mayoría de las veces permite continuar con el trabajo.

Como ya se comentó, otra característica de las tareas es la posibilidad de jerarquizarlas (ver figura 6.3). Esto permite agrupar tareas en otra más detalladas, las tareas así definidas adquieren sus propiedades (fechas, duraciones,...) de las tareas de jerarquía inferior que la forman. En el caso de una escalera de aviones la tarea de nivel jerárquico superior es un avión, para la realización de un avión tenemos una sucesión de estaciones (ver Capítulo V), que a su vez pueden estar formadas por subestaciones.

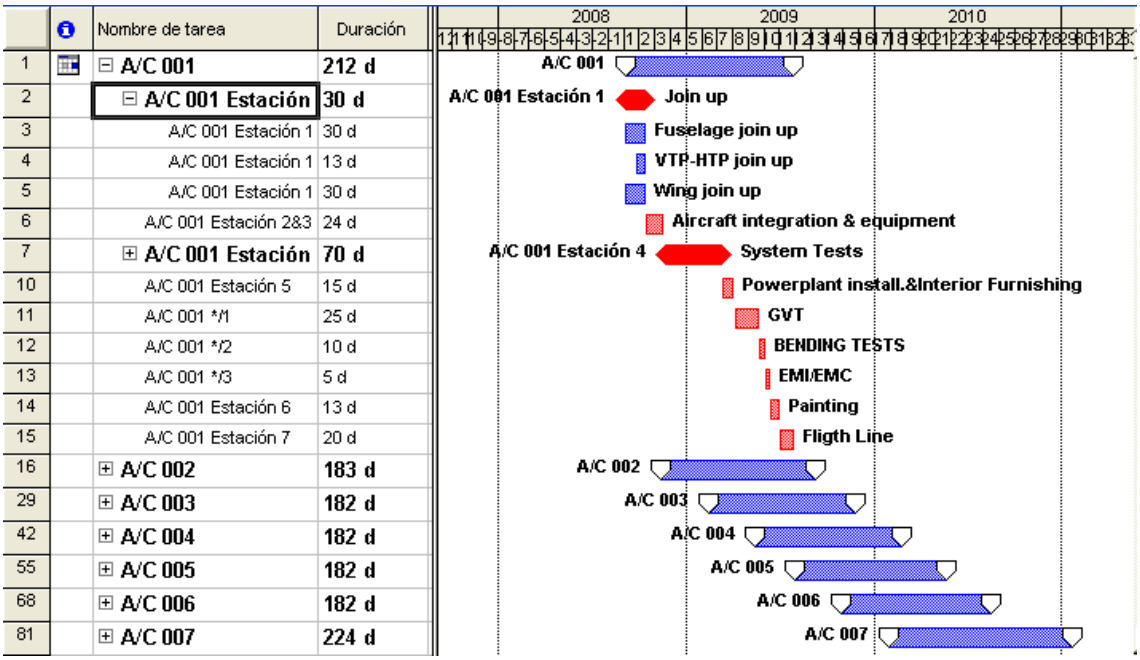


Figura 6.3. Tareas (Aviones, Estaciones, Subestaciones) y diagrama de Gantt correspondiente.

6.3.1. Duraciones de las Tareas

Sólo se necesitan las estimaciones de duración de las tareas de nivel jerárquico inferior, en nuestro caso, la duración de las subestaciones o de las estaciones que no tengan subestaciones. Las duraciones de los aviones son calculadas automáticamente por Project en función de la duración de las subtareas y del tipo de vinculaciones existentes entre las tareas.

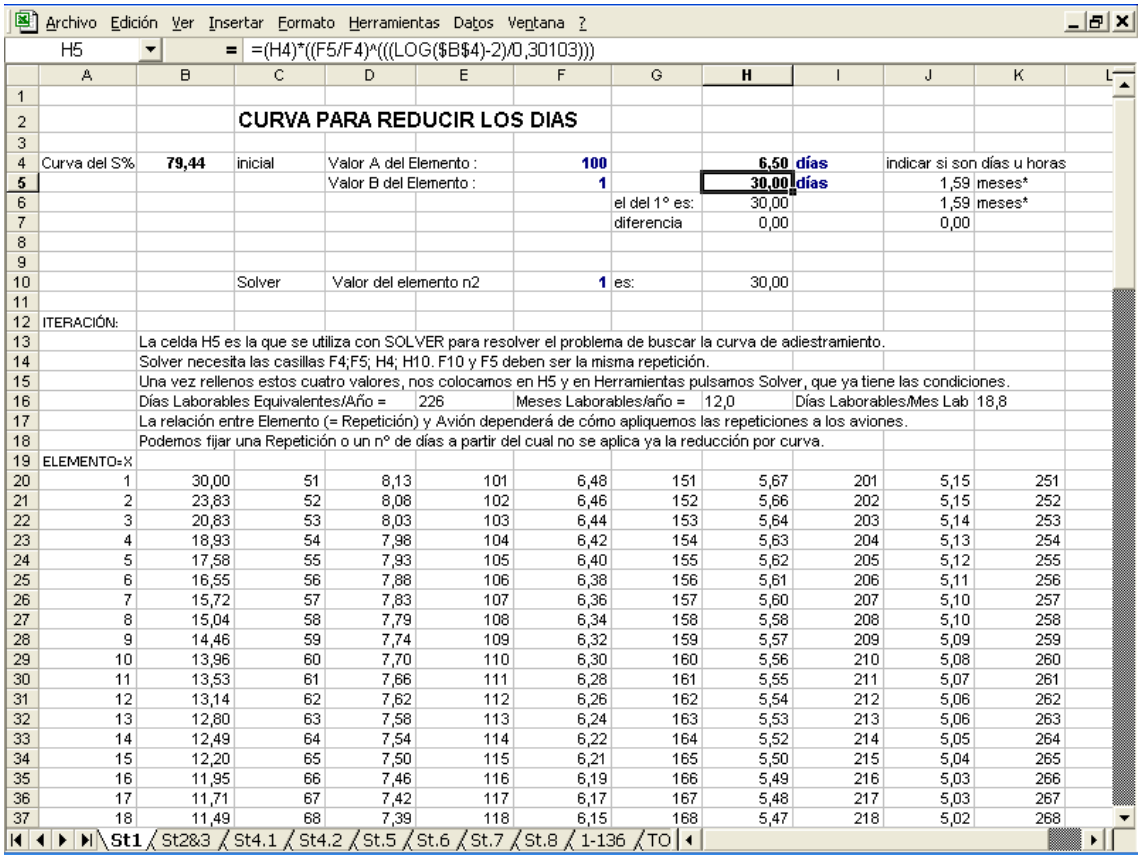
Para el cálculo de duraciones tenemos que tener en cuenta la reducción de tiempos por curva de aprendizaje (capitulo V). Entre los datos aportados por el Departamento de Desarrollo están las estimaciones de las duraciones de las operaciones para un avión (avión de referencia), por lo que es necesario calcular las duraciones para el resto de los aviones teniendo en cuenta la curva de reducción de tiempos (5.2).

Si utilizamos Project en su forma estándar (sin macros ni añadidos adicionales), no podemos calcular las variaciones de duraciones para los distintos aviones. Se hace necesario, pues, el uso de herramientas software adecuadas que sí permiten el uso de fórmulas matemáticas, por ejemplo las hojas de cálculo. En EADS CASA el paquete de ofimática que se usa es Microsoft Office. Dentro del paquete Office, la hoja de cálculo es la aplicación Microsoft Excel. La utilización de Excel favorece las posibilidades de importación-exportación con Project al ser ambas aplicaciones desarrolladas por Microsoft.

6.3.1.1. Cálculo de duraciones con Excel

A continuación vamos a explicar una forma de obtener los valores de la curva de aprendizaje utilizando **Excel**. Se obtendrán los valores para distintas repeticiones y para todas las estaciones, por lo que tenderemos una hoja de cálculo para cada estación. En ocasiones puede ser interesante obtener el factor de mejora que se ha aplicado en la curva, a partir de dos valores de la curva. Podemos tener este caso cuando se estima en función de la experiencia la reducción de tiempos de una tarea para dos repeticiones distintas. Para obtener el factor de mejora se propone el uso de **Solver**, una herramienta adicional de Excel que permite maximizar o minimizar una función objetivo, cuyas variables están sujetas a restricciones haciendo uso de la programación lineal.

Podemos observar en la figura 6.4 que tenemos una hoja por cada estación. En este caso estamos observando la estación 1 (**St1**). Los elementos son las distintas repeticiones de la estación 1 para los que se calcula su duración. Para el cálculo de la duración se ha copiado la formula que aparece en la figura 6.4, cambiando para cada celda el número de repetición, para ello se ha utilizado la opción **Pegado especial** de fórmulas. Esta opción reconoce de forma automática los cambios de celda, evitándonos rescribirlas.



Archivo Edición Ver Insertar Formato Herramientas Datos Ventana ?

H5 = =(H4)*((F5/F4)^(((LOG(B4)-2)/0,30103)))

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2				CURVA PARA REDUCIR LOS DIAS								
3												
4	Curva del S%	79,44	inicial	Valor A del Elemento :		100		6,50	días	indicar si son días u horas		
5				Valor B del Elemento :		1		30,00	días	1,59	meses*	
6							el del 1° es:	30,00		1,59	meses*	
7							diferencia	0,00		0,00		
8												
9												
10			Solver	Valor del elemento n2		1	es:	30,00				
11												
12	ITERACIÓN:											
13		La celda H5 es la que se utiliza con SOLVER para resolver el problema de buscar la curva de adiestramiento.										
14		Solver necesita las casillas F4;F5; H4; H10. F10 y F5 deben ser la misma repetición.										
15		Una vez rellenos estos cuatro valores, nos colocamos en H5 y en Herramientas pulsamos Solver, que ya tiene las condiciones.										
16		Días Laborables Equivalentes/Año =		226		Meses Laborables/año =		12,0		Días Laborables/Mes Lab		18,8
17		La relación entre Elemento (= Repetición) y Avión dependerá de cómo apliquemos las repeticiones a los aviones.										
18		Podemos fijar una Repetición o un n° de días a partir del cual no se aplica ya la reducción por curva.										
19	ELEMENTO=X											
20	1	30,00	51	8,13	101	6,48	151	5,67	201	5,15	251	
21	2	23,83	52	8,08	102	6,46	152	5,66	202	5,15	252	
22	3	20,83	53	8,03	103	6,44	153	5,64	203	5,14	253	
23	4	18,93	54	7,98	104	6,42	154	5,63	204	5,13	254	
24	5	17,58	55	7,93	105	6,40	155	5,62	205	5,12	255	
25	6	16,55	56	7,88	106	6,38	156	5,61	206	5,11	256	
26	7	15,72	57	7,83	107	6,36	157	5,60	207	5,10	257	
27	8	15,04	58	7,79	108	6,34	158	5,58	208	5,10	258	
28	9	14,46	59	7,74	109	6,32	159	5,57	209	5,09	259	
29	10	13,96	60	7,70	110	6,30	160	5,56	210	5,08	260	
30	11	13,53	61	7,66	111	6,28	161	5,55	211	5,07	261	
31	12	13,14	62	7,62	112	6,26	162	5,54	212	5,06	262	
32	13	12,80	63	7,58	113	6,24	163	5,53	213	5,06	263	
33	14	12,49	64	7,54	114	6,22	164	5,52	214	5,05	264	
34	15	12,20	65	7,50	115	6,21	165	5,50	215	5,04	265	
35	16	11,95	66	7,46	116	6,19	166	5,49	216	5,03	266	
36	17	11,71	67	7,42	117	6,17	167	5,48	217	5,03	267	
37	18	11,49	68	7,39	118	6,15	168	5,47	218	5,02	268	

St1 St2&3 St4.1 St4.2 St.5 St.6 St.7 St.8 1-136 TO

Figura 6.4. Cálculo de reducción de días con Excel.

Los datos que necesitamos para el cálculo de duraciones son: un número de repetición (celda F4), el valor para esa repetición (celda H4) y el factor de la curva (celda B4). Si no tenemos un valor para el factor de la curva pero se dispone de dos valores de la curva, hacemos uso de **Solver** para calcular el factor de reducción que hay que aplicar. Para ello se tiene que introducir los datos de repetición conocidos en las celdas (F4 y F5) y las duraciones respectivas en las celdas (H4 y H10), para que los datos sean coherentes F10 y F5 deben tener el mismo valor, puesto que ambos representan la repetición de referencia. A continuación nos colocamos sobre la celda H5 y pulsamos **Solver** (en el menú **Herramientas**), los parámetros de Solver ya están introducidos (figura 6.5) por lo que sólo tenemos que pulsar en Resolver.

Podemos obtener de forma automática la duración de una repetición sin recurrir a la tabla que se ha creado con tan sólo introducir el número de repetición en la celda F5, el valor de la duración lo obtendremos de la celda H5.

En la hoja de cálculo también disponemos de información de los días laborables en el año y un cálculo en meses de trabajo (tiempo real de trabajo) de la duración de la estación. Como información de interés también se puede obtener de forma automática la reducción en días y en meses de la repetición que estamos calculando con respecto a la primera repetición, para ello tenemos que calcular previamente la duración de la primera repetición e introducir su valor en la celda H6.

Para exportar la información obtenida a Project tenemos una hoja (figura 6.6) con todos los aviones (repeticiones) en las que tenemos en distintas filas las duraciones para cada estación-subestación, esta disposición facilita la introducción de las duraciones con sólo **copiar** en Excel y **pegar** en Project. También se puede hacer uso de las opciones de exportación e importación de las que disponen tanto Project como Excel para pasar fácilmente la información de una aplicación a otra.

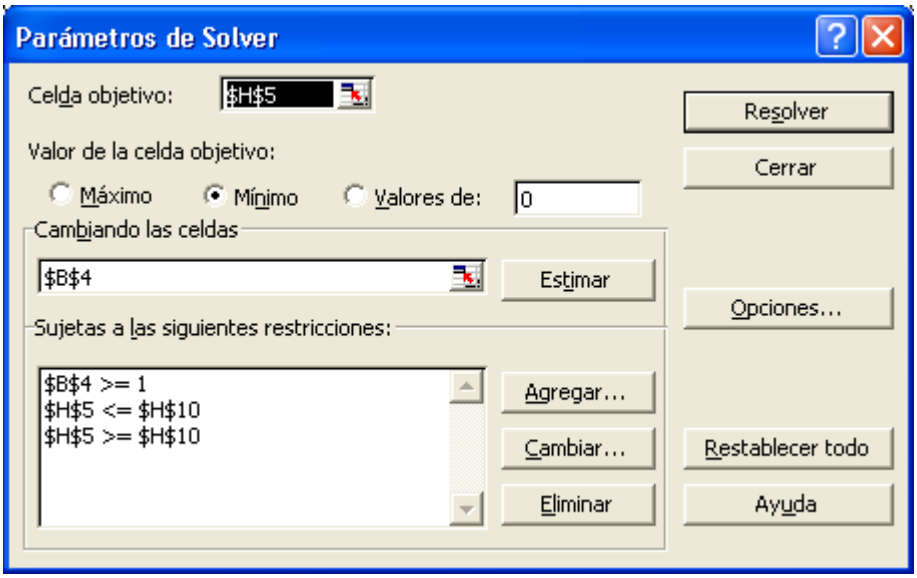


Figura 6.5. Parámetros de Solver.

AVIONES	1-8		9-16		17-24		25-32
	30		14,461507		11,708043		10,300537
	30		14,461507		11,708043		10,300537
	54,5		19,759824		14,731504		12,328479
	17,5		10,909791		9,5151322		8,7577754
	19		11,38915		9,8210358		8,9772899
	15		10,064944		8,9671686		8,3606647
	20		11,698742		10,016793		9,1169823
	45		17,876723		13,684846		11,637739
	23,831315		13,96424		11,487891		10,167239
	23,831315		13,96424		11,487891		10,167239
	39,572951		18,82153		14,34779		12,107221
	15,076406		10,665365		9,3988823		8,6842151
	16,167306		11,113056		9,6911515		8,8956532
	13,22597		9,874209		8,8745767		8,3013312
	16,887335		11,401754		9,8780291		9,0301296
	33,630621		17,102631		13,360118		11,447537
	20,828951		13,529154		11,283456		10,040601
	20,828951		13,529154		11,283456		10,040601
	32,816313		18,011182		13,99403		11,898065

Figura 6.6. Duración de los aviones con Excel.

6.3.2. Vinculación de las Tareas

La vinculación de tareas es uno de los aspectos más importantes y que hay que estudiar con más detalle para obtener una planificación correcta, y lo que es más importante fácilmente modificable. Existen cambios en determinados aspectos de la planificación que inmediatamente se traducen en modificaciones en otros (ya se mencionó el caso de $\mathbf{T_o}$), para que las modificaciones sean automáticas, las tareas tienen que estar correctamente vinculadas.

Vincular las tareas implica relacionarlas unas con otras, esta relación puede ser de diversas formas:

- Fin a comienzo (FC)
- Comienzo a comienzo (CC)
- Fin a fin (FF)
- Comienzo a fin (CF)

Podemos añadir a este tipo de vinculaciones características adicionales, como son tiempos de posposición que pueden ser de adelanto o retraso (tiempos negativos o positivos respectivamente).

Existen otras características de las tareas que pueden afectar a la vinculación, para acceder a ellas disponemos de un menú de delimitación con la posibilidad de fijar fechas. Tenemos las siguientes posibilidades de delimitar fechas:

- Debe comenzar el
- Debe finalizar el
- Lo antes posible
- Lo más tarde posible
- No comenzar antes del
- No comenzar después del
- No finalizar antes del
- No finalizar después del

En el caso de la FAL tenemos dos grupos de vinculaciones: las vinculaciones entre tareas pertenecientes a un mismo avión y la vinculación entre distintos aviones. A las vinculaciones pertenecientes a un mismo avión, las denominaremos vinculaciones horizontales, puesto que son las que hacen que el avión se refleje en el diagrama de Gantt como un todo formado por las distintas estaciones y subestaciones. A las vinculaciones entre los distintos aviones las denominaremos vinculaciones verticales, puesto que son estas vinculaciones las que nos darán la forma de escalera.

Las estaciones, generalmente, empiezan tras finalizar la anterior (serie), por lo que utilizaremos enlaces del tipo Fin a comienzo (FC). Las subestaciones tienen un comportamiento más flexible (serie y paralelo) por lo que se utilizarán vinculaciones, generalmente, del tipo Fin a comienzo (FC) y Comienzo a comienzo (CC).

Existen múltiples formas de vincular que consiguen un mismo efecto. Por ejemplo, para obtener una vinculación entre aviones podemos vincular las tareas “aviones” o se pueden vincular las estaciones entre aviones (fin a comienzo entre la última estación de un avión con la primera del siguiente avión). En ambos casos se obtiene la vinculación

del comienzo de un avión con la finalización del otro. La realización de la escalera implica un arduo y repetitivo trabajo, sobre todo, en la vinculación de tareas. Es por ello que la escalera se realizaba sin utilizar todos los vínculos que a priori y tras analizar el proceso se tienen en la realidad. Aunque las fechas obtenidas pueden ser las mismas, cualquier modificación o desplazamiento temporal (retraso o adelanto de fechas), puede ser un problema puesto que la planificación no se ajustará automáticamente según lo esperado. Cuanto mayor grado de complejidad de vínculos tengamos (el objetivo sería reflejar todos los vínculos reales entre operaciones) mejor será nuestra planificación.

Otro aspecto importante es el computacional. La vínculación es la forma principal que utiliza Project para fijar fechas, realizar el diagrama de Pert y analizar las tareas críticas. Al vincular tareas estamos introduciendo restricciones en nuestra planificación. Puede darse el caso de que algunas de las restricciones sean incompatibles entre sí (Project nos avisa en ese caso) o que el vínculo no produzca el efecto deseado. Por experiencia, el efecto producido al vincular, en ocasiones, es difícil de predecir en Project.

En el capítulo V se estudió cómo afecta la tarea crítica (generalmente la estación-subestación de mayor duración) en el enlace de estaciones, produciéndose un efecto que se denominó "troceo", es pues importante localizar la estación-subestación crítica. Si realizamos una vinculación detallada, vinculaciones verticales entre estaciones, observamos que se trocean los aviones. Una posible solución (la adoptada en PlanFal) es delimitar las tareas, de forma que las estaciones anteriores a la crítica empiecen lo más tarde posible y las posteriores lo antes posible. En el caso de que la tarea crítica sea una subestación el tratamiento es similar, debiendo empezar lo más tarde posible las subestaciones anteriores de la estación que las engloba y lo antes posible las posteriores.

6.4. RECURSOS

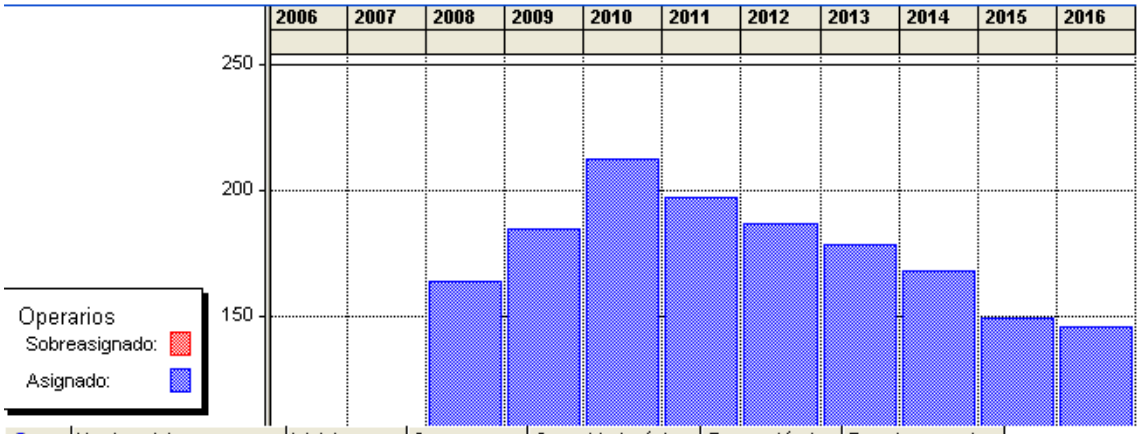
Como ya se ha comentado, para obtener una representación gráfica de la escalera de aviones, en principio, no es necesario especificar los recursos asociados a las tareas. Sin embargo, para la obtención de tiempos sí es necesario estimar los recursos que se necesitan y la disponibilidad de los mismos a lo largo del proyecto.

En Project se pueden asociar recursos a las tareas. Esto nos permite, mediante el uso de las vistas apropiadas (gráfico de recursos, hoja de recursos y uso de recursos), analizar las distribuciones de los recursos en nuestra planificación de un proyecto. Un resultado importante de este análisis es la posible saturación de recursos (figura 6.7). La saturación se produce cuando para la realización de una tarea se necesitan más recursos de los disponibles.

En el tratamiento de los recursos podemos tener dos enfoques en función de las pretensiones de nuestra planificación. En el primer enfoque, los recursos no afectan a la duración de las tareas. La asignación de recursos permite analizar su distribución, pero no repercutirá en las fechas obtenidas. En el segundo los recursos asignados a las tareas y una posible reasignación afecta a la duración de las tareas, esto es lo que se denomina en Project programación condicionada por esfuerzo. El segundo enfoque implica una mayor importancia de la planificación en la definición del proceso, puesto que es la herramienta que se utiliza para analizar la necesidad y disponibilidad de los recursos.

El cálculo de las duraciones de las tareas, en la escalera de aviones, se hace a partir de la duración de una avión de referencia y tras aplicar las curvas de aprendizaje, por lo que las duraciones se tratan como fijas y los recursos asociados, en general, no afectan a la duración de las tareas.

Son muchos los recursos que se necesitan para realizar el ensamblaje de un avión, pero para el análisis de disponibilidad y de saturación, así como para planificación de contrataciones, el recurso más interesante es el de los operarios que intervendrán en la realización del ensamblaje. Los operarios intervienen en todas las estaciones de trabajo, y su disponibilidad puede fijarse por el calendario laboral. Por estas razones y por la complejidad que supone en un primer análisis del proceso de ensamblaje el tratamiento de los recursos, el único recurso que se tiene en cuenta en la escalera es el de los operarios. En una planificación más detallada, donde se profundice más en las operaciones necesarias en cada estación, y donde la planificación de recursos sea importante, el número de recursos asociados a las taras será mas elevado y con mayor profundidad de análisis.



	Nombre del recurso	Iniciales	Grupo	Capacidad máxima	Tasa estándar	Tasa horas extra				
	Operarios	O	Operarios	250	10.000 Pta/h	15.000 Pta/h				
⊟	Operarios		3.529.096,62 h	Trabajo	273.749,26h	318.603,4h	305.220,55h	300.001,72h		
	A/C 001 St.1		*5.985,82 h*	Trabajo						
	A/C 001 St.2&3		*4.759,7 h*	Trabajo						
	A/C 001 St.4.1		*4.257,53 h*	Trabajo						
	A/C 001 St.4.2		*1.562,87 h*	Trabajo						
	A/C 001 St.5 (son 52, se obt		*4.238,98 h*	Trabajo						
	A/C 001 St.6		*774,6 h*	Trabajo						
	A/C 001 St.7		*2.947,87 h*	Trabajo						
	A/C 001 St.8		*0 h*	Trabajo						
	A/C ST St.1		*5.520,12 h*	Trabajo						
	A/C ST St.2&3		*5.952,61 h*	Trabajo						
	A/C ST St.4.1		*0 h*	Trabajo						
	A/C ST St.4.2		*0 h*	Trabajo						
	A/C ST St.5		*0 h*	Trabajo						
	A/C ST St.6		*0 h*	Trabajo						
	A/C ST St.7		*0 h*	Trabajo						
	A/C ST St.8		*0 h*	Trabajo						

Figura 6.7. Vista de Project con información de los recursos.

7. FUNCIONAMIENTO DE LA APLICACIÓN

7.1. INTRODUCCIÓN

Este capítulo se podría describir como un manual de funcionamiento de la herramienta de planificación automática de la línea de montaje final (PlanFal). Es de necesaria lectura para familiarizarse con la aplicación, puesto que aunque se ha procurado simplificar el uso de la misma, existen muchos conceptos que exigen de una justificación, puesto que son particulares del ensamblaje de aviones.

Lo primero que hay que saber es que PlanFal es una macro de Project, es pues importante, comprender que son las macros y cómo se ejecutan. Otro aspecto importante es la interfaz con el usuario. La entrada de datos necesarios para la planificación se hace mediante interfaz gráfica (ventanas). Estudiaremos cómo afectan a la ejecución del programa y cuál es el objetivo de las distintas ventanas y campos a rellenar.

7.2. MACROS

Microsoft Project incluye un lenguaje de programación eficaz y fácil de usar, llamado Microsoft Visual Basic para Aplicaciones.

Se puede utilizar este lenguaje para registrar o crear simples macros, o para escribir programas complejos. En cualquier caso, este lenguaje de programación puede ayudar a hacer más eficiente y productivo el trabajo. Con Visual Basic para aplicaciones se puede: automatizar las tareas cotidianas que consumen tiempo, integrar Microsoft

Project con Microsoft Office y otros programas a fin de facilitar el intercambio de datos entre ellos, cambiar el aspecto de Microsoft Project para adaptarlo a nuestras necesidades, simplificar la obtención y distribución de informes e información.

En Microsoft Visual Basic para Aplicaciones una macro es un conjunto de instrucciones automatizado que se utiliza para llevar a cabo una tarea específica.

Project ofrece dos maneras de crear macros: la grabadora de macros y el Editor de Visual Basic.

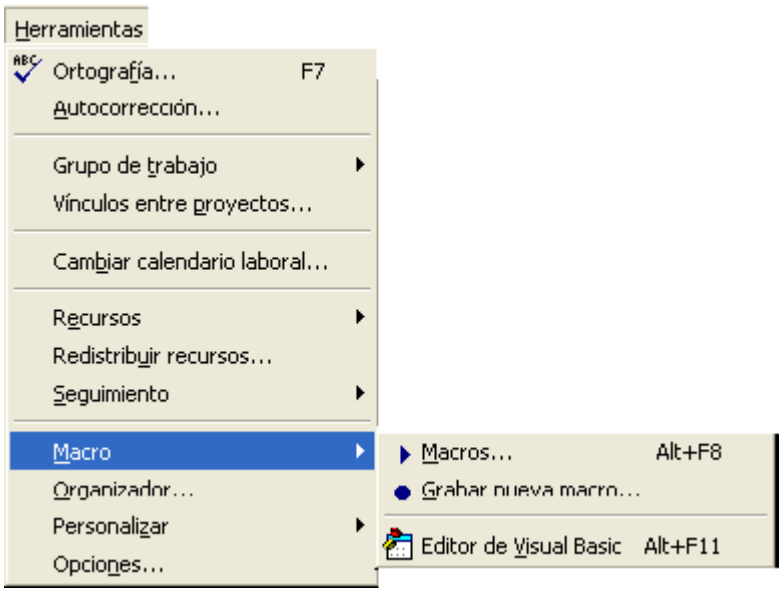


Figura 7.1. Menú de acceso a la macros.

La grabadora de macros es un método para crear macros sin programar directamente en Visual Basic. Es un método sencillo para automatizar tareas, además permite asignar teclas de acceso rápido (tecla de método abreviado) a las macros creadas. Consiste en grabar todas las acciones que hagamos para posteriormente repetirlas con sólo ejecutar la macro.

El editor de Visual Basic da acceso a un entorno de desarrollo (figura 7.2) para programar en Visual Basic. Los programadores acostumbrados a este lenguaje no tendrán ningún problema con este entorno, pues es el mismo que el utilizado en el desarrollo de aplicaciones Visual Basic genéricas.

Para la ejecución de macros debemos hacer clic en **Herramientas -> Macro -> Macros** donde nos aparecerá un menú con todas las macros disponibles, tras seleccionar la macro que queremos ejecutar pulsamos sobre la opción **Ejecutar**.

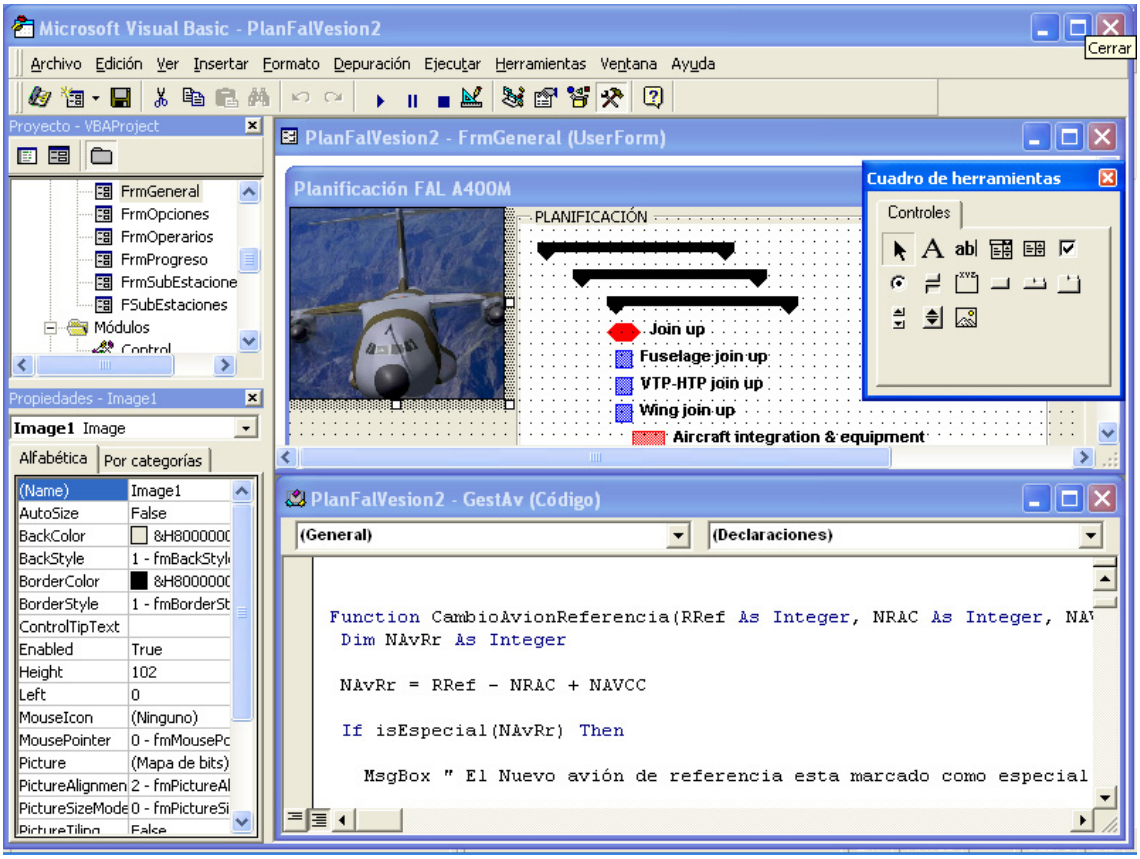


Figura 7.2. Editor de Visual Basic en Project.

7.3. INICIO DE PLANFAL

Al ser PlanFal una macro, esta ligado a una plantilla o a un documento, lo que supedita su ejecución a tener abierto un documento de Project (fichero con extensión mpp o mpt). Frecuentemente nos referiremos a la planificación que estamos desarrollando o a la ya existente en un fichero Project como proyecto.

Para arrancar la aplicación PanFal debemos acceder previamente a la ventana de ejecución de macros de Project (Figura 7.3). El nombre de la macro que arranca PlanFal se llamará “Nombre de fichero”!inicio (módulo inicio del proyecto Visual Basic

"Nombre de fichero"). "Nombre de fichero" es el nombre del archivo que contiene nuestra macro.

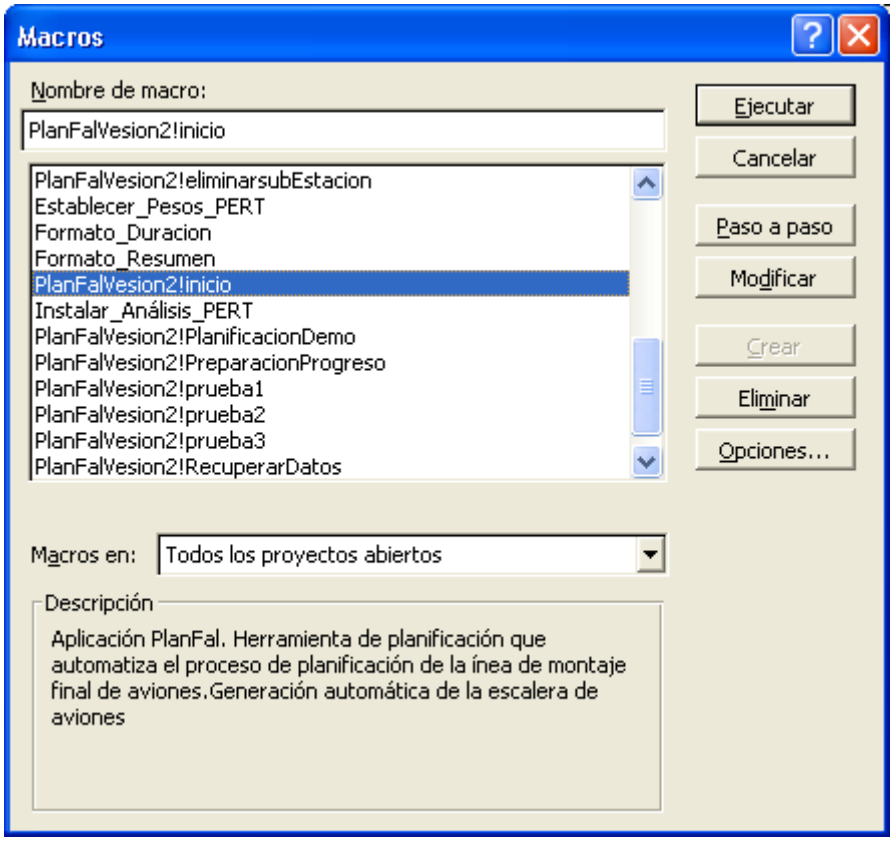


Figura 7.3. Ventana y módulo de acceso a PlanFal.

Para poder tener acceso a inicio, tenemos que tener disponible la macro. PlanFal se ha guardado en una plantilla de Project (extensión mpt). El nombre del fichero no es importante, pero es interesante que haga referencia a su contenido. En la figura 7.3 el nombre del fichero es PlanFalversion2, nombre que hace referencia a la aplicación y a la versión actual. Existen varías posibilidades para tener acceso a PlanFal. [1]

1. Abrir la plantilla, ejecutar la macro y guardar nuestra planificación con extensión mpp y el nombre que se quiera (al tener la macro en una una plantilla el menú por defecto de Guardar es **Guardar como**).

[1] Estas opciones dependen del Sistema Operativo en el que ejecutemos Project. En Windows NT pueden exisitir problemas con la plantilla Global.mpt si no se tiene permiso de administrador.

2. Abrir la plantilla. Abrir un proyecto nuevo u otro existente y trabajar en ese proyecto. El módulo inicio que arranca PlanFal estará disponible al estar abierto (en un segundo plano) la plantilla que contiene la macro. Guardar los cambios en el proyecto creado o ya existente.

3. Renombrar la plantilla como plantilla global (Global.mpt). Sustituir la plantilla global existente por la nueva. Cualquier proyecto que se cree o se abra a partir de ese momento tendrá disponible todas las macros de la nueva plantilla global.

Una característica muy importante de PlanFal es la capacidad que tiene en su arranque de extraer la información de planificación existente en el proyecto. Esto nos permite inicializar automáticamente todas las ventanas. Esto será así, siempre que el proyecto del que partamos se haya creado previamente con PlanFal, ya que la lectura de datos del proyecto se hace utilizando información de múltiples campos, generalmente inutilizados por el usuario de Project.

La inicialización de datos se realiza a partir de la información aportada por el avión de comienzo de curva y no por el de referencia, como sería lo lógico. Esto da una mayor flexibilidad a la aplicación, puesto que nos permite planificar un número de aviones pequeño tomando como datos de referencia un avión que no se encuentra en la escalera, ahorrándonos tiempo al no tener que planificar todos los aviones. El cálculo de los datos del avión de referencia a partir del avión de comienzo de curva lo realiza PlanFal en un nivel interno de programación.

Si partimos de un proyecto con datos que no se corresponden con los esperados por PlanFal, nos saldrá una ventana de aviso previa a la ventana principal de la aplicación.

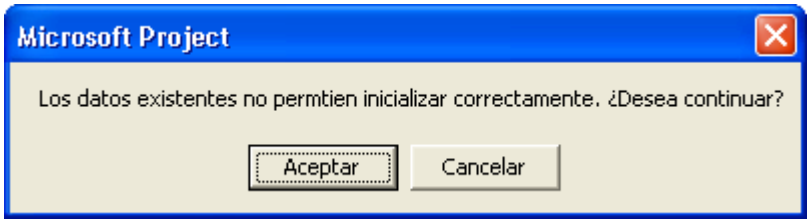


Figura 7.4. Mensaje de inicialización incorrecta.

Para continuar con la ejecución de PlanFal sin inicialización de datos pulsamos en **Aceptar**, en caso contrario saldremos de la aplicación y volveremos al proyecto.

Obviamente, si partimos de un proyecto en blanco, la aplicación arrancará sin ningún tipo de dato. En ocasiones puede ocurrir que no tengamos ninguna advertencia de inicialización incorrecta, pero que los datos de PlanFal no sean los esperados. Esto ocurre cuando PlanFal no encuentra nada incoherente (por ejemplo tipos de datos válidos), pero los campos de Project no contienen los datos preparados para el programa.

7.4. ENTRADA DE DATOS

La entrada de datos a PlanFal se realiza mediante ventanas, por lo que la interfaz con el usuario es sencilla. Tendremos formularios de entrada de datos para cada una de las opciones del programa.

Antes de entrar en detalle en las distintas interfaces que componen la aplicación, conviene explicar la filosofía del programa. El objetivo es generar la escalera de aviones a partir de la información aportada por el departamento de ingeniería encargado de estudiar y definir el proceso de ensamblaje (Ingeniería de Desarrollo del A400M). Los datos principales son:

- Número y nombre de estaciones y subestaciones y secuenciación.
- Repetición de referencia
- Duraciones de las estaciones-subestaciones para la repetición de referencia
- Factores de curva de aprendizaje tipo, para las operaciones de cada estación-subestación.

Con estos datos, ya se puede generar una escalera de aviones uniforme, en la que todos los aviones tienen el mismo número de estaciones – subestaciones.

Si queremos tener una escalera más real, debemos incluir otras características propias del ensamblaje de aviones. Estas son:

- Aviones especiales
- Avión de comienzo de curva
- Avión fin de curva (Avión de régimen)

Estos datos aportan mayor flexibilidad a la escalera, ya que introducen modificaciones en ella. Los aviones especiales son aquellos no siguen la escalera "normal", un motivo puede ser que necesiten un número de operaciones distinto. Como se explicó en el capítulo V, los primero aviones que se ensambla no siguen el proceso general, por lo que la curva no se les puede aplicar a ellos, para solventar esto se define un avión de comienzo de curva, a partir del cual la escalera tiene reducción de tiempos por curvas de aprendizaje. Para darle mayor versatilidad a nuestra planificación tenemos la opción de no aplicar curvas de reducción a partir de un número avión, en el que se considera que se ha alcanzado el permanente (régimen).

7.4.1 Ventana Principal

Es la primera ventana que nos sale tras el arranque del programa. En ella tenemos acceso a los distinto menús y podemos especificar el **Número de aviones** que se van a planificar y la **Repetición de Referencia** para la cual se tienen los datos que se van a introducir. Como dato informativo (el color oscuro de un campo, índica que no se puede modificar) se muestra el número de estaciones que componen la línea de ensamblaje.

Se deben especificar el número de aviones y la repetición de referencia en esta primera ventana porque son datos que afectan a los distintos menús. Podemos trabajar con una repetición (y por ende avión) de referencia mayor que el número de aviones, porque como ya se comentó, la inicialización de datos de PlanFal se realiza a partir del avión de comienzo de curva, el cual sí tiene que ser menor que el número de aviones que se quiere planificar.

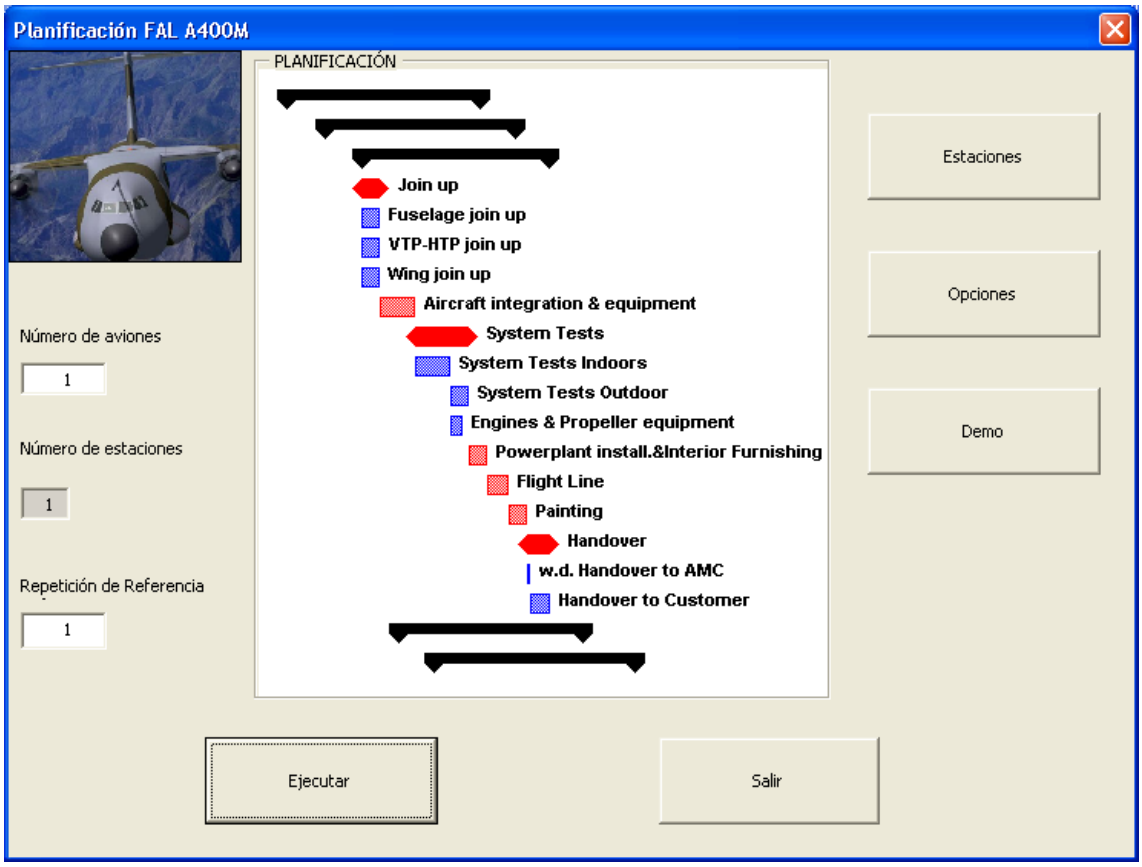


Figura 7.5. Ventana principal.

Observamos que desde esta ventana tenemos acceso a los menús principales: creación de **Estaciones** y menú de **Opciones**.

Demo no es un menú, sino que es una opción añadida al programa, que permite la introducción de valores demostrativos de forma automática. Los valores que se introducen están fijados internamente en el programa y se han tomado parecidos a los típicos que se utilizan en la línea de montaje final. Esta utilidad permite además, observar el funcionamiento del programa sin necesidad de introducir datos, y a su vez puede ser utilizado como un punto de partida para realizar la planificación.

Con el botón **Ejecutar** realizamos la planificación con los datos introducidos en la aplicación y con el botón **Salir** volvemos al proyecto sin hacer nada.

El tiempo de ejecución de PlanFal puede ser considerable para una cantidad de aviones superior a 50. Para facilitar el seguimiento de la evolución de la aplicación y

asegurarnos que todo funciona correctamente, PlanFal muestra una ventana de progreso (figura 7.6) en la que nos informa de las tareas que realiza y número de avión en el que se encuentra.

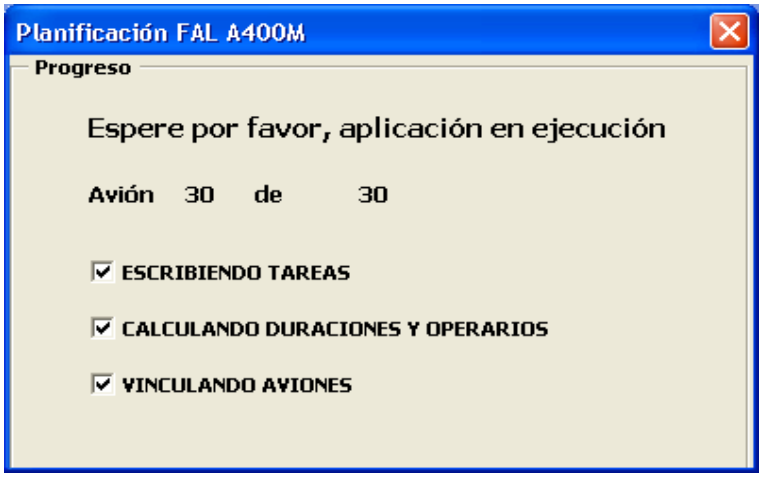


Figura 7.6. Ventana informativa del progreso de la ejecución.

7.4.2. Estaciones

El menú Estaciones es el principal de entrada de datos al programa, desde él también tenemos acceso al menú de subestaciones, siempre y cuando la estación contenga subestaciones. Si la estación esta formado por subestaciones, los datos de duraciones, curvas y operaciones se fijarán el menú **Subestaciones**.

Como ya se explicó en los capítulos V y VI, sólo se necesitan los tiempos de un avión (referencia) y a partir de esos tiempos el programa calcula el resto aplicando la curva de aprendizaje. El número de avión para el que se introducen los datos (ver en figura 7.7 parte superior del formulario) lo calcula automáticamente el programa en función de la repetición de referencia fijada por el usuario (ventana principal) e información adicional como la posición y repetición del avión de comienzo de curva (menú de **Opciones -> Configurar -> Aviones**).

Para el cálculo del número de avión se aplica la siguiente ecuación:

$$NA = NACC + RR - RACC \tag{7.1}$$

donde:

NA = número de avión de referencia

NACC = Número de avión de comienzo de curva

RR = repetición de referencia

RACC = repetición de avión de comienzo de curva

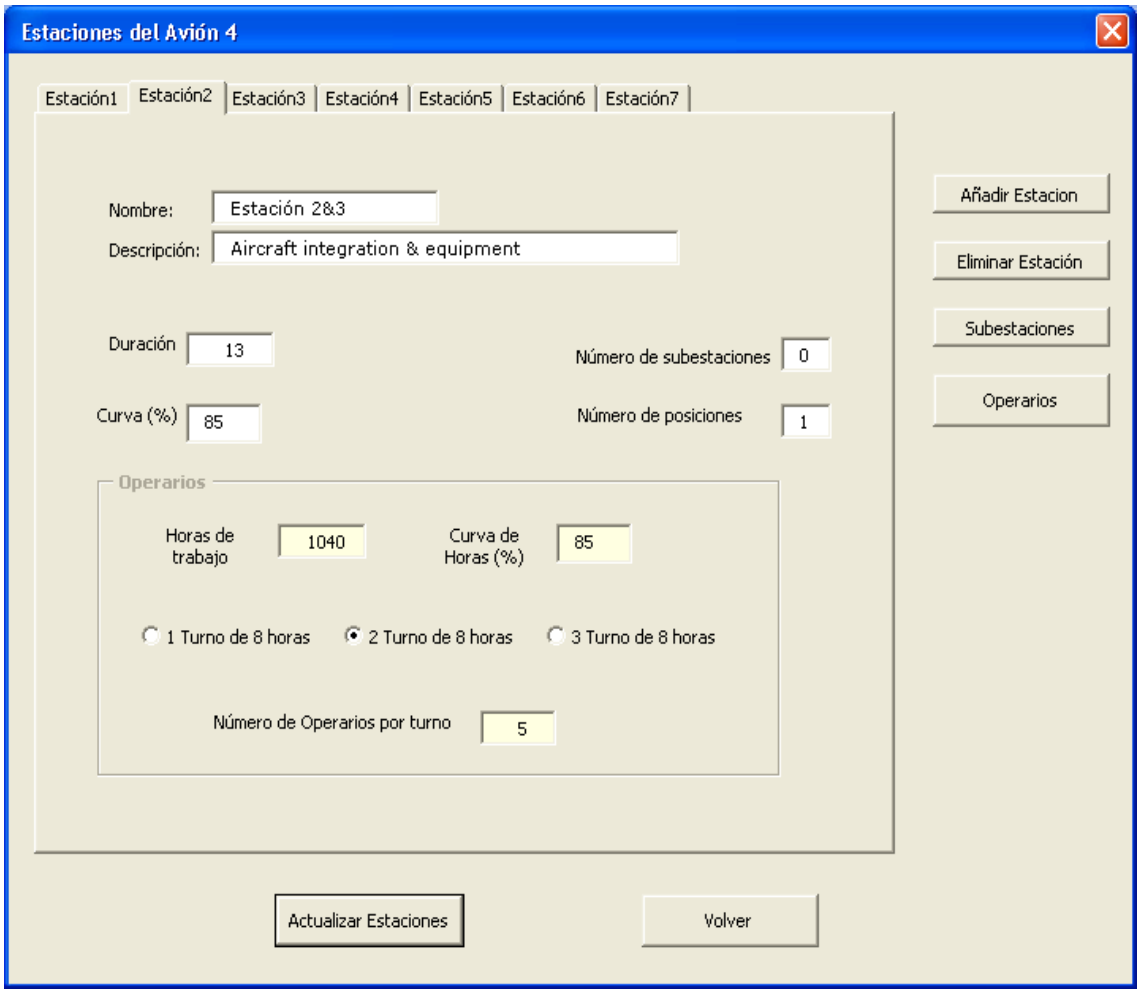


Figura 7.7. Entrada de datos de las estaciones.

Si inicializamos desde un proyecto en blanco partiremos con una estación . Podemos añadir y eliminar estaciones mediante los botones **Añadir Estación** y **Eliminar Estación**. Para controlar la secuenciación, la estación añadida será la siguiente a la estación en la que estemos cuando se añada una estación. Para eliminar una estación, debemos posicionarnos en ella (seleccionar la pestaña de la estación que queremos eliminar) y pulsar en **Eliminar Estación**. La numeración de las estaciones se ajustará de forma automática tras eliminar o añadir una estación. Para el programa las estaciones siempre van en serie (una tras otra) y sin ninguna posibilidad de adelanto o retraso temporal.

Los datos generales que tenemos que introducir en una estación son:

- Duración: duración de la estación en días. Por defecto 0 días.
- Curva (%): factor aplicable de reducción por curva de aprendizaje. Por defecto 100% (no hay reducción).
- Número de subestaciones: número de subestaciones que componen la estación. Por defecto 0 subestaciones.
- Número de Posiciones: el número de posiciones índica cuantas estaciones iguales tenemos en paralelo. Por ejemplo, tener 2 posiciones significa que tenemos duplicada esa estación, lo que se traduce en una reducción de tiempos en la escalera (ver capítulo V).

Los datos de operarios son:

- Turnos: número de turnos al día (turnos de 8 horas).
- Horas de trabajo: duración en horas reales de trabajo que se necesitan para realizar todas las operaciones de las que consta una estación.
- Operarios por turnos: número de operarios por turno y por día de los que se dispone.
- Número de Posiciones: el número de posiciones índica de cuantas estaciones iguales y en paralelo se dispone para las operaciones.
- Curva de horas: factor de reducción de horas debido al aprendizaje. Valor en tanto por ciento.

Los datos relacionados con operarios sólo se pueden visualizar en esta ventana. Para modificarlo se tiene que pulsar sobre la opción **Operarios**.

Los datos de duración de una estación en días, la duración en horas y los operarios que intervienen están relacionados por la ecuación:

$$\text{horas de trabajo} = \text{operarios por turno} \times \text{turnos} \times 8 \times \text{días} \qquad (7.2)$$

En (7.2) observamos la relación entre días, horas y operarios Debemos indicar el dato que permanece fijo cuando modificamos alguna de las variables. Para ello disponemos de tres opciones de fijación, una por cada variable. Cuando nos posicionamos sobre un campo para modificarlo, automáticamente se sombrea la fijación (CheckBox) de ese dato, teniendo la posibilidad de elegir cual de las dos variables restantes se quiere fijar y cual se va a modificar automáticamente. Al depender el número de operarios de los turnos fijados por día, debemos tener en cuenta que el número de operarios que se tendrá en la planificacion de Project dependerá del número de operarios y turnos que tengamos en las casillas correspondientes.

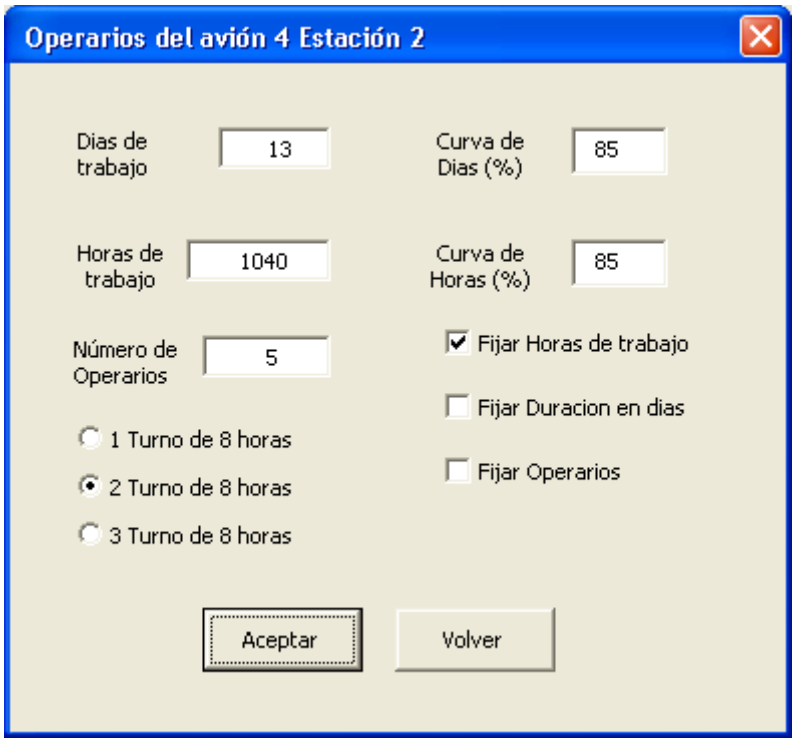


Figura 7.8. Ventana para modificar operarios, turnos, duraciones y curvas.

La posibilidad de poder definir una reducción de horas distinta a la de días (factores de curva distintos) nos permite simular distintos escenarios:

- Si la reducción de horas es mayor que la de días (factor en % menor que el de días), se tarda progresivamente más en hacer las mismas tareas, por lo que el número de operarios se reduce progresivamente.

- Si la reducción de días es mayor que la de horas el número de operarios aumenta progresivamente.

El aumento o reducción de operarios que podemos conseguir con esta técnica sigue una ley exponencial al surgir de diferencias entre comportamientos de curvas de reducción exponenciales. Si los factores de curva son iguales, entonces el número de operarios permanece constante.

Para actualizar los datos de operarios en la ventana de estaciones pulsamos en **Aceptar** (figura 7.8), si no queremos realizar ningún cambio pulsamos en **Volver**.

Cuando queramos guardar los cambio hechos en el menú estaciones pulsamos sobre **Actualizar Estaciones** (figura 7.7). Esto guardará los cambios hechos en todas las estaciones. Es un botón que afecta a todas las estaciones, por lo que no necesitamos actualizar los cambios para cada estación.

7.4.2.1. Subestaciones

Si la estación esta formada por subestaciones (casilla **Número de subestaciones** (Fig.7.7) con un valor mayor que cero), entonces sólo podemos fijar en el menú estaciones el **Nombre**, la **Descripción**, el **Número de posiciones** y **el Número de subestaciones** (Fig.7.7). Al estar formada la estación por subestaciones, es una tarea de jerarquía superior, por lo que la duración se fija automáticamente por Project en función de sus subtareas (subestaciones).

Para acceder al menú de subestaciones debemos pulsar sobre el botón **Subestaciones**. Si el número de subestaciones es cero, nos aparecerá un mensaje de error (figura 7.9).

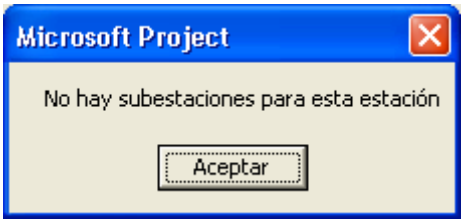


Figura 7.9. Mensaje de error, en el caso de que no haya subestaciones.

Observamos que la ventana (figura 7.10) es parecida a la anterior, con la característica adicional de poder definir la posición que ocupan las subestaciones dentro de la estación. Para ello debemos fijar para cada subestación su predecesora. Para entender mejor como se fijan las posiciones especificando predecesoras vamos a analizar algunos casos concretos.

Si queremos que la subestación empiece al comienzo de la estación debemos marcar la opción **Sin predecesora**. En caso contrario podemos indicar después de que estación empieza. Veamos un caso en el que tenemos una estación formada por tres subestaciones que se llaman Subst1, Subst2 y Subst3 respectivamente.

En el caso de que las tres subestaciones estén en serie y en el orden especificado por su numeración tendríamos que introducir los siguientes datos:

Subst1: sin predecesora.
Subst2: predecesora = 1
Subst3: predecesora = 2

Si queremos que las tres subestaciones estén en paralelo:

Subst1: sin predecesora.
Subst2: sin predecesora
Subst3: sin predecesora

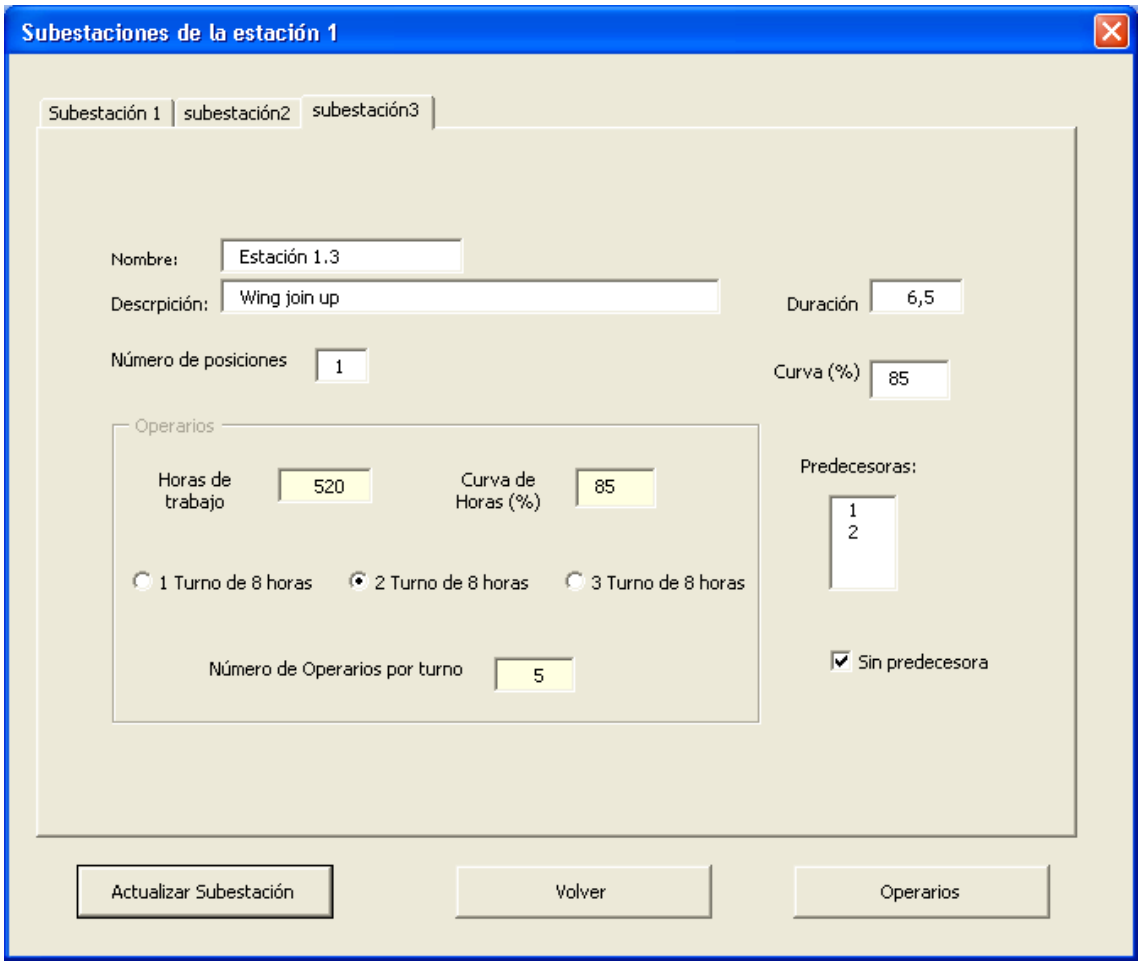


Figura 7.10. Menú Subestaciones.

Si queremos que Subst2 empiece al finalizar Subst1 y la Subst3 esté en paralelo con Subst1:

Subst1: sin predecesora
Subst2: predecesora = 1
Subst3: sin predecesora

El término predecesora tiene sentido para subestaciones anteriores, por lo que sólo podremos escoger como predecesoras subestaciones anteriores a la que estemos. Debido a la posibilidad de especificar la secuenciación de las subestaciones no se incluye la posibilidad de añadir y eliminar subestaciones dentro de este menú. Se puede añadir y eliminar subestaciones desde el menú estaciones con sólo modificar el número de las subestaciones. Si el cambio de número implica una disminución las subestaciones que

se eliminan son las últimas del formulario, y si es un aumento las subestaciones añadidas se posicionan al final.

Al igual que ocurria en el menú estaciones para modificar los datos de operarios debemos acceder al menú **Operarios.**

7.4.3. Opciones

Con este menú podemos modificar algunos parámetros de la aplicación. Podemos acceder a una ventana (Figura 7.11) donde podemos especificar cómo se mostrarán los resultados de la planificación y también podemos acceder a otra de configuración (Figura 7.12), donde se pueden introducir datos adicionales y acceder a las ventanas que nos permiten introducir datos relacionados con los aviones “especiales”.

7.4.3.1. Opciones de visualización

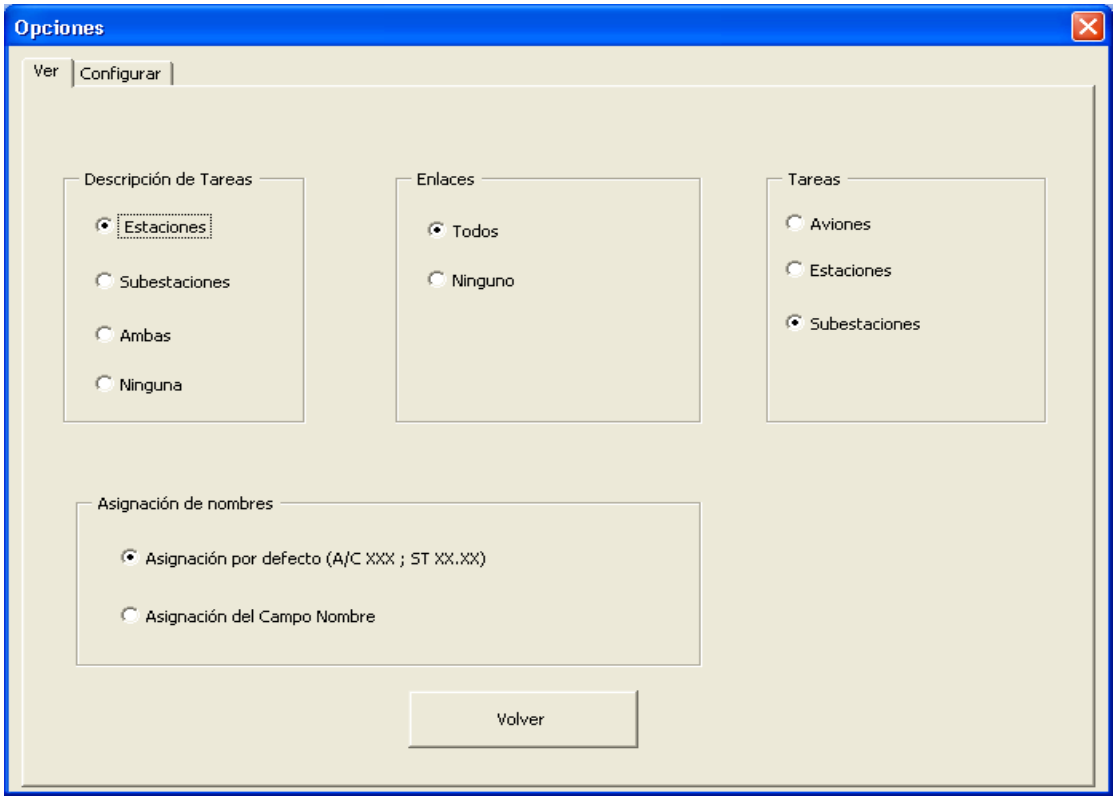


Figura 7.11. Opciones de salida del programa.

Como podemos observar en la Figura 7.11 tenemos la posibilidad de especificar como se mostrarán los resultados de la aplicación.

- Descripción de Tareas: permite visualizar en el diagrama de Gantt una descripción de las tareas. El resultado es un texto a la derecha de la barra que representa la tarea. Podemos optar por visualizar la descripción de las estaciones, subestaciones, ambas o en ningún caso.

- Enlaces: para una correcta planificación es necesario enlazar las tareas. Mediante esta opción podemos elegir entre ver o no los enlaces realizados en el diagrama.

- Tareas: con esta elección especificamos el grado de detalle con el que veremos la planificación al terminar la aplicación.

- Asignación de nombres: se puede tomar como nombre de la tarea el texto especificado en el campo "Nombre" de los datos de Estaciones y Subestaciones. Puesto que este parámetro es fundamental para Project y en previsión de posibles errores o ausencia de información, la asignación de nombres para la tarea se puede hacer de forma automática. Esta es la opción por defecto con la que trabaja la aplicación.

Dentro del menú Opciones-> Ver, existen parámetros que se pueden modificar fácilmente desde Project una vez realizada la planificación, como son la visualización de descripciones, enlaces y tareas. Una opción que no podremos modificar desde Project, sin volver a ejecutar PlanFal, es la asignación de nombres.

7.4.3.1. Opciones de configuración

El menú de configuración (Figura 7.12) es más importante que el de visualización y afecta a la realización de la planificación. Esta dividido en tres secciones, Aviones, Fechas y Operarios. En cada una de ella se pueden modificar parámetros que afectan al comportamiento de la aplicación.

En la primera Sección (Aviones) podemos introducir información adicional en la escalera de aviones. Podemos fijar el avión de comienzo de curva y su repetición, así

como el avión en el que podemos suponer que hemos alcanzado el permanente (avión de régimen o fin de curva). La posibilidad de poder especificar el avión de comienzo de curva y su repetición abre un gran abanico de posibilidades y permite tratar de forma distinta a los primeros aviones que se ensamblan. Una atención especial merece la posibilidad de introducir aviones especiales, por la complejidad que ello implica y las posibilidades que ofrece.

Para crear un avión especial sólo tenemos que poner en el cuadro **Aviones Especiales** el número de avión (orden que ocupa en el ensamblaje) que queremos catalogar como especial y pulsar en **OK**. El campo Aviones Especiales sirve tanto para crear aviones especiales como para acceder a los ya existentes, para consultar o modificar sus datos (Figura 7.12).

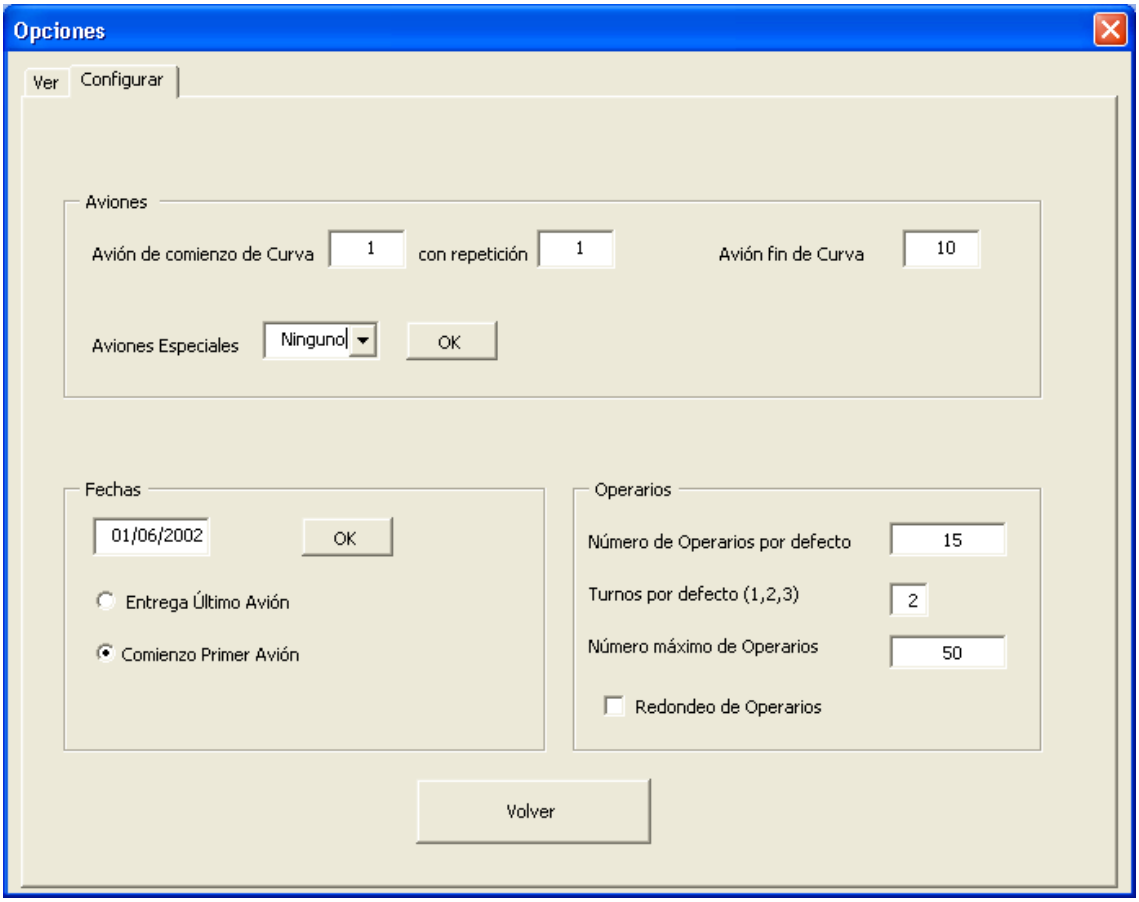


Figura 7.12. Opciones de configuración.

Como ya se explicó anteriormente, los aviones especiales no siguen la "regla" general marcada por el avión de referencia, aún así, normalmente son pocas las variaciones que tienen respecto al ensamblaje del resto de aviones.

Podemos especificar se queremos aplicar curvas de aprendizaje al avión especial seleccionando la opción **Aplicación de Curvas** (figura 7.13). Esta opción sólo se presenta para aviones posteriores al avión de comienzo de curva. La duración de los aviones que tengan seleccionada la opción de aplicación de curvas no vendrá fijada por la duración que tengan sus estaciones o subestaciones, puesto que se asume que estos datos son los de la repetición de referencia. Esta opción es muy interesante para los aviones de refurbishing, puesto que aún siendo aviones especiales, generalmente se les aplican curvas. Los aviones que no tengan seleccionada esta opción toman sus duraciones de los datos que tienen en las ventanas de estaciones y subestaciones, lo que nos permite tener la opción de fijar "exteriormente" duraciones, aunque a priori estemos en una repetición de la curva. La opción se nos presenta por defecto como activada.

Para facilitar la creación de aviones especiales se dispone de la posibilidad de **Copiar** (Figura 7.14). Podemos copiar los datos referentes a estaciones y subestaciones del avión de referencia y del resto de aviones que tengamos catalogados como especiales. Por lo tanto, si queremos crear un avión especial, el proceso lógico sería crearlo, copiarlo y por último modificarlo adaptándolo a las características concretas del avión.

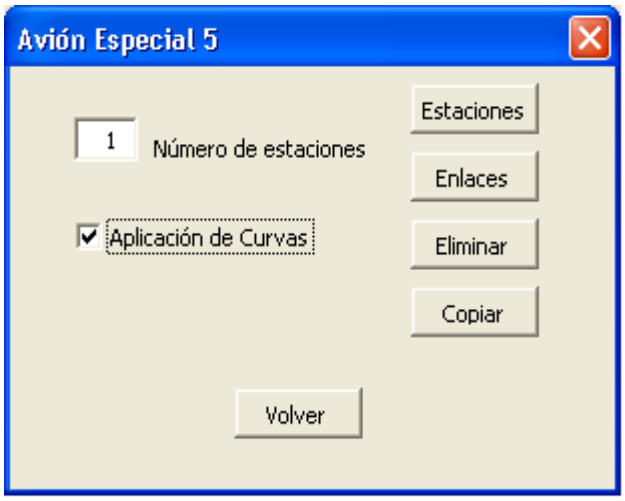


Figura 7.13. Menú de Opciones para aviones especiales

El programa facilita automáticamente los aviones a partir de los cuales podemos copiar los datos. Una vez seleccionado (el número de avión debe estar sombreado), sólo tenemos que pulsar en **Aceptar**.

Figura 7.14. Opción de copiar los datos del avión de otro ya existente.

Para modificar los datos del avión especial pulsamos en aceptar, accediendo a un menú de estaciones igual al de la figura 7.7.

Una característica fundamental de los aviones especiales es que al no seguir la curva de forma normal, sus enlaces con el resto de aviones tienen que definirse de forma manual por el usuario del programa. Esta es quizás, la parte más complicada del manejo de PlanFal. Si no se especifican bien estos enlaces, existe la posibilidad de encontrarnos con un error en tiempo de ejecución, que se produce al intentar enlazar tareas que no existen, además de tener la posibilidad de encontrarnos con una planificación incorrecta. El hecho de que sea el usuario quien en última instancia sea el responsable de los enlaces de los aviones especiales, se debe a la importancia de vincular estos aviones y a la dificultad de hacerlo de forma automática debido a la casuística que implica.

En el caso de que la escalera que resulte no se ajuste a la realidad, podemos delimitar el comienzo de los aviones especiales a una fecha especifica, enlazando posteriormente el avión especial con el siguiente. Gracias a la gran cantidad de vinculaciónes que tienen las tareas “escritas” con PlanFal, la escalera posterior al avión especial se ajustará perfectamente como un bloque.

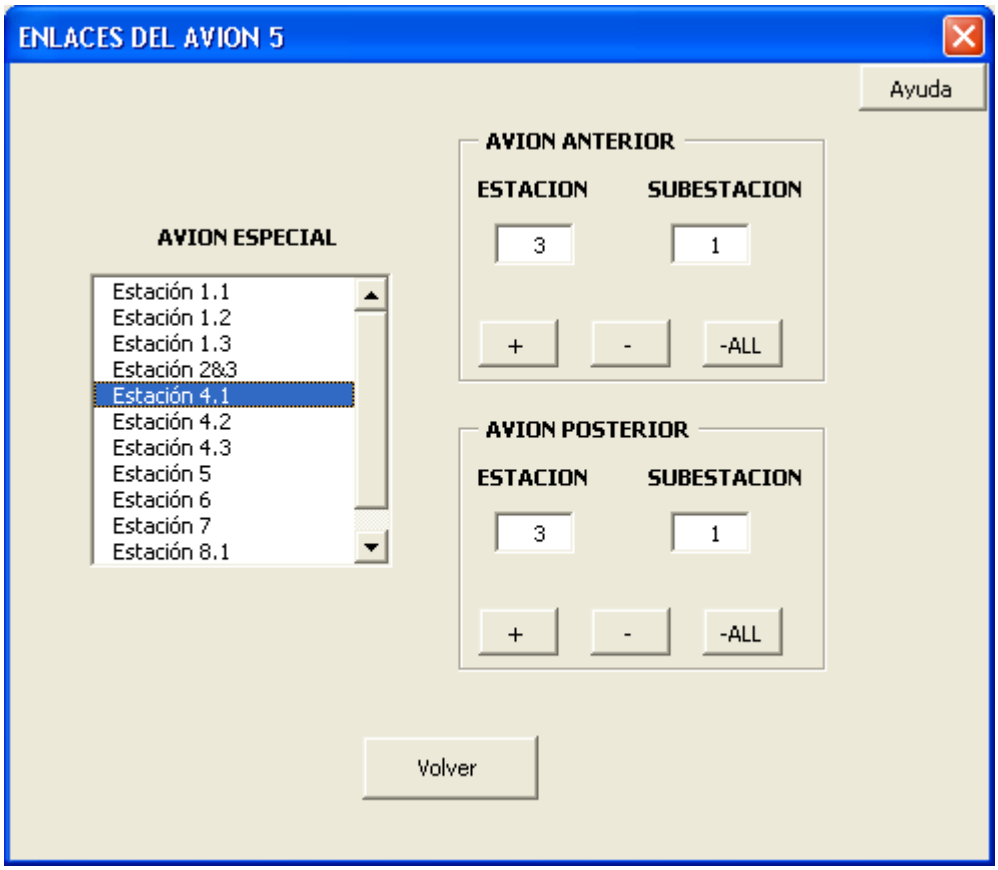


Figura 7.15. Ventana de enlaces de aviones especiales.

Existe una pequeña ayuda accesible desde el programa (botón de ayuda de la esquina superior derecha), que no es más que una pequeña explicación de cómo se debe completar esta ventana.

En el menú de enlaces (figura 7.15) tenemos los siguientes campos:

- Avión especial: el nombre que aparece en el listado de estaciones-subestaciones. Es el que tenemos definido en el campo nombre de la estación-subestación. Si este campo esta vacío tendremos una asignación por defecto del tipo (ST?/?).

- Aviones Anterior-Posterior: el campo a rellenar es el número de la pestaña del formulario estación- subestación correspondiente. Por lo tanto debemos fijarnos en el número de la ventana estación o subestación del avión con el que queramos enlazar (avión posterior) o el número de la ventana estación o subestación del avión enlazado con el especial (avión anterior).

La estación-subestación del avión posterior se enlaza con la estación seleccionada del avión especial y la estación-subestación seleccionada del avión especial (seleccionada en la lista) con la indicada en las casillas del avión anterior.

Para añadir-modificar un enlace con el avión especial, se debe seleccionar de la lista la estación del avión especial y completar las casillas estación y subestación del avión posterior y anterior. Para que el enlace se almacene se tiene que pulsar en el botón + y para eliminar el enlace en -.

El programa presenta por defecto unos enlaces cuando entramos a este menú. Esto se hace para facilitar la tarea de enlazar y de forma seudo-inteligente, es decir, se utiliza una lógica de enlaces adecuados, si bien, estos enlaces pueden ser correctos (no provocan un error en tiempo de ejecución), no se garantiza en absoluto que tengan lógica para la planificación.

Figura 7.16. Mensaje de inserción de enlace incorrecto

Debido a que existe un control de enlaces dentro del programa, en principio no se permiten enlaces inexistentes (Fig 7.16). Esto provoca que en ocasiones no se acepten los cambio hechos en las casillas de enlaces. Cuando esto ocurra, debemos previamente borrar todos los enlaces preexistentes (botón –ALL).

En una segunda sección del menú de configuración podemos fijar la fecha del comienzo del ensamblaje (comienzo primer avión) o la fecha de entrega del último avión. Estas dos opciones se corresponden con dos filosofías de trabajo distintas, en la primera el proyecto tiene una fecha de comienzo fija y establecida y con la planificación obtenida estableceremos, entre otros datos, la duración total del ensamblaje y su fecha

de finalización. Por otra parte, fijando la entrega (obligaciones con el cliente), podemos obtener una fecha adecuada del comienzo del proyecto para cumplir con las exigencias de finalización.

Por defecto la fecha que tenemos es la de inicio del proyecto y la opción de fijar el comienzo del ensamblaje. El formato de fecha es: dd/mm/aaaa

El último de los bloques del menú de configuración nos permite facilitar la entrada de operarios cuando se crean estaciones y subestaciones nuevas (valores por defecto) y establecer condiciones de inserción de recursos en la ejecución del programa. Es interesante el campo de Número de Operarios por defecto, porque nos permitirá realizar análisis de recursos (Gráfico de Recursos en Project), y obtener en qué fechas se han asignado de forma incorrecta recursos. Con el redondeo de operarios garantizamos que el número de operarios siempre es un entero (redondeo siempre al entero inferior).

8. DESARROLLO DE LA APLICACIÓN

8.1. INTRODUCCIÓN

El objetivo de este apartado es servir de guía a quien este interesado en conocer los fundamentos de la aplicación y, junto con el anexo II, constituir una referencia para entender el funcionamiento interno del programa. Para ello, en un primer paso se explicará el desarrollo que ha sufrido el programa hasta alcanzar todas las funcionalidades que presenta actualmente, posteriormente profundizaremos en el enfoque y los objetivos que se buscaron en la etapa de programación.

En el transcurso del desarrollo del programa se dividieron los esfuerzos en dos áreas. Una era controlar Project para realizar las tareas de forma automática, y la otra era el desarrollo de la interfaz gráfica de la aplicación. Las dificultad de la primera radica en conocer los objetos de Project y transformar la información aportada por el usuario en información valida para Project y viceversa. La interfaz gráfica esta influenciada por los requerimientos del usuario, por lo que fue creciendo a medida que el programa evolucionaba.

8.2. EVOLUCIÓN DEL DESARROLLO

El programa ha ido creciendo a medida que se conseguían nuevos objetivos y se sugerían por parte de los futuros usuarios nuevas opciones y mejoras. Es por ello que se puede hablar de varias versiones de PlanFal. Vamos a hacer un recorrido por las distintas versiones, ya que nos permitirá tener una visión global de la evolución de la

aplicación. En la primera versión lo importante era facilitar el desarrollo de las tareas más repetitivas, posteriormente se añadieron mejoras al programa para que este fuera más flexible y por último el objetivo fue que abarcase toda la planificación.

8.2.1. Versión 0

Como ya se explicó anteriormente, la labor de planificar una escalera de aviones con Project puede llegar a ser una tarea ardua y repetitiva, es por ello que se estudió la posibilidad de facilitar la realización de partes de estas tareas. Gran parte de la labor de planificación se desarrollaba con Excel (cálculo de duraciones y definición de las tareas), por lo que en un primer momento se estudió la posibilidad de realizar una hoja de cálculo y exportar los resultados posteriormente a Project. Esta idea se descartó cuando se comprobó el potencial que presentaba en sí mismo Project y la conveniencia de controlarlo en un nivel de programación.

En un primer momento, se centraron los esfuerzos en automatizar procedimientos básicos y costosos en tiempo, como por ejemplo, la escritura de las tareas. Teniendo en cuenta que una escalera de aviones puede llegar a estar formada por miles de tareas (doscientos aviones pueden ser aproximadamente 3500 tareas), era interesante la idea de que la definición de tareas corriese a cargo de un módulo de programación accesible mediante una macro. Básicamente, era una macro que escribía tareas de forma jerarquizada, dándonos la posibilidad de definir tareas en tres categorías, aviones estaciones y subestaciones.

Desarrollada la macro y observando las posibilidades que el desarrollo de las mismas ofrecía, el siguiente paso fue el cálculo de duraciones de las tareas. Llegado a este punto era necesario dotar al programa de una interfaz gráfica, que permitiese al usuario introducir los datos referentes a la duración de las tareas. El esfuerzo implicaba controlar el número de estaciones y subestaciones que iban a formar parte del ensamblaje.

La dificultad principal que presentó la interfaz gráfica era su dinamismo, puesto que existen ventanas (estaciones y subestaciones) que se crean o se eliminan en función de

los datos aportados por el usuario, lo que implica creación de interfaces en tiempo de ejecución.

El manejo de formularios en tiempo de ejecución complica la programación, pues no tenemos la posibilidad de controlar eventos de controles que se crean en tiempos de ejecución, por las limitaciones del entorno de desarrollo utilizado. En Visual Basic, como en todo lenguaje de programación, es muy importante el uso de librerías. Existen entornos de desarrollo de Visual Basic más potentes que el que viene por defecto con Project, que permiten el tratamiento de formularios y arrays dinámicos con controles. En el entorno que se ha desarrollado la aplicación estas funciones no están disponibles, y en un esfuerzo por no tener problemas con la portabilidad de la aplicación, se ha adoptado la política de no añadir librerías adicionales, por lo que la solución adoptada finalmente para la creación de formularios no es la más elegante, pero sí la única posible, o al menos la única encontrada, para crear nuevos formularios (estaciones y subestaciones) en tiempo de ejecución. En esta solución tenemos un formulario adicional sin métodos, es decir, sólo objetos gráficos (controles), que se copia y pega cuando se añaden nuevas estaciones o subestaciones. Debido a la imposibilidad de controlar eventos de los nuevos controles, la gestión de eventos se realiza por controles pertenecientes al formulario principal.

Las características fundamentales de esta versión son:

- Duración en días.
- Curvas de aprendizaje en días.
- Avión de referencia.

Con esta versión limitada se obtenía una escalera uniforme, en la que se podía observar la reducción de tiempos debido a la curva de aprendizaje, que servía como punto de partida para una planificación más detallada, puesto que posteriormente se introducían los cambios necesarios para aproximarnos a la realidad.

Un aspecto importante del programa, y en general para todas las versiones, es la duración de la realización de la escalera. Para una escalera de 200 aviones el programa tardaba en esta primera versión alrededor de una hora en un ordenador medianamente

rápido (600 MHz). Si bien se ha conseguido reducir el tiempo de ejecución en versiones posteriores, la reducción no es muy significativa con grandes cantidades de aviones (a partir de 100 aviones), puesto que se ha observado que el tiempo que necesita Project para ejecutar la macro sigue una evolución exponencial, es decir, tarda más en realizar el avión 100 que el 1, y mucho más en realizar el 200 que el 100.

8.2.2. Versión 1

La versión 1 significó un cambio fundamental en el programa, primordialmente en un nivel de programación. La diferencia fundamental con la versión anterior radica en la posibilidad de tener aviones especiales.

Anteriormente se trabajaba con un único avión (avión de referencia) que era la base de la escalera, por lo que solamente teníamos datos para ese avión. Los cálculos para el resto de los aviones se hacían cuando la aplicación insertaba los datos en Project. Al tener que trabajar con más aviones se creó un objeto avión, que tuviese todas las variables con las que trabajamos antes, más todas las propiedades nuevas que nos permiten distinguir entre distintos tipos de aviones.

Este cambio de filosofía, impuesto por las nuevas necesidades, implicó una reescritura del código casi completa. La decisión de volver a escribir el código con un nuevo punto de vista, trajo consigo notables beneficios. Aparte de los propios de disponer de un objeto propio (avión), se mejoró el tiempo de ejecución de la aplicación gracias a la experiencia adquirida en los desarrollos de algoritmos para la versión anterior. La programación también se hizo mucho más modular, puesto que de esta forma tenemos código más reutilizable y un mayor control de errores a costa de un mayor número de llamadas a métodos.

La introducción de los aviones especiales se debió principalmente al interés en planificar los primeros aviones del ensamblaje. Estos aviones tienes procesos distintos al resto. Para poder definir a partir de que avión empieza la curva “normal”, se introdujo la posibilidad de definir un avión de comienzo de curva. Otra característica de esta

versión es que ya no hablamos de avión de referencia sino de repetición de referencia, con lo que dotamos de flexibilidad a la curva.

Para gestionar estos nuevos datos se creó un menú de Opciones, este nuevo menú, aparte de contener la interfaz gráfica que nos permite acceder a los aviones especiales, da la opción de indicar cómo queremos ver los resultados en el diagrama de Gantt.

Las características fundamentales de esta versión son:

- Duración en días.
- Curvas de aprendizaje en días.
- Repetición de referencia.
- Avión de comienzo de curva.
- Aviones especiales.
- Menú de opciones.

8.2.3. Versión 2

Es esta la versión definitiva de PlanFal. Las mejoras se centran en nuevas funcionalidades y no existen muchos cambios fundamentales a nivel de código. Se mejora el menú de opciones, dando posibilidades de fijar fechas y se mejora la introducción de datos referentes a los aviones especiales, como es la opción de copiado de aviones. Donde realmente existe cambios fundamentales con la versión anterior es en la posibilidad de introducir datos relativos a los operarios y horas de trabajo.

Para un mejor seguimiento de la evolución del programa cuando realiza la escalera, y dado al elevado tiempo de ejecución, esta versión dispone de un formulario en el que vemos el progreso del trabajo que esta realizando Project y en que avión se encuentra.

Las características fundamentales de esta versión son:

- Duración en días.
- Curvas de aprendizaje en días.

- Repetición de referencia.
- Avión de comienzo de curva.
- Aviones especiales.
- Menú de opciones mejorado (fechas y operarios).
- Operarios.
- Horas de trabajo y curva de horas.
- Visualización del progreso en ejecución.

8.3. ESTRUCTURA DEL PROGRAMA

Si accedemos al entorno de desarrollo de Visual Basic del que se dispone en Project, y observamos el examinador de objetos, nos encontramos con los siguientes elementos:

- Microsoft Project Objetos: son los documentos asociado al proyecto. En nuestro caso el fichero de Project que contiene la macro.

- Formularios de usuario: son todos los ficheros[1] .frm asociados al proyecto. Contienen objetos gráficos llamados controles y código asociado a esos controles.

- Módulos: todos los módulos .bas para el proyecto. Sólo contienen código, al estilo de programación en Basic tradicional

- Módulos de clase: todos los archivos .cls del proyecto. Objetos creados por el programador con propiedades y métodos. Accesibles desde el examinador de objetos.

- Referencias: establece referencias a otros proyectos. Son las librerías que nos permites usar objetos y controles de otros proyectos y aplicaciones.

[1] Al ser PlanFal una macro no tenemos ficheros adicionales al documento Project, pero es importante ver las analogías con un entorno de desarrollo independiente. Las distintas partes de la macro se pueden exportar como ficheros independientes.

Como ya se explicó, un aspecto importante de la aplicación es su capacidad de inicializarse con información, si se parte de un proyecto previamente realizado con PlanFal. La información que se necesita se guarda en campos (columnas del diagrama de Gantt) que normalmente no se utilizan.

- ✓ Número1: horas de trabajo
- ✓ Numero2: porcentaje de reducción de días (curvas de días)
- ✓ Número3: porcentaje de reducción de horas (curvas de horas)
- ✓ Número4: posiciones de la estación-subestación
- ✓ Número5: turnos de trabajo
- ✓ Número6: número de la repetición del avión de comienzo de curva
- ✓ Número7: número de la repetición de referencia
- ✓ Número8: avión de fin de curva
- ✓ Text1: nombre
- ✓ Tex2: descripción
- ✓ Flag1: avión de comienzo de curva
- ✓ Flag2: avión especial
- ✓ Flag3: indicador de aplicación de curvas
- ✓ Flag4: indicador de enlaces especiales

Los campos numéricos 6, 7 y 8 sólo tienen sentido para el avión de comienzo de curva (avión con flag1 = true). El resto de columnas se tienen para todas las tareas (filas del diagrama de Gantt).

La información de las columnas en la vista de Gantt, realmente son propiedades de las tareas en Project. Para guardar la información relativa a los recursos (valor del menú opciones de operarios) se han utilizado propiedades del objeto Resource, de esta forma tenemos:

- ✓ ResourceNumber1: número de operarios por defecto
- ✓ ResourceNumber2: número de turnos por defecto
- ✓ ResourceMaxUnits: número máximo de operarios

Los módulos principales de PlanFal son los siguientes:

- Módulos de clase

  - **Avion:** consta de 19 métodos y 21 propiedades. Con los métodos gestionamos las propiedades e inicializamos las variables. En las propiedades, generalmente tipos single, integer y variant (usadas como matrices) se guardan toda la información importante de cada avión.

- Formularios de usuario

  - **FrmGeneral:** es el formulario principal de la aplicación. Tiene 8 métodos que gestionan los eventos principales del usuario. Existen dos métodos importantes, el de arranque del formulario (UserForm_Initialize) y el de ejecución de la planificación.

  - **FrmEstaciones:** ventana principal de entrada de datos. Consta de 7 métodos.

  - **FEstaciones:** formulario sin métodos que sirve como "contenedor" de controles para la creación de ventanas en tiempo de ejecución.

  - **FrmSubEstaciones:** formulario de entrada de datos para las subestaciones. Tiene 5 métodos.

  - **FsubEstaciones:** al igual que FEstaciones es un formulario sin eventos. Posee controles adicionales a FrmSubEstaciones que permiten ubicar una subestación dentro de la estación.

  - **FrmOperarios:** formulario diseñado para introducir datos de horas, días, curvas y operarios teniendo en cuenta la relación entre los datos. Tiene 15 métodos que gestionan los eventos del formulario.

  - **FrmOpciones:** formulario compuesto por 2 páginas (pages). Tiene 13 métodos. Los principales métodos se encargan de los eventos producidos por la opción de

configuración. Es el formulario de acceso a los formularios encargados de la gestión de aviones especiales.

- **FrmAvEspecial:** formulario principal de la gestión de aviones especiales. Tiene 8 métodos, la mayoría se encargan de proporcionar acceso a otras ventanas y preparar los datos que se van a presentar.

- **FrmCopiar:** formulario con 3 métodos. La complejidad de este formulario está en la presentación de datos a copiar.

- **FrmEnlacesEspeciales:** formulario con 10 métodos. Puede ser el formulario más complejo de todos, pues se encarga de mostrar los enlaces de los aviones especiales, y de dar al usuario la capacidad de modificarlos.

- **FrmAyudaEnlaces:** formulario sin métodos que presenta una ayuda en forma de texto al uso de formulario de enlaces especiales.

- Módulos:

El funcionamiento de PlanFal se puede dividir en tres etapas: inicialización de datos (arranque de la aplicación), entrada de datos (uso de formularios) y ejecución de la planificación (introducción de datos en Project). Los módulos están orientados a cada una de estas actividades.

1. Módulos encargados de la gestión de formularios:

- **GestFrmEstaciones**: consta de 7 métodos. Se encarga de todas las operaciones relacionadas con la creación y eliminación de estaciones así como de la actualización de las variables internas del programa.

- **GestFrmSubestaciones:** tiene 7 métodos, que se encargan de todas las operaciones relacionadas con la gestión del formulario de subestaciones.

2. Módulos encargados de la gestión de aviones, cálculos especiales y control:

- **GestAv:** tiene 10 métodos. La mayoría gestionan operaciones relacionadas con los aviones especiales. También tiene métodos que se encargan de cálculo de operarios, días y horas de trabajo.

- **Control:** se encarga del cálculo de datos, enlaces y en general contiene métodos que ayudan a otros métodos. Esta formado por 12 métodos

3. Módulos encargados de la inicialización de variables y arranque del programa:

- **Inicialización:** formado por 8 métodos es el encargado de gestionar el arranque del programa, inicializando todas las variables internas con la información obtenida de Project.

- **Demo:** formado por un método que proporciona valores demostrativos. Puede ser utilizado como un punto de partida para la planificación real.

- **Inicio:** módulo cuya única misión es hacer accesible PlanFal desde Project como macro, y por lo tanto comenzar su ejecución.

4. Módulos encargados de la ejecución del programa:

- **Interno:** gestiona la ejecución y el progreso del programa. Tiene un método.

- **Tareas:** formado por dos métodos, se encarga de "escribir" las tares en Project, dotándolas de jerarquía y con las características adecuadas

- **Duración:** realiza todos los cálculos relacionados con la curva de aprendizaje y los operarios. Asigna las duraciones a las tareas. Esta formado por 5 métodos.

- **Vinculación:** se encarga de la vinculaciones entre las tareas. Esta formado por 7 métodos.

9. CONCLUSIONES Y LÍNEAS FUTURAS

9.1. INTRODUCIÓN

El desarrollo de herramientas que ayudan a reducir tiempo en el trabajo es fundamental para conseguir una mayor productividad. Porque al facilitar la labor del ingeniero, este puede dedicarse a labores de investigación, análisis de resultados y en general a cuestiones relacionadas con la capacidad creativa y no a realizar tareas rutinarias y mecánicas.

Aun siendo el proyecto un desarrollo software, se ha intentado realizar un análisis general de las necesidades informáticas de una empresa de ingeniería y del trabajo asociado a la gestión de proyectos. Tras este análisis se observa que los sistemas globales son los que mejor satisfacen y se adecuan a la labor de ingeniería. La ingeniería concurrente y la división del trabajo obliga a tener un sistema de gestión y control de la información, esto se traduce en interfaces transparentes y automáticas entre las distintas aplicaciones informáticas que se necesitan en la ejecución de un proyecto.

Las ventajas de disponer de un sistema informático adaptado a las necesidades concretas de la empresa son obvias. Es frecuente el uso de herramientas software "genéricas" que tras una labor de customización responden mejor a las necesidades del cliente. Hoy en día existen muchas empresas que realizan software con soluciones para la industria y la empresa, pero con un gran margen de "adaptabilidad", siendo precisamente esta característica su principal virtud.

Siendo la planificación una de las primeras etapas de un proyecto, el estudio de distintos escenarios tiene un papel fundamental. La capacidad de análisis se ve limitada si para obtener resultados se necesitan muchos recursos o tiempo. PlanFal da una solución a la necesidad de optimizar el software del que se dispone para conseguir, principalmente, un ahorro de tiempo, de ahí su importancia en esta etapa del Proyecto del A400M

Otra consecuencia implícita en la automatización de tareas es la productividad del trabajador. Si en algo nos distinguimos de las máquinas es en la capacidad creativa y de razonamiento. Un trabajo repetitivo agota y mata la creatividad. Para este tipo de tareas se crearon las máquinas, que no sienten ni fatiga ni aburrimiento. Evidentemente el proceso de automatizar tareas y desarrollar nuevas herramientas software conlleva un trabajo y tiempo asociado. Es por lo tanto importante analizar hasta que punto es rentable, puesto que no siempre será la mejor opción.

9.2. VENTAJAS OBTENIDAS

A continuación vamos a enumerar las principales ventajas obtenidas con este proyecto:

1. Ahorro de tiempo.

   Antes de la realización de este proyecto, el tiempo estimado para la realización de una escalera de aviones era de 15 a 20 días. El tiempo que se necesita con PlanFal no supera la hora.

2. Capacidad de análisis y supuestos.

- Necesidades crecientes.

  A medida que se van definiendo con más detalle los procesos que intervienen en el ensamblaje, es necesario ajustar y modificar los tiempos de las tareas, así como detallar los recursos asociados a las mismas. Los plazos de entrega y pedidos de

materiales son otro aspecto importante que se tienen que concretar a medida que un proyecto avanza en el tiempo. Todo este tipo de resultados implican ajustar y modificar planificaciones bases con las que se trabaja. PlanFal permite este tipo de modificaciones de forma rápida, ya que disponemos de la opción de partir de planificaciones existentes.

- Posibilidad de estudiar distintos escenarios.

Al poder realizar planificaciones en poco tiempo, podemos comparar resultados trabajando con distintos escenarios. Un caso muy frecuente es el que tenemos con el LEAD-TIME o tiempo mínimo. Si queremos reducir el plazo entre entregas tenemos que reducir la estación crítica. Para ello caben distintas alternativas, por ejemplo duplicar estaciones. Con PlanFal obtener una planificación con duplicación de la estación crítica partiendo de una preexistente sin duplicación es casi automático.

3. Punto de partida para planificaciones más detalladas.

La planificación es en definitiva una cuestión de sentido común, pues se trata de obtener la mejor forma de hacer las cosas dadas una restricciones y unos objetivos. Existen muchas alternativas y opciones que se escapan a PlanFal. Si bien no se puede pretender que el resultado de PlanFal nos de la planificación definitiva, si que es una importante base para trabajar. Al haberse hecho la escalera con un gran número de vinculaciones (que de forma manual hubiese sido casi imposible), las posibles modificaciones son más fáciles, puesto que implican cambios automáticos en las tareas vinculadas.

4. Utilización de una herramienta estándar en la empresa y a nivel mundial.

Al ser PlanFal una macro de Project podría pensarse que estamos limitando su uso al uso de Project y perdemos la libertad de las aplicaciones autónomas. Veamos a continuación las ventajas que se derivan de la programación de PlanFal como macro.

- Facilidades en el intercambio de planificaciones.

- Conocimiento de Project por parte de los trabajadores
- Aplicación de uso extendido y ya instalada en la empresa
- No necesita instalación en los ordenadores.
- No necesita ficheros adicionales.
- Perfectamente integrada en Project.
- No necesita de recompilación ante posibles modificaciones
- En sistemas en red la sustitución de la plantilla global por el fichero con la Macro permite el uso general de la aplicación.

5. Conocimiento profundo de Project y de Visual Basic.

El conocimiento de los objetos de Project a un nivel de Programación abre un gran número de posibilidades para todo tipo de añadidos adicionales que se quieran hacer a Project. El hecho de que Project pertenezca a la familia Microsoft resulta además muy interesante, pues todas sus aplicaciones presentan un entorno de desarrollo similar. El dominio de Visual Basic para aplicaciones abre nuevos camino al uso conjunto de aplicaciones como Word, Excel, Access, PowerPoint y Project, con todo el potencial que ello implica.

9.3. LÍNEAS FUTURAS

El uso de las herramientas de programación de algunos entornos es en sí mismo una puerta abierta, ya que nos ofrece un camino para encontrar soluciones a los problemas que se nos presentan.

Uno de los aspectos que se cuidó en la realización del programa fue su capacidad de crecer. Para ello se hizo un esfuerzo en realizar una programación muy modular y con funciones genéricas. Esto permitiría modificar y añadir nuevas funcionalidades al programa sin grandes cambios en su estructura. Es esta capacidad de la aplicación una de sus principales líneas de avance, puesto que fueron las fases de depuración y de customización las que se vieron más afectadas por la limitación de terminar la aplicación en un tiempo fijado. Una posible mejora a PlanFal sería la posibilidad de dotarlo de facilidades adicionales que ayuden a la planificación o generación de

documentos partiendo de la información obtenida con la escalera y nuevas opciones e interfaces en función de ideas aportadas por los usuarios.

Al ejecutar el programa nos encontramos con un comportamiento "peculiar" de Project, este se vuelve más lento a medida que aumenta el número de avión que se ensambla. Esto puede llegar a ser un problema en ordenadores lentos, e incluso hacer inestable el sistema informático. Sería interesante estudiar porqué ocurre esto y encontrar una solución para hacer más rápida la ejecución del programa. Una posible solución (habría que estudiarla en profundidad) sería dotar a PlanFal de la capacidad de "enlazar" escaleras, por lo que una escalera de 100 aviones se podría realizar como la "unión" de 10 series de 10 aviones. Esta solución implica modificar la inicialización y la ejecución de PlanFal. Para no complicar la interfaz gráfica, la solución pasaría por ser transparente para el usuario, teniendo este, en todo caso, la posibilidad de especificar el número de aviones que componen cada serie.

Existen tareas que sin la utilización de macros se convierten en muy complicadas o imposibles de realizar. Un ejemplo real que se presentó al margen de PlanFal y durante el desarrollo del mismo fue la necesidad de calcular fechas a partir del comienzo de una estación determinada. Era importante, para fijar pedidos a los proveedores, saber las fechas que se correspondían con 10 días antes del comienzo de la estación 5. El problema radicaba en la necesidad de distinguir ente festivos y laborables. No se encontró la forma de hacerlo satisfactoriamente ni en Excel ni en Project en su forma estándar, y el cálculo manual era una tarea lenta y repetitiva, con posibilidad de cometer errores. Los conocimientos de Project y Visual Basic ayudaron a la realización de una macro (figura 9.1 y anexo III) que permite el cálculo automático de fechas a partir de una columna de Project, mediante un criterio de inserción y búsqueda de fechas, pudiendo configurar el retraso o adelanto (números negativos) en días y la posibilidad de tener en cuenta los días festivos como días no válidos.

Este es un claro ejemplo de la potencialidad que ofrece el uso de macros y de posibles funcionalidades que se pueden añadir a PlanFal.

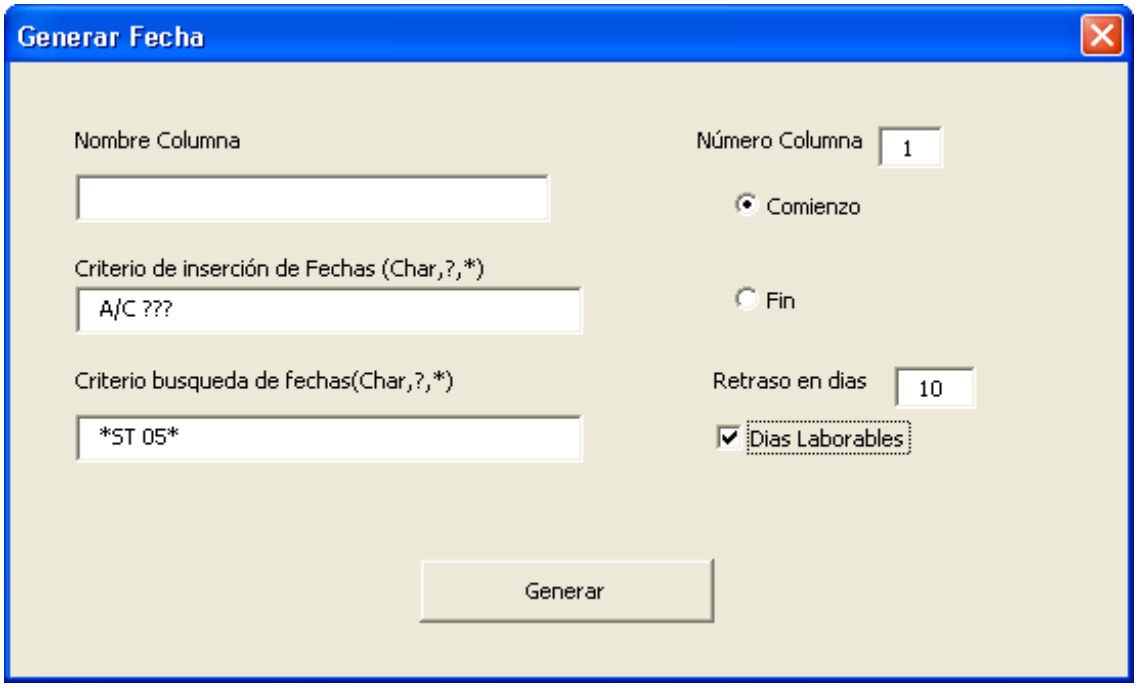


Figura 9.1. Macro que facilita el calculo de fechas.

ANEXOS

ANEXO I

MANUAL BÁSICO DE MICROSOFT PROJECT 98

I.1. INTRODUCCIÓN

El objetivo de este capítulo es comprender el funcionamiento básico de la herramienta de planificación Microsoft Project. Utilizaremos la versión MS Project 98, por ser la versión que se utilizó durante el desarrollo de la aplicación.

MS Project es el software de planificación más extendido en el ámbito empresarial. La mayoría de las empresas software que desarrollan soluciones para la gestión de proyectos y planificación son capaces de importar y exportar datos a Project. Las nuevas versiones de Project tienden a incrementar la capacidad de gestionar multiproyectos y de administrar información mediante acceso a una base de datos.

No se pretende explicar todas las posibilidades del programa Microsoft Project al detalle, sino dar las nociones básicas para entender el funcionamiento de esta herramienta de ayuda a la planificación y seguimiento de proyectos. Para una mayor comprensión, o ante dudas que no queden resueltas, el programa dispone de una ayuda muy completa.

El programa es útil en dos etapas diferentes: la planificación original del proyecto y su seguimiento. En este manual también se incluye una apartado que explica cómo se realiza la presentación de los resultados obtenidos. Por último se hará una pequeña introducción al uso de macros y cómo acceder al módulo de programación en Visual Basic.

I.2. INFORMACIÓN GENERAL SOBRE EL PROGRAMA

Antes de comenzar a trabajar con Microsoft Project es conveniente familiarizarse con algunas de sus características.

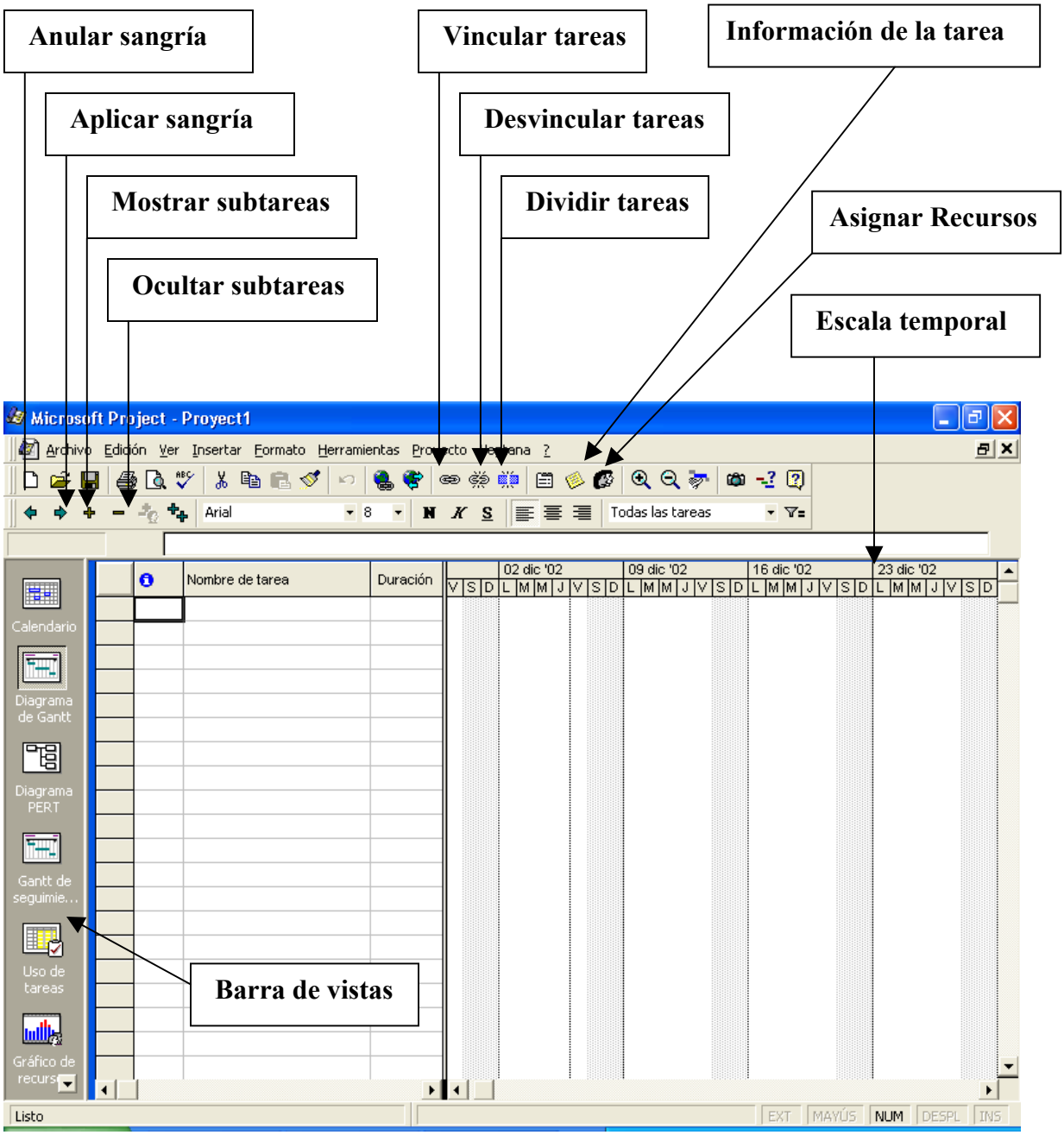


Figura I.1. Aspecto general de la ventana de Microsoft Project.

Para rellenar (al principio) o consultar información general o estadísticas del proyecto tenemos la ficha **Propiedades** en el menú **Archivo** o la ficha **Información del proyecto** en el menú **Proyecto**.

El programa nos mostrará distintas vistas o presentaciones de la información, como un calendario, diagrama de Gantt, etcétera. Para elegir la vista que queramos sólo tenemos que hacer clic en el icono correspondiente de la **Barra de vistas**, que es una barra vertical a la izquierda de la pantalla. Algunas vistas tienen una parte con un aspecto parecido a una hoja de cálculo, en que cada fila corresponde a una tarea y en cada columna se facilita un tipo de información sobre la tarea. Podemos elegir qué información se representa en esta vista de hoja seleccionando, en el menú **Ver->Tabla**, y eligiendo la tabla que se quiera aplicar. También podemos añadir una columna en particular mediante **Insertar->Columna**. El cambio de vista no añade ni quita información del proyecto, sólo cambia el tipo de información que se muestra en pantalla.

En muchas vistas suele haber a la derecha un eje temporal horizontal, en el que se representa gráficamente información sobre las tareas. Podemos cambiar la escala de este eje haciendo doble clic sobre él y rellenando el cuadro que aparece.

La primera vez que se guarda el archivo nos preguntará si queremos guardar una línea de base. Debemos contestar que no a menos que ya tengamos la planificación original del proyecto completamente terminada, o sea, cuando hayamos completado todos los puntos del apartado 3 de este manual.

Por último, señalar que en la parte superior de la pantalla aparece una barra de herramientas horizontal, a la que nos referiremos más adelante.

I.3. PLANIFICACIÓN

I.3.1. Establecer la fecha de comienzo

El primer paso al abrir un nuevo proyecto es decidir la fecha de comienzo (la fecha de fin quedará determinada por la información que se vaya añadiendo).

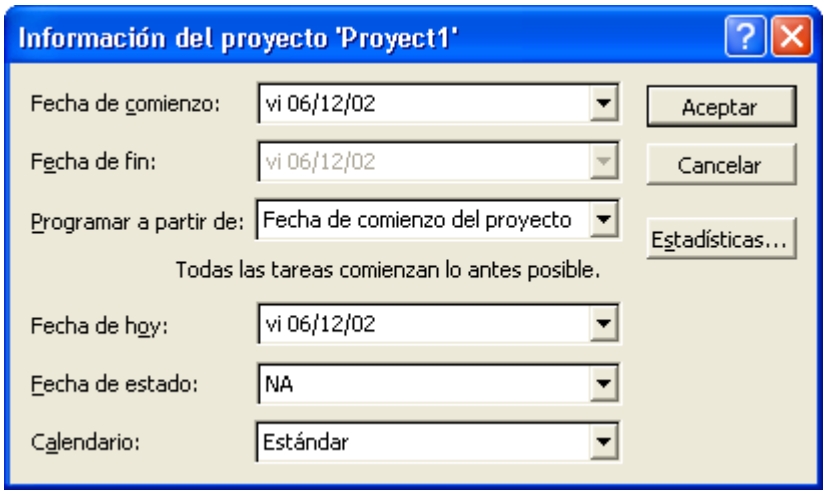


Figura I.2. Ventana principal de información del proyecto.

I.3.2. Personalizar el calendario laboral

A continuación debemos personalizar el calendario laboral, para determinar de cuánto tiempo en total disponemos realmente. Para ello, en el menú **Herramientas**, seleccionamos **Cambiar calendario laboral**. El calendario se puede cambiar para cada día de la semana. Por ejemplo, si los lunes estamos ocupados mañana y tarde, desde el punto de vista de proyecto conviene señalar el lunes como no laborable. Para ello hacemos clic en "lunes" y en "Periodo no laborable": todos los lunes del calendario figurarán como no laborables. Si disponemos de la mañana de los domingos para dedicarle al proyecto conviene hacer clic en "D", "Periodo laborable" y señalar las horas de trabajo. Una vez determinados todos los días de la semana en general, se pueden señalar excepciones. Por ejemplo, si queremos tomamos la semana de feria de vacaciones seleccionamos los días de esta semana en particular y los señalamos como no laborables. De esta manera el administrador del proyecto podrá distribuir el trabajo como convenga a los componentes del grupo. (El programa permite asignar un calendario personalizado a cada trabajador, pero con un calendario general que dé una idea del número de horas a la semana disponibles para el proyecto es suficiente). Por último, pulsar **Aceptar**.

Figura I.3. Ventana para personalizar el calendario laboral

I.3.3. Dividir el proyecto en tareas

El siguiente paso es hacer una lista de las tareas de que se compone el proyecto. Una tarea debe ser lo suficientemente concreta como para que se pueda saber cuándo ha acabado. El número de tareas no debe ser demasiado grande ni demasiado pequeño, pero dependerá del proyecto concreto, y hay que tener en cuenta que pueden dividirse en subtareas. Pulsando **Diagrama de Gantt** en la **Barra de Vistas**, aparece un cuadro donde se puede escribir el **Nombre de tarea** y su **Duración**, que puede estar dada en días, horas, etc. La estimación de la duración de cada tarea debe ser conservadora. En cualquier momento se podrá modificar la duración de una tarea o cualquier otro dato. No hay que rellenar los campos de Comienzo y Fin, pues el programa los calculará.

A la derecha en esta vista aparece el diagrama de Gantt propiamente dicho: un eje temporal en el que cada tarea se representa por una barra azul de longitud proporcional a su duración. En este diagrama también quedarán representados los vínculos que se establezcan entre tareas.

Si hay tareas que se repitan cada cierto tiempo, como reuniones semanales, se hace así: en el campo **Nombre de tarea** se selecciona una fila. Del menú **Insertar** se selecciona **Tarea repetitiva** y se rellena el cuadro.

La lista de tareas se puede reorganizar: para seleccionar una tarea se hace clic en el número identificador de la tarea (para seleccionar un grupo de tareas mantener pulsada CTRL) y se usa **Cortar, Copiar y Pegar** del menú **Edición**. Si la fila seleccionada de destino ya está rellena, la información pegada se insertará justo antes de esta fila. También se puede **Insertar->Nueva tarea**.

Puede que sea conveniente dividir una tarea en subtareas. La tarea de resumen aparece en negrita y las subtareas que la componen aparecen debajo de ella con sangrías. En el **Diagrama de Gantt**, podemos seleccionar tareas en el campo **Nombre de tarea** y hacer clic en **Aplicar sangría** o **Anular sangría** de la barra de herramientas. También se pueden pulsar **Mostrar subtareas** y **Ocultar subtareas** (ver figura 1).

Otra posibilidad interesante consiste en dividir una tarea en segmentos, útil cuando conviene interrumpir temporalmente el trabajo en una tarea para dedicarse a otra. Para conseguirlo, en el **Diagrama de Gantt**, hay que hacer clic en **Dividir tarea** (ver figura I.1), colocar el puntero en el punto de la barra de la tarea elegida donde se quiera cortar, y volver a hacer clic. Esto provocará un aplazamiento de una unidad de la escala de tiempo. Para conseguir un retraso mayor, al hacer clic sobre la barra de tarea, antes de soltar el botón, arrastrar la barra hacia la derecha. Para eliminar una división, arrastrar un trozo de la barra hasta que entre en contacto con el otro trozo.

En un proyecto, además de las tareas, suele ser necesario establecer "hitos", puntos relevantes en la evolución del proyecto, que puede que se deban alcanzar en una fecha determinada. En este programa los hitos se definen como tareas de duración 0.

I.3.4. Establecer las relaciones entre tareas

Una vez terminada la lista de tareas, hay que establecer los vínculos entre ellas. Este programa permite muchos tipos de relaciones entre tareas, por ejemplo "fin a

comienzo", es decir, una tarea no puede empezar hasta que no acabe otra. Para ordenar varias tareas de forma secuencial, en **Diagrama de Gantt** se seleccionan estas tareas en el campo Nombre de tarea en el orden en que se tengan que realizar, y se hace clic en **Vincular tareas** (ver figura 1). Por defecto el vínculo que se establece es de fin a comienzo. Si se quiere otro tipo de vínculo se hace doble clic en la línea que representa el vínculo y se rellena el cuadro **Dependencia entre tareas** (figura I.4). Por ejemplo, si se quiere que la segunda tarea empiece 1 día más tarde que la primera habrá que establecer una relación "comienzo a comienzo" con una posposición de 1 día. Si la posposición es negativa en lugar de un retraso habrá un adelanto.

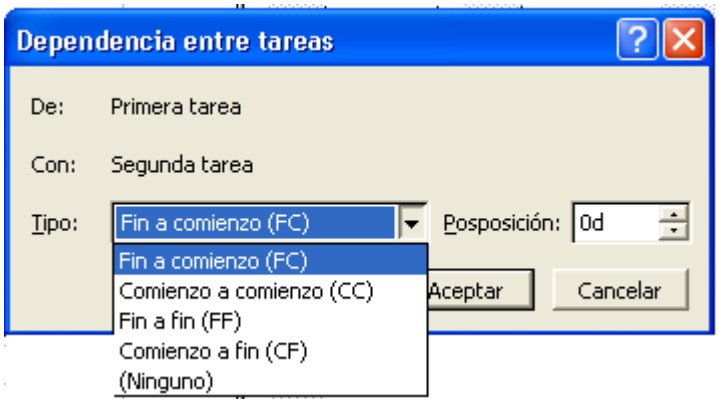


Figura I.4. Cuadro de dependencia entre tareas.

Para eliminar vínculos se seleccionan las tareas implicadas y se pulsa **Desvincular tareas**.

Normalmente el procedimiento más eficaz es especificar las duraciones de las tareas, y el programa se encarga de calcular las fechas de comienzo y fin en función de las vinculaciones establecidas. Sin embargo puede que interese imponer delimitaciones de otro tipo, por ejemplo "no comenzar antes del", "no finalizar después del", etcétera. Para ello seleccionamos un **Nombre de tarea** y pulsamos **Información de la tarea**. En la ficha **Avanzado** rellenamos el cuadro **Delimitar tarea** (figura I.5). Si imponemos demasiadas restricciones se pueden crear conflictos, el programa nos avisará a tiempo para que eliminemos alguna de las restricciones. Por ejemplo, un conflicto común se da cuando programamos a partir de la fecha de comienzo del proyecto y queremos imponer como restricción que el proyecto debe estar terminado antes de una determinada fecha:

es mejor no imponer esta restricción, y que el administrador del proyecto tenga en cuenta si se va a terminar el trabajo a tiempo o no. Lo mismo sucede con los hitos.

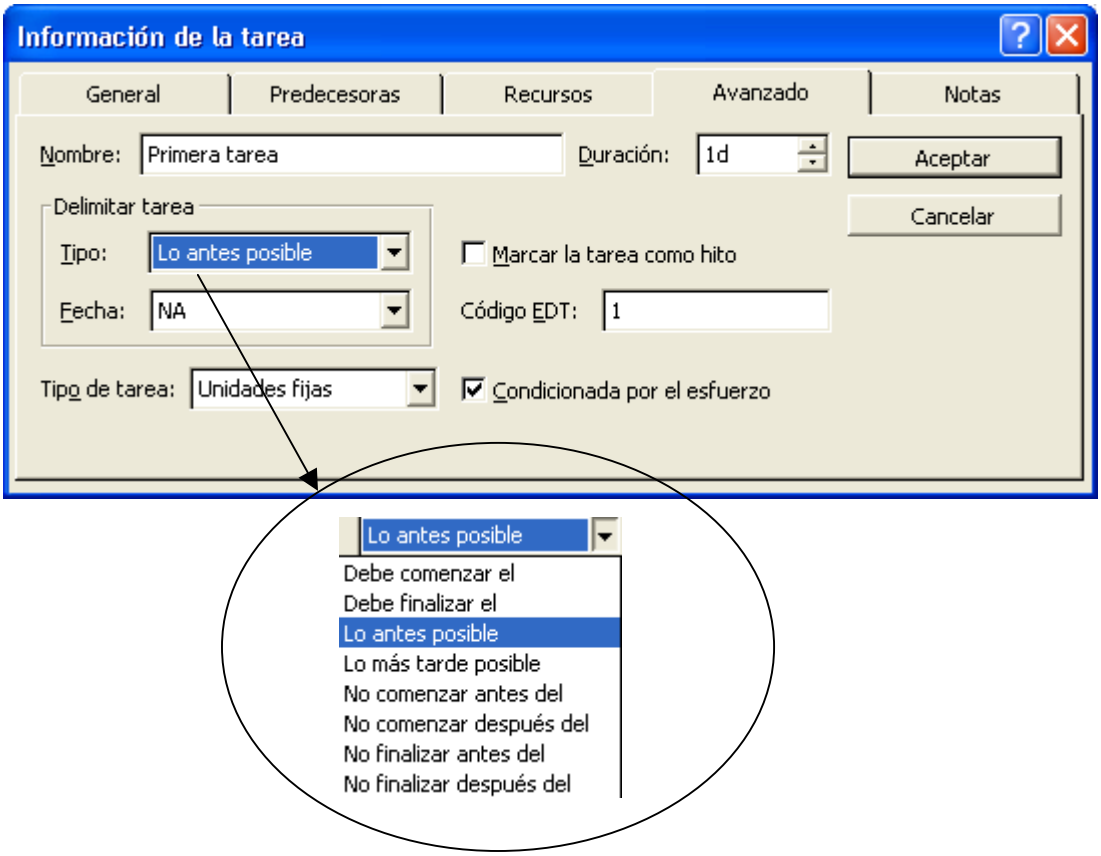


Figura I.5. Delimitación de tareas y ventana de información avanzada.

Hay que recordar que toda la estructura de tareas es flexible: por ejemplo, si se cambia la estimación de la duración de una tarea, se reajusta automáticamente el gráfico de todas las tareas que le siguen.

Cuando consideremos que hemos terminado de establecer todas las relaciones entre tareas, tenemos que comprobar que el proyecto termina en su límite.

Si sobrepasamos la fecha límite o queremos dejar un margen de seguridad mayor del que ha resultado, tenemos que hacer cambios en las delimitaciones, hasta que consigamos un plan razonable para el proyecto. Para hacerlo es conveniente identificar la ruta crítica, que está formada por las tareas críticas, que son las que no se pueden retrasar sin que se modifique la fecha de fin del proyecto. Para averiguar cuáles son las

tareas críticas el método más inmediato es seleccionar en la Barra de Vistas la vista **Diagrama Pert**, que es un diagrama de bloques en el que cada tarea es un bloque, los vínculos entre tareas se representan con flechas, y las tareas críticas aparecen resaltadas en rojo.

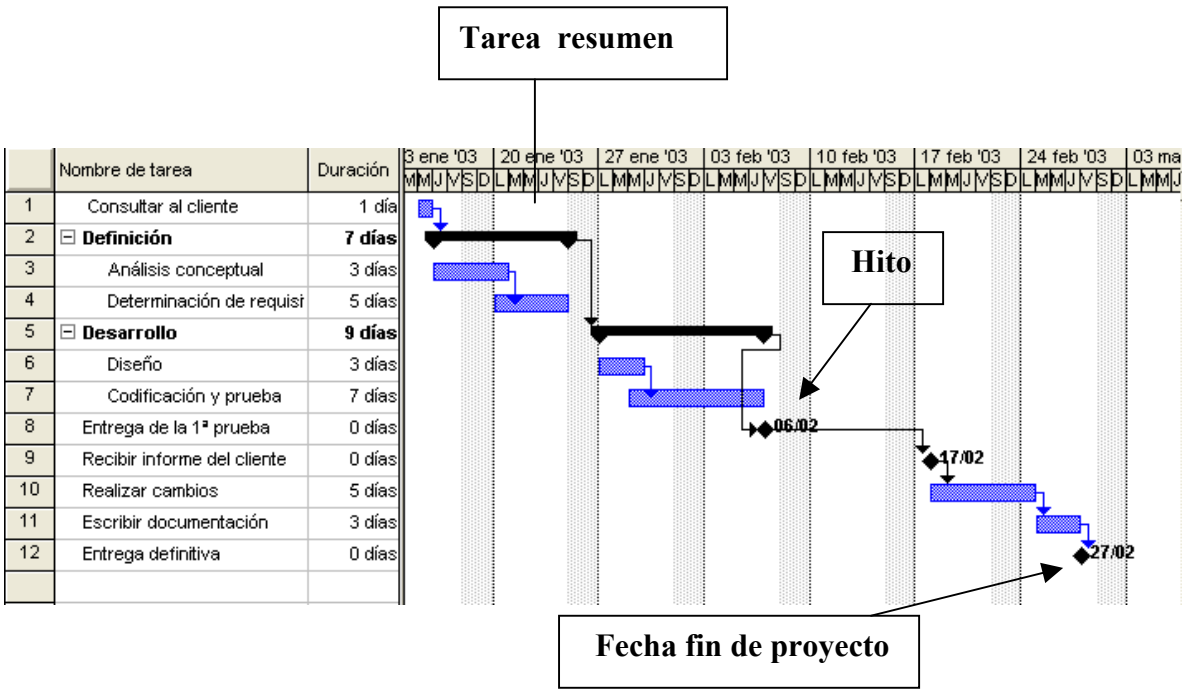


Figura I.6. Vista Diagrama de Gantt del Proyecto de ejemplo.

Puede que sea conveniente dejar los cambios del diagrama de Gantt para cuando estén asignados los recursos, ya que la capacidad de estos también determinará hasta qué punto se pueden superponer y concentrar tareas en el tiempo.

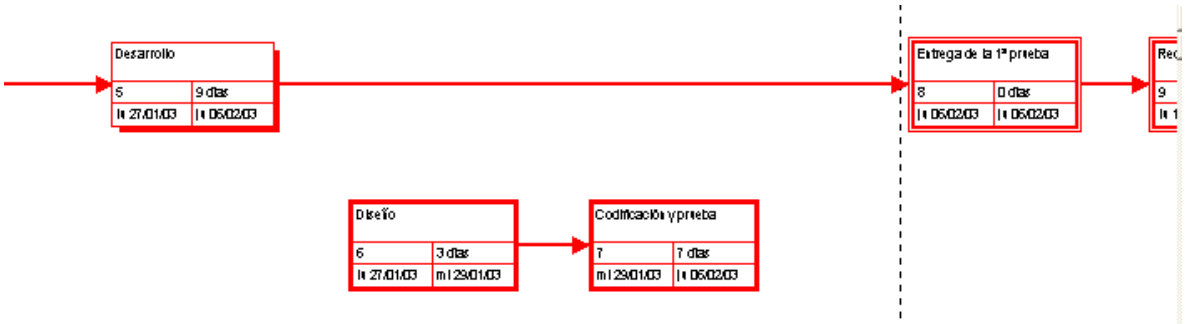


Figura I.7. Vista Diagrama de Pert de Proyecto de ejemplo.

I.3.5. Asignación de recursos

Una vez definidas todas las tareas que componen el proyecto y las relaciones entre ellas, el siguiente paso es asignar recursos para llevar a cabo estas tareas. Si se dispone de un equipo fijo de trabajadores, lo más eficaz es crear una lista de recursos. Para ello, en la **Barra de Vistas,** seleccionamos **Hoja de recursos**. En el menú **Ver**, seleccionamos **Tabla- >Entrada.** y rellenamos **Nombre del recurso** y **Capacidad máxima**. La capacidad máxima indica el número de unidades disponibles de ese recurso en forma de porcentajes. Por ejemplo, para el recurso "ingeniero", 100 podría significar 1 ingeniero a tiempo completo, 2 ingenieros al 50 del tiempo, etc. Dos ingenieros a tiempo completo se indicarían con 200. etc. En muchas ocasiones, para ajustarse lo más posible a la realidad, lo más práctico es indicar en la lista los nombres de los componentes del equipo, cada uno al 100, para asignarles tareas concretas.

Antes de asignar recursos a las tareas conviene especificar el tipo de cada tarea, ya que puede que al variar unos parámetros el programa cambie automáticamente otros, por ejemplo si se duplican los recursos la duración de la tarea se reduce a la mitad, para que siempre se cumpla la ecuación

$$\text{Duración} = \frac{\text{Trabajo}}{\text{Recursos}}$$

Microsoft Project ofrece tres tipos de tareas: unidades fijas, trabajo fijo y duración fija. En general se recomienda utilizar el tipo de tarea que el programa trae por defecto, que es el de unidades fijas, para que el programa no nos modifique la cantidad de personas y tiempo que consideramos disponible para realizar una tarea. Lo que sí hay que especificar es si la tarea es condicionada por el esfuerzo o no. es decir, si su duración va a depender o no de la cantidad de recursos que estén asignadas a ella. Por ejemplo, una tarea como imprimir una determinada información no va a durar menos porque haya dos personas encargándose de ella. Para indicar esto al programa, hacemos doble clic en el **Nombre de tarea**, seleccionamos la ficha **Avanzado** y activamos o desactivamos la casilla **Condicionada por el esfuerzo**.

Para asignar recursos a las tareas, en **Diagrama de Gantt**, seleccionamos un **Nombre de tarea** y pulsamos **Asignar recursos**. Hay que rellenar el Nombre del recurso y el número de unidades en % o en decimales (esta opción es configurable en **Herramientas->Opciones->Programación->Mostar las unidades de asignación**) , como se acaba de explicar. Si es una tarea condicionada por el esfuerzo, hay que tener cuidado porque el programa entiende que la duración que hemos especificado corresponde a un solo tipo de recurso al 100 de capacidad. Si fuera necesario, habrá que corregir el campo Duración de la tarea.

Una vez que tenemos el cuadro **Asignar recursos** abierto, no hace falta cerrarlo y volverlo a abrir para cambiar de tarea, sólo hay que seleccionar un nuevo **Nombre de tarea**.

I.3.6 Garantizar que la planificación original es viable

Para completar la planificación original sólo falta comprobar que la fecha de finalización del proyecto es aceptable y que los recursos no estén sobre asignados. Para verlo podemos seleccionar la vista de recursos en la Barra de Vistas. En esta vista, a la izquierda aparecen una serie de columnas con el nombre del recurso, las horas totales de trabajo, etc (podemos seleccionar la información que queremos ver en el menú **Ver->Tablas->...**). A la derecha tenemos un calendario en el que aparecen las horas de trabajo de cada recurso según la fecha.

		Nombre del recurso	Trabajo	Detalles	20 ene '03					
					J	V	S	D	L	M
1		Ingeniero software	208 horas	Trabajo	8h	8h			16h	8h
		Análisis conceptual	*24 horas*	Trabajo	8h	8h			8h	
		Determinación de requisitos	*40 horas*	Trabajo					8h	8h
		Diseño	*24 horas*	Trabajo						
		Codificación y prueba	*56 horas*	Trabajo						
		Realizar cambios	*40 horas*	Trabajo						
		Escribir documentación	*24 horas*	Trabajo						
2		Comercial	8 horas	Trabajo						
		Consultar al cliente	*8 horas*	Trabajo						
		Entrega de la 1ª prueba	*0 horas*	Trabajo						
		Recibir informe del cliente	*0 horas*	Trabajo						
		Entrega definitiva	*0 horas*	Trabajo						

Figura I.8. Vista Uso de Recursos .

Otra manera de ver la distribución de trabajo es con la **vista Gráfico de recursos** (figura I.9). En esta vista, para cada recurso tenemos un eje horizontal de tiempos, sobre el que se representa, en forma de columnas, la cantidad de trabajo que tiene asignado un recurso en cada unidad de tiempo. Esta columna será azul a menos que el recurso esté sobre asignado, en cuyo caso la fracción de trabajo que sobrepasa la capacidad del recurso estará en rojo.

Si hay recursos sobre asignados habrá que hacer cambios en la planificación del proyecto. Para ello nos pueden ayudar estas dos vistas, ya que permiten saber si la distribución de tareas entre los trabajadores ha sido equitativa, o si a algún recurso determinado se le acumulan las tareas en un periodo demasiado corto de tiempo. El Gráfico de recursos es especialmente útil en este sentido ya que permite ver rápidamente si cada recurso tiene el trabajo más o menos uniformemente repartido a lo largo del tiempo disponible para realizar el proyecto.

Es conveniente que la planificación original de todo proyecto contemple la distribución equitativa de tareas entre los recursos y en el tiempo, dejando un margen de seguridad antes de la fecha límite de presentación del proyecto. Este margen dependerá de la importancia y los perjuicios que pueda ocasionar un retraso en el inicio del proyecto o cualquier causa inesperada, como retrasos en las entregas o en las tareas, influyendo especialmente las de la ruta crítica.

Para repartir mejor una serie de tareas que por estar acumuladas en el tiempo sobrepasan la capacidad de los recursos, se podrían dividir tareas en segmentos, aumentar los tiempos de posposición en tareas que se superpongan, añadir delimitaciones del tipo "no comenzar antes del", etc. Para hacer esto de forma sistemática puede ser muy útil, en la **Barra de Vistas**, **seleccionar Más vistas->Asignación de recursos**, y **Aplicar**. La pantalla quedará dividida en dos partes: la superior es la ya comentada vista **Uso de recursos** y la inferior es la vista **Gantt de redistribución**, en la que aparecen las tareas que ocurren en fechas sobre asignadas.

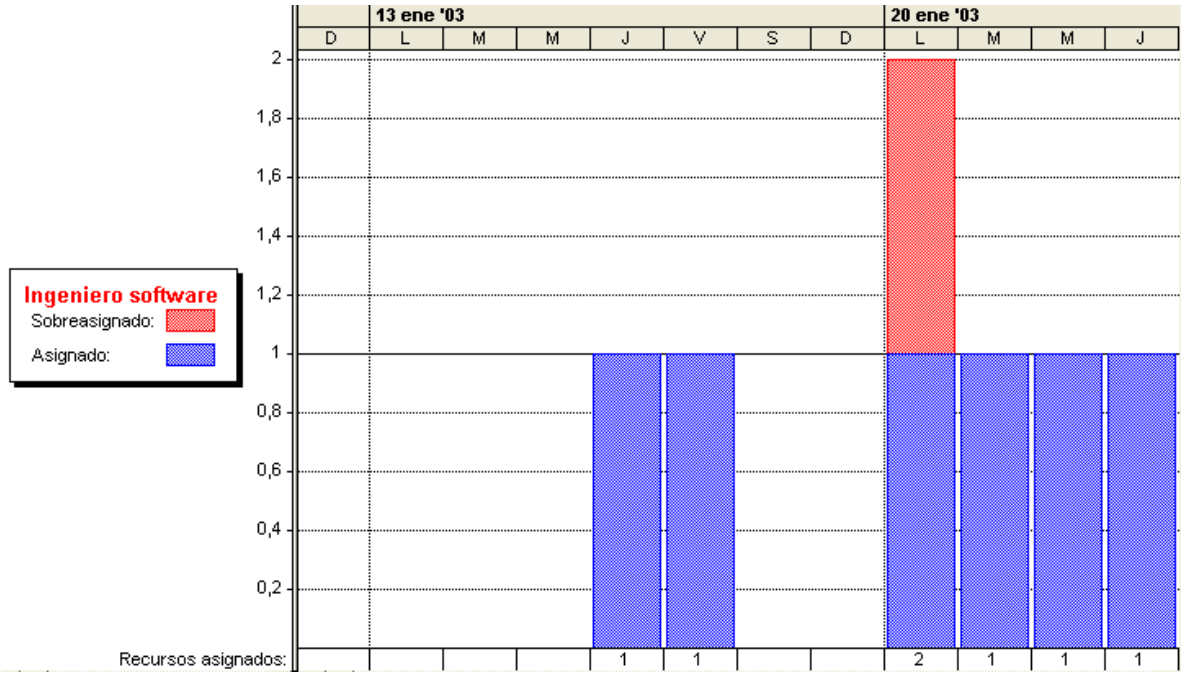


Figura I.9. Vista Gráfico de recursos .

En el eje de tiempos aparece para cada tarea una barra que representa el margen de demora, o sea el máximo retraso que puede tener la tarea sin que afecte a las tareas sucesoras o a la fecha de fin del proyecto. Podemos rellenar el campo **Retraso por redistribución** con el retraso que queremos aplicar a la tarea para conseguir que el trabajo se reparta mejor en el tiempo. Para esto puede ser bastante más útil seleccionar para la parte superior la vista **Gráfico de recursos** (simplemente haciendo clic en su icono), ya que muestra, para un recurso determinado, cuándo tiene más tiempo libre, de manera que se pueden elegir los retrasos por redistribución de las tareas de forma inmediata.

Microsoft Project también ofrece la posibilidad de resolver la asignación de recursos de forma automática, que puede ser muy útil cuando la cantidad de tareas y de problemas de sobre asignación es muy grande, ero que puede tener el inconveniente de que nos lleve a posibilidades no deseadas. Puede consultarse más información sobre este punto en la ayuda del programa.

I.3.7 Guardar una línea de base

Si se han superado los puntos anteriores, se tiene un plan previsto optimizado para el proyecto, y ya se está en condiciones de comenzar la realización del mismo. En este momento se debe guardar una línea de base, que es el plan original, que podrá ser comparado posteriormente con el desarrollo real del proyecto. Seleccionamos en el menú **Herramientas->Seguimiento->Guardar línea de base**, y hacemos clic en **Proyecto completo**, (a menos que queramos añadir nuevas tareas a una línea de base existente, en cuyo caso tenemos que hacer clic en Tareas seleccionadas). Con esto queda terminada la planificación original. Ya se puede comenzar la realización del proyecto, para cuyo seguimiento y control seguirá siendo útil el programa.

I.4. SEGUIMIENTO

I.4.1. Objetivo del seguimiento

Una vez que se ha comenzado la realización del proyecto, es conveniente hacer un seguimiento de su desarrollo, para comprobar si se va cumpliendo el plan previsto. En la vida real normalmente se producen retrasos en el comienzo o fin de determinadas tareas, por eso es importante detectar lo antes posible cuáles tareas son las que se desvían del plan original, para hacer reajustes en el plan de trabajo y así evitar que las desviaciones afecten a la fecha de finalización del proyecto. Hay que prestar especial atención a las tareas de la(s) ruta(s) crítica(s).

I.4.2. Introducir datos reales

La vista más útil para el seguimiento de un proyecto es el **Diagrama de Gantt de seguimiento**. Cuando se ha seleccionado esta vista, conviene seleccionar en el menú **Ver->Tabla-> Variación**, para que en la hoja de la izquierda aparezcan para cada tarea los campos **Comienzo y Fin** actuales y previstos, y **la Variación** del comienzo y del fin, que es el valor real menos el valor actual. Para que aparezca esta información de variación es necesario que se haya guardado al menos una línea de base, y, por supuesto, que se hayan introducido los valores actuales que ya se conozcan en los

campos Comienzo y Fin. Los valores actuales no tienen por qué ser hechos reales, pueden ser las estimaciones más fiables según las circunstancias actuales.

Cuando tenemos datos reales, para introducirlos hay que seleccionar el **Nombre de tarea** que se quiera actualizar y **Herramientas-> Seguimiento->Actualizar tareas**. La ficha **Actualizar tareas** se puede rellenar de varias maneras diferentes: escribir la fecha real de comienzo y/o fin, rellenar los campos **Duración real** y **Duración restante** (que se refieren a la cantidad de tiempo que ya se ha dedicado y la que falta por dedicar a una tarea, y que sólo tiene sentido rellenarlos en el caso de tareas no condicionadas por el esfuerzo), o escribir el porcentaje del tiempo total de la tarea que ya se ha completado. De estas posibilidades, podemos actualizar la información que directamente tengamos, y el programa se encargará de recalcular los otros datos. Por ejemplo, si tenemos una tarea que dura dos horas, en principio "duración real" vale 0 y "duración restante" vale 2h. Si actualizamos la tarea rellenando el campo "% completado" con 25%, el campo "duración real" se actualizará a 0,5 h y "duración restante" a l,5h.

Cuando una tarea está completada, en el campo "indicadores" aparecerá un símbolo de verificación:

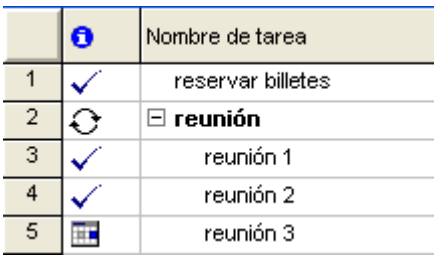


Figura I.10. Indicadores de las tareas.

I.4.3 Planes provisionales

Hay que notar que las actualizaciones de tareas cambian los datos reales, sin afectar a los datos previstos, que corresponderán a la línea de base que se ha guardado. Sólo debe existir una línea de base, que corresponde a la planificación original, pero se pueden guardar hasta 10 planes provisionales como puntos de control del progreso real del proyecto. Para guardar un plan provisional, seleccionar **Herramientas-> Seguimiento->Guardar línea de base,** hacer clic en **Guardar plan provisional**,

rellenar el campo **Copiar** con el nombre del plan que se desee guardar, el campo **En** con el nombre con el que se desea guardar el plan, y seleccionar la parte de la programación que se desea guardar como **Proyecto completo** o **Tareas seleccionadas**. Por ejemplo, si tenemos los campos **Comienzo** y **Fin** actualizados y queremos guardar el estado actual como un plan provisional, podemos Copiar "Comienzo/Fin" En "Comienzo1/Fin1". El plan provisional guarda los datos de fechas, pero no los de recursos. En la hoja de la izquierda podemos **Insertar->Columna** con el campo "Comienzo1" por ejemplo, para compararlo con el valor actual "Comienzo", o el de la línea de base "Comienzo previsto".

I.4.4. INTERPRETACIÓN DEL DIAGRAMA DE GANTT DE SEGUIMIENTO NORMAL

El eje temporal de la derecha en el **Diagrama de Gantt de seguimiento** nos muestra para cada tarea dos barras: la inferior, negra, corresponde a las fechas de la línea de base, mientras que la superior representa las fechas programadas. La barra superior será igual que en el diagrama de Gantt normal (azul claro) si la tarea no se ha comenzado, azul intenso en la fracción que se haya completado y roja si la tarea es crítica.

Cuando se actualizan los datos reales de una tarea si no coinciden con los previstos, automáticamente se reprograman todas las tareas que dependen de ella, esta estructura es la que representan las barras superiores. El diagrama de Gantt se seguimiento es muy útil porque permite ver gráficamente de forma inmediata las desviaciones del desarrollo del proyecto respecto al plan previsto.

Si tras haber introducido datos reales sobre el progreso del proyecto nos vamos al **Diagrama de Gantt** normal, observaremos que ha cambiado las barras de tareas cuya ejecución ya ha empezado tiene una barra delgada a lo largo de una fracción de su longitud igual al porcentaje de la duración da la tarea que ya se ha completado.

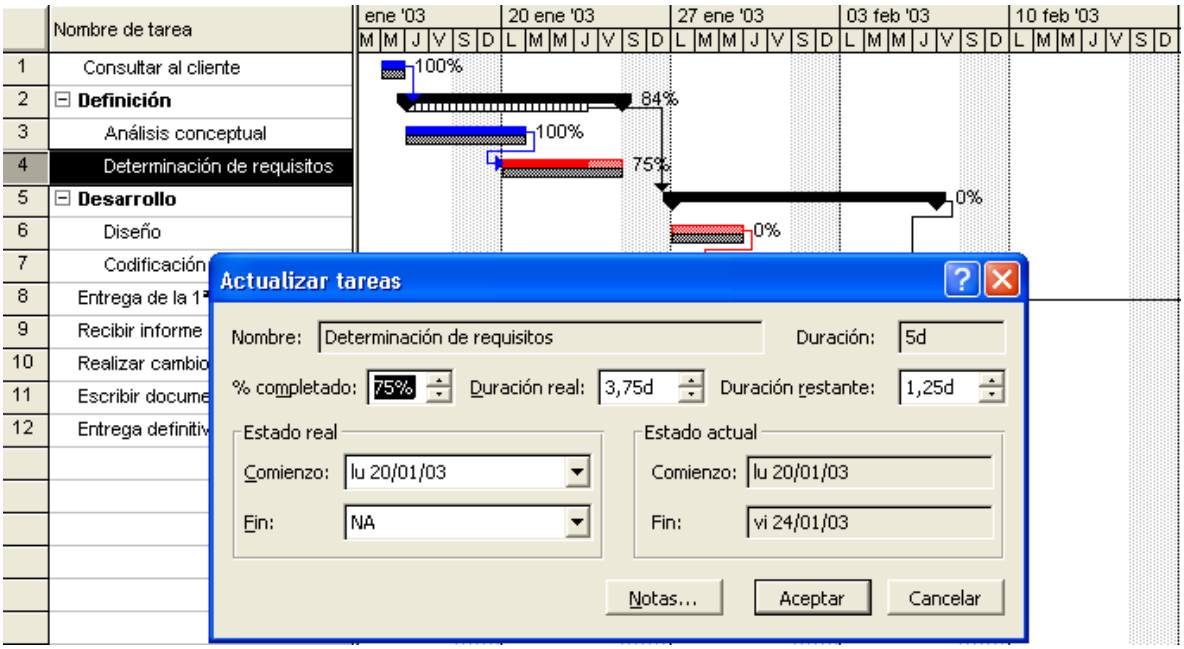


Figura I.11. Actualización de tareas en el seguimiento del proyecto.

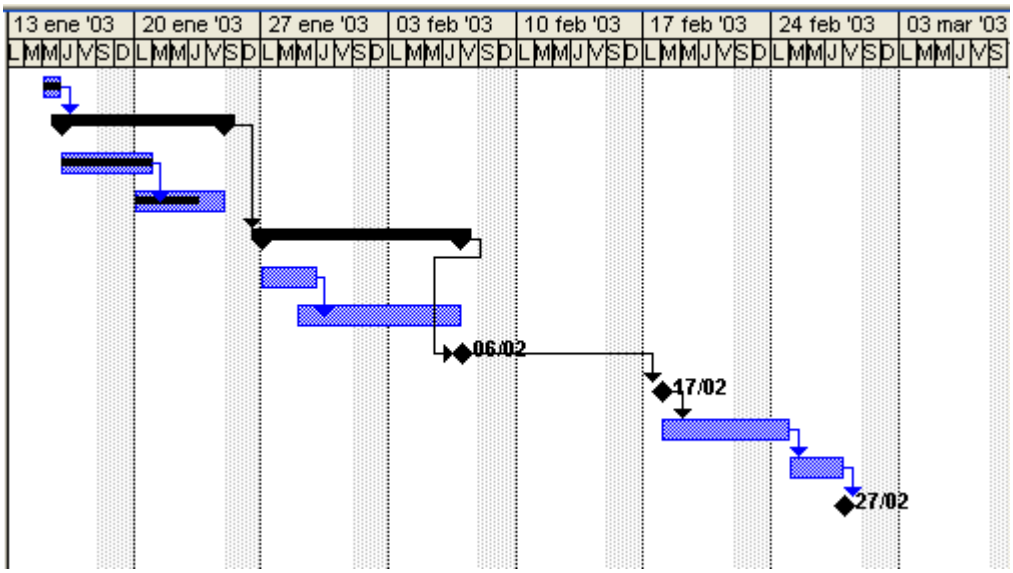


Figura I.12. Vista Diagrama de Gantt en el seguimiento del proyecto.

I.4.5. REPROGRAMACIÓN DE TAREAS

Si el estado actual de la programación (o sea, la estructura de dependencia entre tareas y sus fechas asociadas) no es conveniente, podemos cambiarla mediante los procedimientos explicados en los puntos I.3.3, I.3.4, y I.3.6: cambiar las dependencias

entre tareas, buscar márgenes de demora en la programación, superponer tareas o agregar tiempo de posposición entre ellas, etc.

Para esto puede ser útil seleccionar en la **Barra de Vistas** , **Más Vistas** y **Gantt detallado**, que es un diagrama de Gantt en el que a la derecha de cada tarea sin completar aparece una barra delgada cuya longitud representa la demora permisible de la tarea, que es el máximo retraso que puede tener esa tarea sin que afecte a tareas sucesoras o a la fecha de fin del proyecto.

I.5. PRESENTACIÓN DE LA INFORMACIÓN

Microsoft Project permite cambiar el formato de la información que se va a imprimir. Por ejemplo, si en el diagrama de Gantt no se quiere que junto a las barras de tarea aparezcan los nombres de los recursos, hay que seleccionar en el menú **Formato->Estilos de barra**, y en la ficha Texto borrar "Nombres de los recursos".

Si por ejemplo se quieren poner las tareas críticas en rojo, lo primero será averiguar cuáles son las tareas críticas eligiendo la vista **Diagrama Pert**, como se explica en el punto 3.4. A continuación, en la vista Diagrama de Gantt, habrá que seleccionar sus **Nombres de tareas** (se pueden seleccionar varias a la vez pulsando CTRL), buscar en el menú **Formato->Barra** y en la ficha Forma de la barra cambiar el **Color** de la **Barra Central**.

Antes de imprimir una determinada vista puede interesar añadir un encabezado con información sobre el proyecto, o números de página, etc. Para diseñar la presentación seleccionamos **Archivo->Confígurar página** y hacemos los cambios deseados en el cuadro de diálogo (ver figura I.13).

Para incluir información del proyecto hay que elegir el dato deseado en el cuadro desplegado de abajo y pulsar Agregar. Esos datos correspondientes aparecen dentro de &[ ] y los proporciona el programa, por eso, para que el cuadro anterior dé el deseado, hay que asegurarse de que en la fícha Resumen de '"Organización" se hayan rellenado

con el titulo del proyecto, el nombre del responsable del grupo de trabajo y el número que identifica al grupo.

En el cuadro "Muestra" (figura I.13) podemos ver la apariencia que tendrá el encabezado que estamos diseñando.

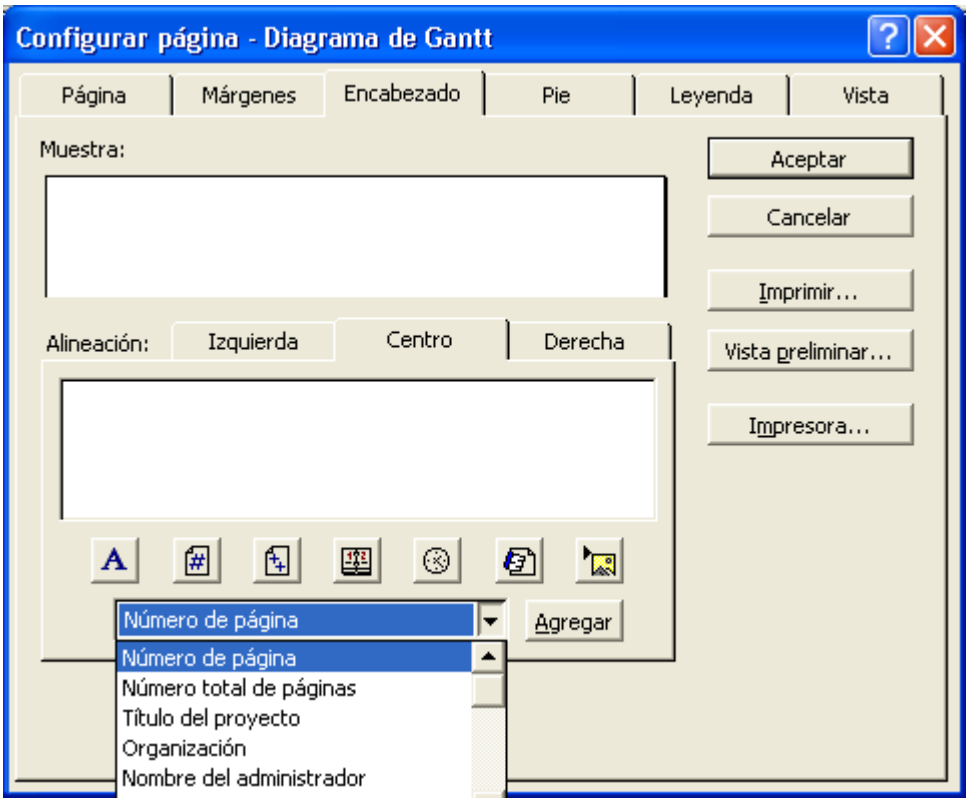


Figura I.13. Configuración de la presentación.

Análogamente, si queremos añadir el número de página al pie y a la derecha tendremos que seleccionar **Pie, Derecha** y **Agregar** "Número de página".

En cuanto a la ficha **Leyenda**, nos permite decidir si queremos tener una leyenda en cada página, si va a haber una página de leyenda o si no se va a imprimir leyenda.

También podemos decidir qué información va a aparecer en una determinada vista, eligiendo la tabla a aplicar y moviendo las líneas verticales de separación con el puntero (se pueden incluso hacer desaparecer columnas). Por ejemplo, si en un diagrama de Gantt queremos que aparezcan las columnas del número, nombre y duración de tarea lo

más rápido es elegir **Ver->Tabla->Resumen** y mover la línea vertical de separación con el gráfico para que sólo se vean estas tres primeras columnas.

Antes de **Imprimir** una vista es conveniente comprobar que va a tener el aspecto deseado seleccionando **Vista Preliminar**. Ambas opciones están en el menú **Archivo** y en la ficha **Configurar página**.

Para ver en pantalla el proyecto completo hay que **seleccionar Ver -> Zoom -> proyecto completo**. Si se quiere imprimir así , en una sola página, a veces ocurre que la fecha del último hito (fin del proyecto) se queda fuera de la **Vista preliminar**. Para solucionarlo, se puede poner esta fecha a la izquierda del hito, seleccionando su **Nombre de tarea** y, en la ficha **Texto de la barra** de **Formato-> Barra**, poner **Comienzo** en **Izquierda** y nada en **Derecha**.

ANEXO II

LISTADO DEL PROGRAMA

Con el listado del código se pretende dar una facilidad a las personas que estén interesadas en conocer en profundidad PlanFal. Para facilitar esta tarea se ha intentado evitar las complicaciones que implica la lectura de un código comentado. Para ello, se hace una breve descripción (si fuese necesario) antes de cada método y variable, y se presenta la interfaz gráfica de los formularios, con el nombre de los componentes más relevantes.

Para una mejor comprensión de la estructura del código, se especifican las llamadas entre métodos. Esto nos permite comprender la evolución del programa sin realizar un diagrama de flujos, que sería muy complicado debido a la gran comunicación que exige con el usuario el uso de PlanFal. Para no recargar el listado no se han tenido en cuenta, excepto en contadas ocasiones, como llamadas a métodos, las que se realizan al objeto avión, tanto a sus propiedades como a sus métodos, puesto que la modificación de las propiedades del objeto avion se realiza a través de sus métodos.

Para el listado del código del programa se ha utilizado el siguiente criterio:

- Objetos de Microsoft: el objeto de Microsoft es el fichero Project que contiene la macro. Utilizamos este"módulo" para definir las variables globales de la aplicación. Tenemos 7 variables globales.

- Módulos de Clase: tenemos un objeto creado para PlanFal (avion), con 19 métodos y 21 propiedades.

- Formularios: Se han creado 12 formularios con un total de 57 métodos.

- Módulos: se tienen 11 módulos con un total de 61 métodos.

MICROSOFT PROJECT OBJETOS (PlanFal.mpt)

' Variables globales a la aplicación

NOMBRE: NumeroAviones
DESCRIPCIÓN: indica el número de aviones que componen la planificación.
SINTAXIS: Public NumeroAviones As Integer

NOMBRE: RepRef
DESCRIPCIÓN: indica el número de repetición en la curva de aprendizaje del avión de referencia.
SINTAXIS: Public RepRef As Integer

NOMBRE: AvCC
DESCRIPCIÓN: indica el número de avión de comienzo de curva de aprendizaje.
SINTAXIS: Public AvCC As Integer

NOMBRE: NumeroRepAvCC
DESCRIPCIÓN: indica el número de repetición en la curva de aprendizaje del avión de comienzo de curva..
SINTAXIS: Public NumeroRepAvCC As Integer

NOMBRE: AvFC
DESCRIPCIÓN: indica el número de avión de fin de curva de aprendizaje. Conocido como avión de régimen.
SINTAXIS: Public AvFC As Integer

NOMBRE: OpMax
DESCRIPCIÓN: Número de operarios máximos. (máxima unidad de recursos en Project).
SINTAXIS: Public OpMax As Single

NOMBRE: OpDefecto
DESCRIPCIÓN: Número de operarios que tendremos por defecto al crear una nueva estación.
SINTAXIS: Public OpDefecto As Single

NOMBRE: TurnoDefecto

DESCRIPCIÓN: turnos de trabajo de 8 horas que tendremos por defecto al crear una nueva estación.

SINTAXIS: Public TurnoDefecto As Integer

MÓDULOS DE CLASE

Avion

PROPIEDADES

TaskAvion

TIPO: Task

DESCRIPCIÓN: objeto tarea de Project que corresponde al avion.

Especial

TIPO: Boolean

DESCRIPCIÓN: indica (true) sin el avión es especial.

EE

TIPO: Boolean

DESCRIPCIÓN: indica (true) si el avión tiene enlaces especiales.

AplicaCurva

TIPO: Boolean

DESCRIPCIÓN: indica si para el calculo de duraciones se le aplica curva de reducción.

NumeroAvion

TIPO: Integer

DESCRIPCIÓN: número de avión en la planificación.

NumeroEstaciones

TIPO: Integer

DESCRIPCIÓN: número de estaciones asociadas al ensamblaje del avion.

NumeroMaximoSubestaciones

TIPO: Integer

DESCRIPCIÓN: cantidad máxima de subestaciones dentro de una estación de entre todas las estaciones

TaskGlobal

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como un array de todas las tareas (tasks) (avión, estaciones, subestaciones) asociadas al avión.

Estaciones

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como un array de enteros.

Estaciones (x) = y, donde:

x : número de estación –1

y: número de subetaciones que tiene la estación

Duraciones

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Single.

Duraciones (x,y) = z, donde :

x : número de estación –1

y: número de subetacion

z: Duación en días de la estación- subestación.

HorasTrabajo

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Single. Igual que Duraciones pero z son las horas de trabajo reales.

Operarios

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Single. Igual que Duraciones pero z son los operarios totales de la estación-subestación.

Curvas

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Single. Igual que Duraciones pero z son los factores de reducción de la curva de aprendizaje asociada a la duración en días.

CurvasHoras

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Single. Igual que Curvas pero aplicado a la reducción de horas de trabajo.

Posiciones

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Integer. Igual que Duraciones pero z son las posiciones que tiene la estación-subestación.

turnos

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Integer. Igual que Duraciones pero z es el número de turnos de trabajo al día (1,2 ó 3).

Descripcion

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos String. Igual que Duraciones pero z es la descripción de cada estación-subestación.

Nombre

TIPO: Variant

DESCRIPCIÓN: matriz igual que descripción, en la que se guardan los nombres de las estaciones-subestaciones.

EnlaceEstaciones

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Integer. Igual que Duraciones pero z es la estación del avión anterior con la que enlaza la estación – subestación (x,y).

EnlaceSubestaciones

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 2 X 2 de tipos Integer. Igual que Duraciones pero z es la subestación del avión anterior con la que enlaza la estación – subestación (x,y).

Predecesoras

TIPO: Variant

DESCRIPCIÓN: variant que se utilizará como una matriz 3 X 3 de tipos Integer. Nos informa como son los enlaces entre subestaciones dentro de una estación.

Predecesoras (x,y,z) = k, donde :

x : número de estación –1

y: número de subestacion

z:número de subestación con la que enlaza y. Siempre menor que y.

K: si vale 1, la subestación y la subestación z estan enlazadas Fin a Comienzo, en caso contrario (cualquier otro valor), no estan enlazadas.

MÉTODOS

Inicializa

DESCRIPCIÓN: método que da los valores iniciales a la creación de un objeto avión.

CÓDIGO:

```
Sub Inicializa()
   NumeroAvion = 1
   NumeroEstaciones = 1
   NumeroMaximoSubestaciones = 0
   Especial = False
   EE = False
   AplicaCurva = False
   Set TaskAvion = Nothing
   ReDim Estaciones(0) As Integer
   ReDim Posiciones(0, 0) As Integer
   ReDim Nombre(0, 0) As String
   ReDim Descripcion(0, 0) As String
   ReDim EnlaceEstaciones(0, 0) As Integer
   ReDim EnlaceSubestaciones(0, 0) As Integer
   ReDim Predecesoras(0, 0, 0) As Integer
   ReDim Duraciones(0, 0) As Single
   ReDim Curvas(0, 0) As Single
   ReDim Operarios(0, 0) As Single
   ReDim CurvasHoras(0, 0) As Single
   ReDim HorasTrabajo(0, 0) As Single
   ReDim turnos(0, 0) As Integer
End Sub
```

RDTask

DESCRIPCIÓN: redimensiona la variable TaskGlobal

CÓDIGO:

```
Sub RDTask()
Dim x, y As Integer
x = NumeroEstaciones
y = NumeroMaximoSubestaciones
ReDim TaskGlobal(x - 1, y) As Task
End Sub
```

Actualiza1D

DESCRIPCIÓN: actualiza y redimensiona la variable Estaciones.

PARÁMETROS:

numdim: Nueva dimensión del vector [0,numdim].

pos : Opcional. Posicion que se crea o elimmina [1,numdim+1]

Eliminar: Opcional. Si Eliminar = true, entonces pos se elimina, por defecto se crea.

CÓDIGO:

```
Sub Actualiza1D(numdim As Integer, Optional pos As Integer, Optional Eliminar As Boolean)
Dim buffer As Variant
 Dim i, p As Integer
If pos = 0 Then
pos = numdim + 2
 End If
 p = pos - 1
 buffer = Estaciones
 ReDim Preserve Estaciones(numdim)
  If Eliminar = False Then
  If p >= 0 And p < numdim Then
   For i = p To (numdim - 1)
     Estaciones(i + 1) = buffer(i)
    Next
     Estaciones(p) = 0
   End If
End If
If Eliminar = True Then
 For i = p To (numdim)
      Estaciones(i) = buffer(i + 1)
  Next
  End If
NumeroEstaciones = numdim + 1
End Sub
```

Actualiza2D

DESCRIPCIÓN: actualiza y redimensiona las propiedades de avion que son matrices de 2 X2.

PARÁMETROS:

numdimx: [0,numdimx], numero de estaciones-1

numdimy: [0,numdimy], numero máximo de subestaciones

posx: Opcional. posicion que se crea o elimina [1,numdimx+1]

Eliminar: Opcional. Indica (true) si la posición pox se elimina.

CÓDIGO:

```
Sub Actualiza2D(numdimx As Integer, numdimy As Integer, Optional posx As Integer, Optional Eliminar As Boolean)
Dim buffer1, buffer2 , buffer3 , buffer4 , buffer5 , buffer6,  buffer7,  buffer8,  buffer9 As Variant
Dim buffer10, buffer11 As Variant
 Dim i ,  j As Integer
 buffer1 = Duraciones
 buffer2 = Curvas
 buffer4 = Posiciones
 buffer3 = Descripcion
 buffer5 = Nombre
 buffer6 = EnlaceEstaciones
 buffer7 = EnlaceSubestaciones
 buffer8 = HorasTrabajo
 buffer9 = CurvasHoras
 buffer10 = Operarios
 buffer11 = turnos
 ReDim Preserve buffer1(UBound(buffer1, 1), numdimy)
 ReDim Preserve buffer2(UBound(buffer2, 1), numdimy)
 ReDim Preserve buffer3(UBound(buffer3, 1), numdimy)
 ReDim Preserve buffer4(UBound(buffer4, 1), numdimy)
 ReDim Preserve buffer5(UBound(buffer5, 1), numdimy)
 ReDim Preserve buffer6(UBound(buffer6, 1), numdimy)
 ReDim Preserve buffer7(UBound(buffer7, 1), numdimy)
 ReDim Preserve buffer8(UBound(buffer8, 1), numdimy)
 ReDim Preserve buffer9(UBound(buffer9, 1), numdimy)
 ReDim Preserve buffer10(UBound(buffer10, 1), numdimy)
 ReDim Preserve buffer11(UBound(buffer11, 1), numdimy)
 ReDim Duraciones(numdimx, numdimy)
 ReDim Curvas(numdimx, numdimy)
 ReDim Descripcion(numdimx, numdimy)
 ReDim Posiciones(numdimx, numdimy)
 ReDim Nombre(numdimx, numdimy)
```

```
 ReDim EnlaceEstaciones(numdimx, numdimy)
 ReDim EnlaceSubestaciones(numdimx, numdimy)
 ReDim Operarios(numdimx, numdimy)
 ReDim HorasTrabajo(numdimx, numdimy)
 ReDim turnos(numdimx, numdimy)
 ReDim CurvasHoras(numdimx, numdimy)
If posx = 0 Then
 posx = numdimx + 2
End If
p = posx - 1
If Eliminar = False Then
  If p < numdimx Then
 For i = 0 To p - 1
   For j = 0 To numdimy
    Duraciones(i, j) = buffer1(i, j)
    Curvas(i, j) = buffer2(i, j)
    Descripcion(i, j) = buffer3(i, j)
    Posiciones(i, j) = buffer4(i, j)
    Nombre(i, j) = buffer5(i, j)
    EnlaceEstaciones(i, j) = buffer6(i, j)
    EnlaceSubestaciones(i, j) = buffer7(i, j)
    HorasTrabajo(i, j) = buffer8(i, j)
    CurvasHoras(i, j) = buffer9(i, j)
    Operarios(i, j) = buffer10(i, j)
    turnos(i, j) = buffer11(i, j)
     Next j
 Next i
  For i = p + 1 To numdimx
   For j = 0 To numdimy
     Duraciones(i, j) = buffer1(i - 1, j)
     Curvas(i, j) = buffer2(i - 1, j)
     Descripcion(i, j) = buffer3(i - 1, j)
     Posiciones(i, j) = buffer4(i - 1, j)
     Nombre(i, j) = buffer5(i - 1, j)
     EnlaceEstaciones(i, j) = buffer6(i - 1, j)
     EnlaceSubestaciones(i, j) = buffer7(i - 1, j)
     HorasTrabajo(i, j) = buffer8(i - 1, j)
     CurvasHoras(i, j) = buffer9(i - 1, j)
     Operarios(i, j) = buffer10(i - 1, j)
     turnos(i, j) = buffer11(i - 1, j)
```

```
    Next j
  Next i
  Else
 For i = 0 To UBound(buffer1, 1)
   For j = 0 To numdimy
    Duraciones(i, j) = buffer1(i, j)
    Curvas(i, j) = buffer2(i, j)
    Descripcion(i, j) = buffer3(i, j)
    Posiciones(i, j) = buffer4(i, j)
    Nombre(i, j) = buffer5(i, j)
    EnlaceEstaciones(i, j) = buffer6(i, j)
    EnlaceSubestaciones(i, j) = buffer7(i, j)
    HorasTrabajo(i, j) = buffer8(i, j)
    CurvasHoras(i, j) = buffer9(i, j)
    Operarios(i, j) = buffer10(i, j)
    turnos(i, j) = buffer11(i, j)
     Next j
 Next i
    End If
  End If
If Eliminar = True Then
  If p <= numdimx Then
 For i = 0 To p - 1
   For j = 0 To numdimy
    Duraciones(i, j) = buffer1(i, j)
    Curvas(i, j) = buffer2(i, j)
    Descripcion(i, j) = buffer3(i, j)
    Posiciones(i, j) = buffer4(i, j)
    Nombre(i, j) = buffer5(i, j)
    EnlaceEstaciones(i, j) = buffer6(i, j)
    EnlaceSubestaciones(i, j) = buffer7(i, j)
    HorasTrabajo(i, j) = buffer8(i, j)
    CurvasHoras(i, j) = buffer9(i, j)
    Operarios(i, j) = buffer10(i, j)
    turnos(i, j) = buffer11(i, j)
     Next j
    Next i
  For i = p To numdimx
   For j = 0 To numdimy
     Duraciones(i, j) = buffer1(i + 1, j)
```

```
    Curvas(i, j) = buffer2(i + 1, j)
    Descripcion(i, j) = buffer3(i + 1, j)
    Posiciones(i, j) = buffer4(i + 1, j)
    Nombre(i, j) = buffer5(i + 1, j)
    EnlaceEstaciones(i, j) = buffer6(i + 1, j)
    EnlaceSubestaciones(i, j) = buffer7(i + 1, j)
    HorasTrabajo(i, j) = buffer8(i + 1, j)
    CurvasHoras(i, j) = buffer9(i + 1, j)
    Operarios(i, j) = buffer10(i + 1, j)
    turnos(i, j) = buffer11(i + 1, j)
      Next j
 Next i
Else
For i = 0 To numdimx
  For j = 0 To numdimy
   Duraciones(i, j) = buffer1(i, j)
   Curvas(i, j) = buffer2(i, j)
   Descripcion(i, j) = buffer3(i, j)
   Posiciones(i, j) = buffer4(i, j)
   Nombre(i, j) = buffer5(i, j)
   EnlaceEstaciones(i, j) = buffer6(i, j)
   EnlaceSubestaciones(i, j) = buffer7(i, j)
   HorasTrabajo(i, j) = buffer8(i, j)
   CurvasHoras(i, j) = buffer9(i, j)
   Operarios(i, j) = buffer10(i, j)
   turnos(i, j) = buffer11(i, j)
     Next j
 Next i
  End If
End If
 For i = 0 To numdimx
  For j = 0 To numdimy
   If Curvas(i, j) = 0 Then
    Curvas(i, j) = 100
    End If
   If CurvasHoras(i, j) = 0 Then
    CurvasHoras(i, j) = 100
    End If
    If turnos(i, j) = 0 Then
      turnos(i, j) = planning.TurnoDefecto
```

```
      End If
     If Operarios(i, j) = 0 Then
       Operarios(i, j) = planning.OpDefecto
      End If
    If Posiciones(i, j) = 0 Then
      Posiciones(i, j) = 1
     End If
    Next j
 Next i
' Actualizamos las propiedades relacionadas
 NumeroEstaciones = numdimx + 1
 NumeroMaximoSubestaciones = numdimy
End Sub
```

Actualiza3D

DESCRIPCIÓN: actualiza y redimensiona las propiedades de avion que son matrices de 3 X3.

PARÁMETROS:

numdimx: [0,numdimx], numero de estaciones-1

numdimy: [0,numdimy], numero máximo de subestaciones

posx: Opcional. posicion que se crea o elimina [1,numdimx+1]

Eliminar: Opcional. Indica (true) si la posición pox se elimina.

CÓDIGO

```
Sub Actualiza3D(numdimx As Integer, numdimy As Integer, Optional posx As Integer, Optional Eliinar As Boolean)
 Dim buffer1 As Variant
 Dim buffer2() As Integer
 Dim i ,  j , k,  p As Integer
 Dim numdimz As Integer
 If numdimy = 0 Then
  numdimz = 0
  Else
  numdimz = numdimy - 1
 End If
  buffer1 = Predecesoras
 Erase Predecesoras
 ReDim Predecesoras(numdimx, numdimy, numdimz)
 ReDim buffer2(numdimx, numdimy, numdimz)
  If posx = 0 Then
  posx = numdimx + 2
 End If
```

```
 p = posx - 1
If Eliminar = False Then
   For i = 0 To UBound(buffer1, 1)
   For j = 0 To UBound(buffer1, 2)
   For k = 0 To UBound(buffer1, 3)
   buffer2(i, j, k) = buffer1(i, j, k)
   Next k
  Next j
 Next i
 If p < numdimx Then
 For i = 0 To p - 1
   For j = 0 To numdimy
    For k = 0 To numdimz
    Predecesoras(i, j, k) = buffer2(i, j, k)
    Next k
   Next j
 Next i
  For i = p + 1 To numdimx
   For j = 0 To numdimy
    For k = 0 To numdimz
     Predecesoras(i, j, k) = buffer2(i - 1, j, k)
     Next k
    Next j
  Next i
 Else
 For i = 0 To UBound(buffer1, 1)
   For j = 0 To numdimy
    For k = 0 To numdimz
    Predecesoras(i, j, k) = buffer2(i, j, k)
    Next k
   Next j
 Next i
   End If
  End If
If Eliminar = True Then
  For i = 0 To numdimx
  For j = 0 To UBound(buffer1, 2)
   For k = 0 To UBound(buffer1, 3)
    buffer2(i, j, k) = buffer1(i, j, k)
   Next k
```

```
 Next j
Next i
 If p <= numdimx Then
For i = 0 To p - 1
 For j = 0 To numdimy
  For k = 0 To numdimz
  Predecesoras(i, j, k) = buffer2(i, j, k)
  Next k
 Next j
Next i
 For i = p To numdimx
 For j = 0 To numdimy
   For k = 0 To numdimz
   Predecesoras(i, j, k) = buffer1(i + 1, j, k)
   Next k
  Next j
 Next i
Else
For i = 0 To numdimx
 For j = 0 To numdimy
  For k = 0 To numdimz
  Predecesoras(i, j, k) = buffer2(i, j, k)
  Next k
 Next j
Next i
End If

End If
 NumeroEstaciones = numdimx + 1
 NumeroMaximoSubestaciones = numdimyEnd Sub
```

setEstaciones

DESCRIPCION: modifica la propiedad Estaciones

PARÁMETROS:

i: Estación 1

e: Númerode subestaciones

CÓDIGO:

```
Sub setEstaciones(i As Integer, e As Integer)
 Estaciones(i) = e
End Sub
```

setDuraciones

DESCRIPCIÓN: modifica la propiedad Duraciones

PARÁMETROS:

i: Estación -1

j: Subestación

d: duración en días

CÓDIGO:

```
Sub setDuraciones(i As Integer, j As Integer, d As Single)
 Duraciones(i, j) = d
End Sub
```

SetHorasTrabajo

DESCRIPCIÓN: modifica la propiedad HorasTrabajo

PARÁMETROS:

i: Estación -1

j: Subestación

h: horas de trabajo

CÓDIGO:

```
Sub setHorasTrabajo(i As Integer, j As Integer, h As Single)
 HorasTrabajo(i, j) = h
End Sub
```

setOperarios

DESCRIPCIÓN: modifica la propiedad Operarios

PARÁMETROS:

i: Estación -1

j: Subestación

Op: Número de operarios

CÓDIGO:

```
Sub setOperarios(i As Integer, j As Integer, Op As Single)
 Operarios(i, j) = Op
End Sub
```

setTurnos

DESCRIPCIÓN: modifica la propiedad turnos

PARÁMETROS:

i: Estación -1

j: Subestación

t: Número de turnos (1,2 ó 3)

CÓDIGO:

```
Sub setTurnos(i As Integer, j As Integer, t As Integer)
 turnos(i, j) = t
End Sub
```

setCurvas

DESCRIPCIÓN: modifica la propiedad Curvas

PARÁMETROS:

i: Estación -1

j: Subestación

c: factor de curva de días (%)

CÓDIGO:

```
Sub setCurvas(i As Integer, j As Integer, c As Single)
 Curvas(i, j) = c
End Sub
```

setCurvasHoras

DESCRIPCIÓN: modifica la propiedad CurvasHoras

PARÁMETROS:

i: Estación -1

j: Subestación

ch: factor de curva de horas de trabajo (%)

CÓDIGO:

```
Sub setCurvasHoras(i As Integer, j As Integer, ch As Single)
 CurvasHoras(i, j) = ch
End Sub
```

setPosiciones

DESCRIPCIÓN: modifica la propiedad Posiciones

PARÁMETROS:

i: Estación -1

j: Subestación

p: número de posiciones

CÓDIGO:

```
Sub setPosiciones(i As Integer, j As Integer, p As Integer)
 Posiciones(i, j) = p
End Sub
```

SetEnlaceEstaciones

DESCRIPCIÓN: modifica la propiedad EnlaceEstaciones

PARÁMETROS:

i: Estación -1

j: Subestación

e: estación de enlace del avión anterior

CÓDIGO:

```
Sub setEnlaceEstaciones(i As Integer, j As Integer, e As Integer)
 EnlaceEstaciones(i, j) = e
End Sub
```

SetEnlaceSubestaciones

DESCRIPCIÓN: modifica la propiedad EnlaceSubestaciones

PARÁMETROS:

i: Estación -1

j: Subestación

s: Subestación de enlace del avión anterior

CÓDIGO:

```
Sub setEnlaceSubestaciones(i As Integer, j As Integer, s As Integer)
 EnlaceSubestaciones(i, j) = s
End Sub
```

SetNombre

DESCRIPCIÓN: modifica la propiedad Nombre

PARÁMETROS:

i: Estación -1

j: Subestación

n: nombre de la estación.subestación

CÓDIGO:

```
Sub setNombre(i As Integer, j As Integer, n As String)
 Nombre(i, j) = n
End Sub
```

SetDescripcion

DESCRIPCIÓN: modifica la propiedad Decripcion

PARÁMETROS:

i: Estación -1

j: Subestación

d: nombre de la estación.subestación

CÓDIGO:

```
Sub setDescripcion(i As Integer, j As Integer, d As String)
 Descripcion(i, j) = d
```

End Sub

SetTaskGlobal

DESCRIPCIÓN: modifica la propiedad TaskGlobal

PARÁMETROS:

i: Estación -1

j: Subestación

o: Tarea de Project

CÓDIGO:

```
Sub setTakGlobal(i As Integer, j As Integer, o As Task)
 Set TaskGlobal(i, j) = o
End Sub
```

SetPredecesoras

DESCRIPCIÓN: modifica la propiedad Predecesoras

PARÁMETROS:

i: Estación -1

j: Subestación

k: Subestación de enlace

p: indicador de enlace

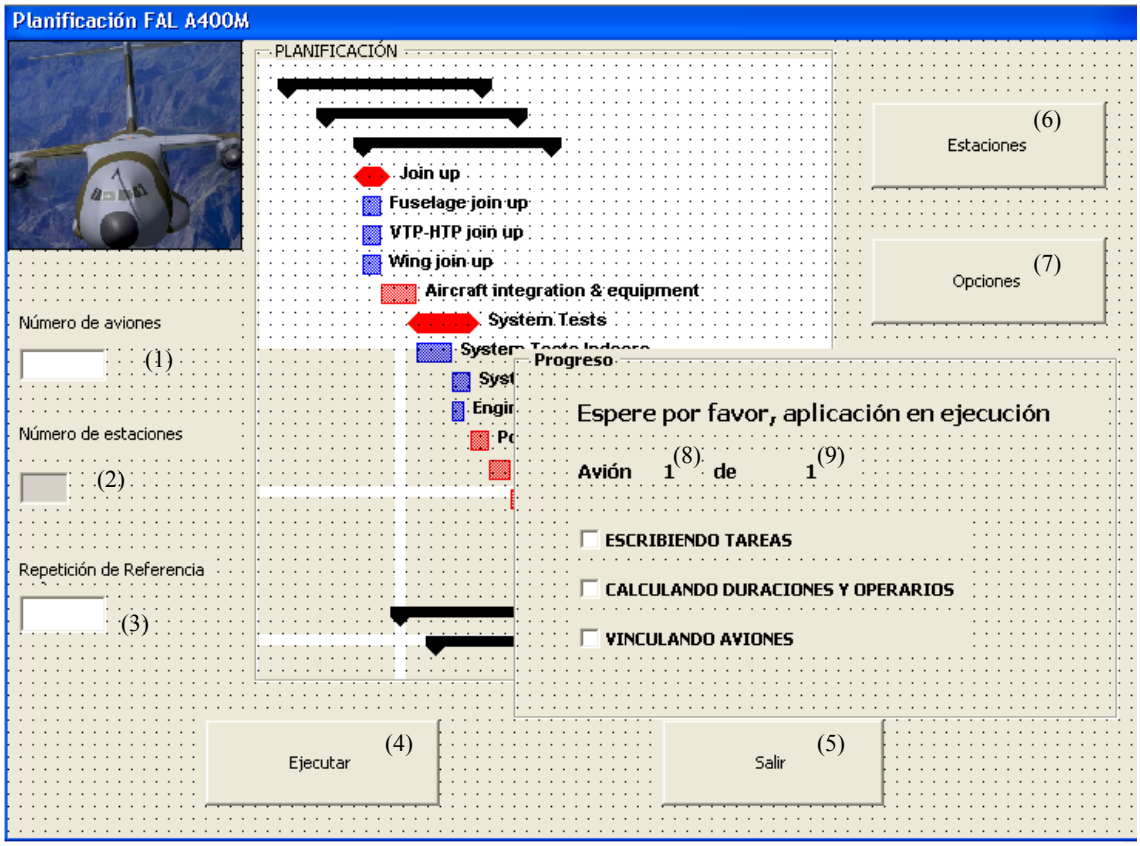
CÓDIGO:

```
Sub setPredecesoras(i As Integer, j As Integer, k As Integer, p As Integer)
 Predecesoras(i, j, k) = p
End Sub
```

FORMULARIOS

MENÚ PRINCIPAL (FrmGeneral)

INTERFAZ GRÁFICA



(1),(2),(3): NA, NE, RR. (TextBox)

(4),(5),(6),(7):Ejecutar, Salir, Estaciones, Opciones, DemoButton. (Command-Button)

(8),(9): Lav, Lnav. (Label)

MÉTODOS

Userform_Initialize

DESCRIPCIÓN: método que se ejecuta cuando se carga por primera vez el formulario (FrmGeneral.Show).

LLAMADAS ENTRANTES:

Inicio.inicio

LLAMADAS SALIENTES:

Inicializacion.RecuperarDatos

Control.ControlDatosValidos (AvRR)

CÓDIGO:

```
Private Sub UserForm_Initialize()
On Error GoTo fallo:
ViewApply Name:="Diagrama de Gantt"
OptionsSchedule AssignmentUnits:=1
Inicializacion.RecuperarDatos
If Control.ControlDatosValidos(AvRR) = False Then
 GoTo fallo
End If
NE.Text = AvRR.NumeroEstaciones
NA.Text = planning.NumeroAviones
RR.Text = planning.RepRef
Exit Sub
fallo:
 Dim respuesta As Variant
 respuesta = MsgBox("Los datos existentes no permtien inicializar correctamente. ¿Desea continuar?", vbOKCancel)
  If respuesta = vbOK Then
        NA.Text = 1
        NE.Text = 1
       RR.Text = 1
        FrmOpciones.CC.Text = 1
        FrmOpciones.NRCC.Text = 1
        FrmOpciones.TBOpDef = planning.OpDefecto
        FrmOpciones.TBOpMax = planning.OpMax
        GestAv.AvRR.Inicializa
        GestAv.AvRR.setCurvas 0, 0, 100
        GestAv.AvRR.setPosiciones 0, 0, 1
      Else
     End
```

```
  End If
End Sub
```

NA_AfterUpdate

DESCRIPCIÓN: controla y actualiza los cambios en el número de aviones. Realiza un control de datos válidos. Avisa si una reducción de aviones implica eliminar aviones catalogados como especiales , el avión de referencia o el de comienzo de curva.

LLAMADAS SALIENTES:

GestAv.EliminarAvionesEpeciales(GestAv.AvEspeciales(i))

CÓDIGO:

```
Private Sub NA_AfterUpdate()
On Error GoTo fin
Dim respuesta As Variant
Dim i As Integer
Dim j As Integer
j = GestAv.NumeroAvEspeciales
If NA.Text <= 0 Then
 MsgBox " El número de aviones debe ser positivo", vbOKOnly
 NA.Text = planning.NumeroAviones
 Exit Sub
End If
If NA.Text < planning.AvCC Then
   MsgBox " Este número de aviones supondria la eliminación del avión de Comienzo de Curva!! "
   NA.Text = planning.NumeroAviones
   Exit Sub
End If
If j > 0 Then
 If NA.Text < GestAv.AvEspeciales(j - 1) Then
 respuesta = MsgBox("esta modificación puede eliminar aviones catalogados como especiales.¿Desea continuar?", vbOKCancel)
 If respuesta = 1 Then
 For i = j - 1 To 0 Step -1
   If NA.Text < GestAv.AvEspeciales(i) Then
     GestAv.EliminarAvionEspecial (GestAv.AvEspeciales(i))
   Else
    Exit For
   End If
  Next
 planning.NumeroAviones = NA.Text
 ReDim Preserve aviones(1 To planning.NumeroAviones)
```

```
 Else
  NA.Text = planning.NumeroAviones
  Exit Sub
 End If
Else
 planning.NumeroAviones = NA.Text
 ReDim Preserve aviones(1 To planning.NumeroAviones)
End If
Else
 planning.NumeroAviones = NA.Text
 ReDim Preserve aviones(1 To planning.NumeroAviones)
End If
Exit Sub
fin:
  MsgBox "    Dato incorrecto"
  NA.Text = planning.NumeroAviones
End Sub
```

RR_AfterUpdate

DESCRIPCIÓN: controla y actualiza cambios en el valor de la repetición de referencia.

LLAMADAS SALIENTES:

GestAv.CambioAvionReferencia(RR.Text, planning.NumeroRepAvCC, planning.AvCC)

CÓDIGO:

```
Private Sub RR_AfterUpdate()
On Error GoTo fin
If RR.Text > 0 Then
  If GestAv.CambioAvionReferencia(RR.Text, planning.NumeroRepAvCC, planning.AvCC) Then
  Else
   RR.Text = planning.RepRef
  End If
Else
  MsgBox " El número de aviones debe ser positivo", vbOKOnly
  RR.Text = planning.RepRef
End If
Exit Sub
fin:
    MsgBox "    Dato incorrecto"
   RR.Text = planning.RepRef
End Sub
```

Ejecutar_Click

DESCRIPCIÓN: empezamos la ejecución de la planificación. Si existen problemas durante la ejecución, se muestra un mensaje de error y volvemos a Project.

LLAMADAS SALIENTES:

Interno.Ejecutar

CÓDIGO:

```
Private Sub Ejecutar_Click()
On Error GoTo fallo:
Interno.Ejecutar
End
Exit Sub
fallo:
 MsgBox " Hay datos incorrectos o no esperados por el programa", vbOKOnly
 Application.SelectCell 1, 3
 End Sub
```

Salir_Click

DESCRIPCIÓN: salimos del programa (descarga en memoria el formulario)sin hacer ningún cambio.

CÓDIGO:

```
Private Sub Salir_Click()
  Unload Me
End Sub
```

Estaciones_Click

DESCRIPCIÓN: cargamos el formulario estaciones. Al ser llamado desde el menú principal, los datos mostrados son los del avión de referencia.

LLAMADAS SALIENTES:

FrmEstaciones.Show

CÓDIGO:

```
Private Sub Estaciones_Click()
Set GestFrmEstaciones.av = AvRR
FrmEstaciones.Show
End Sub
```

Opciones_click

DESCRIPCIÓN: cargamos el formulario opciones.

LLAMADAS SALIENTES:

FrmOpciones.Show

CÓDIGO:

```
Private Sub Opciones_Click()
```

FrmOpciones.Show
End Sub

DemoButton_Click

DESCRIPCIÓN: cargamos los valores de demostración que se encuemtran en el módulo Demo.

LLAMADAS SALIENTES:

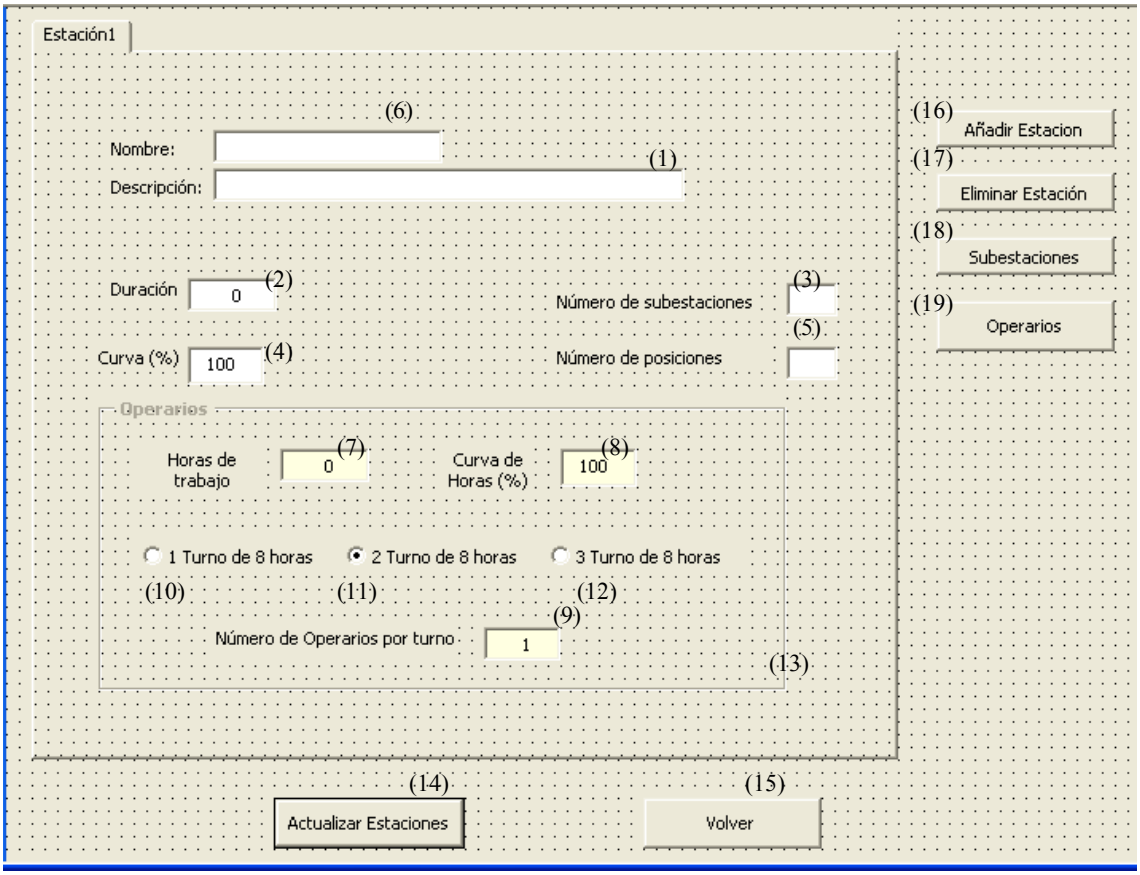
Demo.PlanificacionDemo

CÓDIGO:

```
Private Sub DemoButton_Click()
 Dim res As Integer
 res = MsgBox(" Esta acción sustituirá los valores actuales por los valores de demostración"vbOKCancel)
 If res = 1 Then
  Demo.PlanificacionDemo
 End If
End Sub
```

MENÚ ESTACIONES (FrmEstaciones)

INTERFAZ GRÁFICA[1]



(1),(2),(3),(4),(5),(6),(7),(8),(9):TextBox1, TextBox2, TextBox3, TextBox4, TextBox5, TextBox6, TextBox7, TextBox8, TextBox9 (TextBox)

(10),(11),(12):OptionButton1,OptionButton2,OptionButton3 (OptionButton)

(13): Operarios (Frame)

(14),(15),(16),(17),(18),(19): ActualizarEstación,Volver,AñadirEstación, EliminarEstación,Subestaciones,Operarios (CommandButton)

[1] Existen dos interfaces gráficas FrmEstaciones y Festaciones similares. Festaciones se "copia y pega" en tiempo de ejecución cuando se crean nuevas estaciones. Los elementos de ambas tienen los mismos nombres.

VARIABLES

AvCopia

TIPO: Avion

DESCRIPCIÓN: variable en la que se almacenan los cambios hechos en los formularios, hasta que finalmente se "aceptan" y se registran en el avión correspondiente. Sirve también para poder utilizar el mismo formulario para el avión de referencia y los especiales, porque la inicializamos copiando los datos del avión que queremos mostrar.

MÉTODOS

UserForm_Initialize()

DESCRIPCIÓN: método que prepara y muestra el formulario estaciones de forma correcta.

LLAMADAS ENTRANTES:

Estaciones_Click

LLAMADAS SALIENTES:

GestAv.CopiarAviones av, AvCopia

GestFrmEstaciones.obtenerDatosEstacion 1, 0; GestFrmEstaciones.obtenerDatosEstacion nx, 1

GestFrmEstaciones.añadirEstación

CÓDIGO:

```
Private Sub UserForm_Initialize()
 Dim i As Integer
 Dim n As Integer
 Dim st As Pages
 Dim npag As Integer
Dim nx As Integer
Dim av As Avion
Dim nms As Integer
Set av = GestFrmEstaciones.av
 GestAv.CopiarAviones av, AvCopia
 nms = av.NumeroMaximoSubestaciones
 FrmEstaciones.Caption = " Estaciones del Avión " & av.NumeroAvion
 Set st = FrmEstaciones.Multiestacion.Pages
 npag = FrmEstaciones.Multiestacion.Value
 ReDim GestFrmEstaciones.pgs(0)
 Set GestFrmEstaciones.pgs(0) = st.Item(0)
 If Not av.Especial Then
  n = FrmGeneral.NE.Text
  If n = 0 Then
   n = 1
```

```
  FrmGeneral.NE.Text = n
 End If
Else
 n = av.NumeroEstaciones
End If
 GestFrmEstaciones.obtenerDatosEstacion 1, 0
   nx = n
  For i = 2 To n
   GestFrmEstaciones.añadirEstación
   GestFrmEstaciones.obtenerDatosEstacion nx, 1
   nx = nx - 1
  Next I
End Sub
```

AñadirEstacion_Click

DESCRIPCIÓN: método que responde al evento de pulsar sobre el botón de añadir estaciones, llama al método que gestiona esta acción y actualiza las variables del programa conforme a los nuevos valores.

LLAMADAS SALIENTES:

AvCopia.Actualiza1D NE - 1, numeropagina + 2

AvCopia.Actualiza2D NE - 1, nms, numeropagina + 2

AvCopia.Actualiza3D NE - 1, nms, numeropagina + 2

GestFrmEstaciones.añadirEstación

GestFrmEstaciones.obtenerDatosEstacion numeropagina + 2, numeropagina + 1

CÓDIGO:

```
Private Sub AñadirEstacion_Click()
 Dim numeropagina As Integer
 Dim NE As Integer
 Dim nms As Integer
  numeropagina = FrmEstaciones.Multiestacion.Value
 NE = AvCopia.NumeroEstaciones + 1
 nms = AvCopia.NumeroMaximoSubestaciones
AvCopia.Actualiza1D NE - 1, numeropagina + 2
 AvCopia.Actualiza2D NE - 1, nms, numeropagina + 2
 AvCopia.Actualiza3D NE - 1, nms, numeropagina + 2
 GestFrmEstaciones.añadirEstación
 GestFrmEstaciones.obtenerDatosEstacion numeropagina + 2, numeropagina + 1
 End Sub
```

BotonOperarios_Click

DESCRIPCIÓN: carga y presenta de forma correcta el menú de operarios.

LLAMADAS SALIENTES:

FrmOperarios.Show

CÓDIGO:

```
Private Sub BotonOperarios_Click()
 Dim n As Integer
 Dim numsubestaciones As TextBox
 Dim st As Pages
 Dim pg As Page
 Dim numpag As Integer
  Set st = FrmEstaciones.Multiestacion.Pages
 numpag = FrmEstaciones.Multiestacion.Value
 Set pg = st.Item(numpag)
 Set numsubestaciones = pg.Controls.Item(5)
 n = numsubestaciones.Text
 If (n = 0) Then
   FrmOperarios.Show
 Else
  MsgBox " Para visualizar Datos entrar en Subestaciones", vbOKOnly
 End If
End Sub
```

Volver_Click

DESCRIPCIÓN: descargamos el formulario sin realizar cambios

CÓDIGO:

```
Private Sub Volver_Click()
 Unload Me
End Sub
```

Subestaciones_Click

DESCRIPCIÓN: cargamos el formulario de subestaciones., asegurándonos que tenemos subestaciones y en ese caso, preparando los datos a mostrar

LLAMADAS SALIENTES:

GestFrmEstaciones.insertarDatosEstacion numpag + 1

FrmSubEstaciones.Show

CÓDIGO:

```
Private Sub Subestaciones_Click()
 Dim st As Pages
 Dim pg As Page
 Dim i As Integer
 Dim numpag As Long
```

```
 Dim n As Integer
 Dim numsubestaciones As TextBox
GestFrmSubestaciones.modo = 1
 Set st = FrmEstaciones.Multiestacion.Pages
 numpag = FrmEstaciones.Multiestacion.Value
 FrmSubEstaciones.Caption = " Subestaciones de la estación " & (numpag + 1)
 Set pg = st.Item(numpag)
 Set numsubestaciones = pg.Controls.Item(5)
 n = numsubestaciones.Text
 If (n <> 0) Then
 GestFrmEstaciones.insertarDatosEstacion numpag + 1
 FrmSubEstaciones.Show
 Else
  MsgBox " No hay subestaciones para esta estación", vbOKOnly
  Unload FrmSubEstaciones
 End If
End Sub
```

EliminarEstacion_Click

DESCRIPCIÓN: llamamos al procedimiento que gestiona la eliminación y actualizamos las variables si no ha habido ningún error.

LLAMADAS SALIENTES:

GestFrmEstaciones.EliminarEstacion

AvCopia.Actualiza1D NE - 1, numpag + 1, True

AvCopia.Actualiza2D NE - 1, nms, numpag + 1, True

AvCopia.Actualiza3D NE - 1, nms, numpag + 1, True

CÓDIGO

```
Private Sub EliminarEstacion_Click()
 Dim numpag As Integer
 Dim n As Integer
 Dim NE As Integer
 Dim nms As Integer
  numpag = FrmEstaciones.Multiestacion.Value
 If FrmEstaciones.Multiestacion.Count = 1 Then
  Exit Sub
 End If
GestFrmEstaciones.EliminarEstacion
 NE = AvCopia.NumeroEstaciones - 1
 nms = AvCopia.NumeroMaximoSubestaciones
 AvCopia.Actualiza1D NE - 1, numpag + 1, True
```

AvCopia.Actualiza2D NE - 1, nms, numpag + 1, True
AvCopia.Actualiza3D NE - 1, nms, numpag + 1, True
End Sub

ActualizarEstacion_Click

DESCRIPCIÓN: procedimiento que captura la actualización de datos por parte del usuario. Se introducen los datos nuevos en las variables internas del avión copiaia, se comprueban si son correctos y si es así se procede a actualizar las variables del avión correspondiente.

LLAMADAS SALIENTES:

GestFrmEstaciones.insertarDatosEstacion n
GestFrmEstaciones.ActualizarFrmOperariosEstaciones AvCopia
Control.ControlDatosValidos(AvCopia)
GestAv.CopiarAviones AvCopia, GestFrmEstaciones.av
Control.EnlacesDefecto GestFrmEstaciones.av

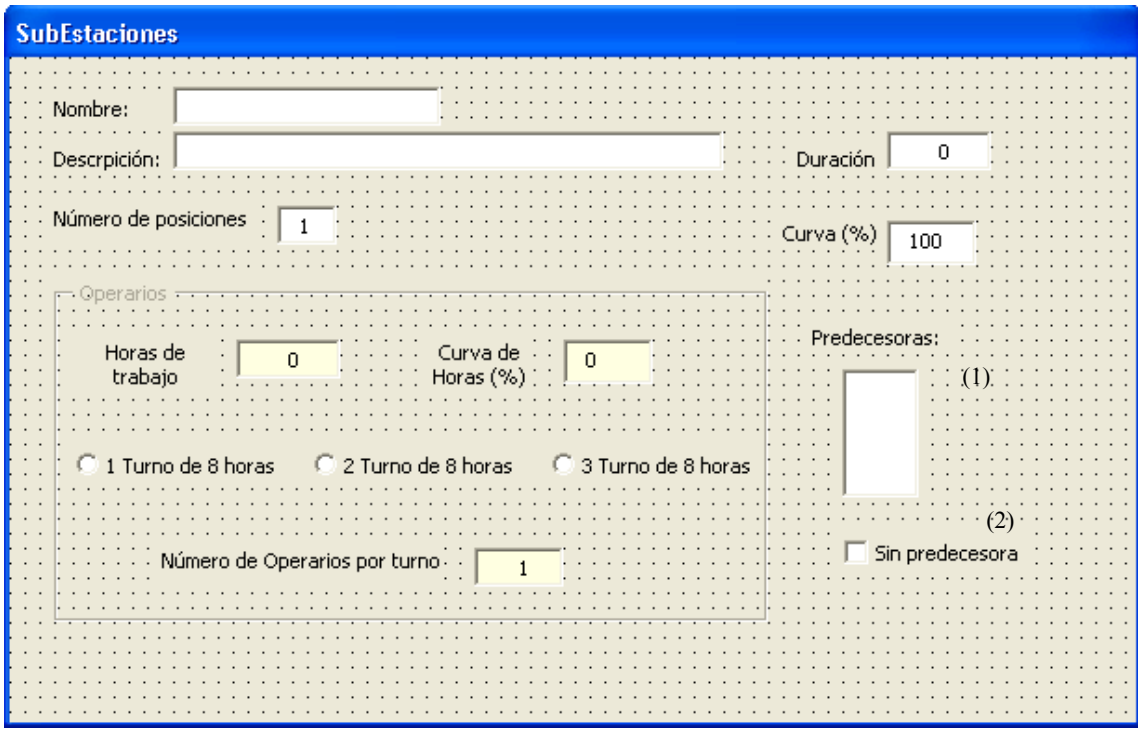
CÓDIGO:

```
Private Sub ActualizarEstacion_Click()
On Error GoTo fin:
Dim st As Pages
Dim numpag As Long
Dim n As Integer
Dim nms As Integer
 Set st = FrmEstaciones.Multiestacion.Pages
 numpag = st.Count
 If Not GestFrmEstaciones.av.Especial Then
    FrmGeneral.NE.Text = numpag
  Else
     FrmAvEspecial.NEst = numpag
 End If
 For n = 1 To numpag
 GestFrmEstaciones.insertarDatosEstacion n
 Next
 GestFrmEstaciones.ActualizarFrmOperariosEstaciones AvCopia
If Control.ControlDatosValidos(AvCopia) = False Then
 GoTo fin
End If
GestAv.CopiarAviones AvCopia, GestFrmEstaciones.av
If GestFrmEstaciones.av.Especial = False Then
 Control.EnlacesDefecto GestFrmEstaciones.av
End If
Exit Sub
```

```
fin: MsgBox " Dato/s incorrecto/s"
End Sub
```

FrmSubEstaciones

INTERFAZ GRÁFICA[2]



(1),(2): ListBox1 (ListBox),**CheckBox1** (CheckBox)

VARIABLES

NEst

TIPO: Integer

DESCRIPCIÓN: variable que se inicializa en el método UserForm_Initializa y que almacena el número de estación en el que nos encontramos.

[2] Vemos el formulario FSubEstaciones, que se copia y pega en tiempo de ejecución. Es un formulario con más elemento que FrmSubEstaciones, con lo que los componentes no tienen los mismos nombres y eso hay que tenerlo en cuenta en el código. El código se encuentra asociado a FrmSubestaciones, que es un multipage al estilo de FrmEstaciones. Sólo indicamos los elementos nuevos respecto a FrmEstaciones

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: método que crea el formulario que se va a visualizar con los datos inicializados correctamente.

LLAMADAS ENTRANTES:

FrmEstaciones. Subestaciones_Click()

LLAMADAS SALIENTES:

av.Actualiza2D NE - 1, Ns

av.Actualiza3D NE - 1, Ns

GestFrmSubestaciones.obtenerDatosSubestacion numpag, 1, 0;

GestFrmSubestaciones.obtenerDatosSubestacion numpag, nxs, 1

GestFrmSubestaciones.añadirSubEstación

CÓDIGO:

```
Private Sub UserForm_Initialize()
 Dim st As Pages
 Dim pg As Page
 Dim i As Integer
 Dim numpag As Integer
 Dim Ns As Integer
 Dim numsubestaciones As TextBox
 Dim nxs As Integer
 Dim substations As Pages
 Dim av As Avion
 Dim NE As Integer
 Set av = FrmEstaciones.AvCopia
 NE = av.NumeroEstaciones
 Set st = FrmEstaciones.Multiestacion.Pages
 numpag = FrmEstaciones.Multiestacion.Value
 NEst = numpag
 FrmSubEstaciones.Caption = " Subestaciones de la estación " & (numpag + 1)
 Set pg = st.Item(numpag)
 Set numsubestaciones = pg.Controls.Item(5)
 ReDim GestFrmSubestaciones.pgs(0)
 Set substations = FrmSubEstaciones.MultiSubestacion.Pages
 Set GestFrmSubestaciones.pgs(0) = substations.Item(0)
 Ns = numsubestaciones.Text
 If (Ns > av.NumeroMaximoSubestaciones) Then
  av.Actualiza2D NE - 1, Ns
  av.Actualiza3D NE - 1, Ns
```

```
 End If
 GestFrmSubestaciones.obtenerDatosSubestacion numpag, 1, 0
 nxs = Ns
 For i = 2 To Ns
   GestFrmSubestaciones.añadirSubEstación
   Dim lb As ListBox
   Dim j As Integer
   Set lb = GestFrmSubestaciones.pgs(1).Controls.Item(5)
   For j = 0 To nxs - 2
    lb.AddItem j + 1
   Next j
   GestFrmSubestaciones.obtenerDatosSubestacion numpag, nxs, 1
   nxs = nxs - 1
   Next i
End Sub
```

ActualizarSubestacion_Click

DESCRIPCIÓN: se almacenan en las variables internas del programa los datos introducidos por el usuario. Sólo se actualiza la subestación correspondiente.

LLAMADAS SALIENTES:

GestFrmSubestaciones.insertarDatosSubestacion NEst, NSubest + 1

GestFrmSubestaciones.ActualizarFrmOperariosEstaciones FrmEstaciones.AvCopia, NEst, NSubest + 1

CÓDIGO:

```
Private Sub ActualizarSubestacion_Click()
On Error GoTo fin:
Dim NSubest As Integer
NSubest = FrmSubEstaciones.MultiSubestacion.Value
GestFrmSubestaciones.insertarDatosSubestacion NEst, NSubest + 1
GestFrmSubestaciones.ActualizarFrmOperariosEstaciones FrmEstaciones.AvCopia, NEst, NSubest + 1
Exit Sub
fin:
  MsgBox "Dato/s Incorrecto/s"
End Sub
```

BOperarios_Click

DESCRIPCIÓN: muestra el formulario de operarios.

LLAMADAS SALIENTES:

FrmOperarios.UserFormInitialize

CÓDIGO:

```
Private Sub BOperarios_Click()
```

```
 FrmOperarios.Show
End Sub
```

Volver_Click

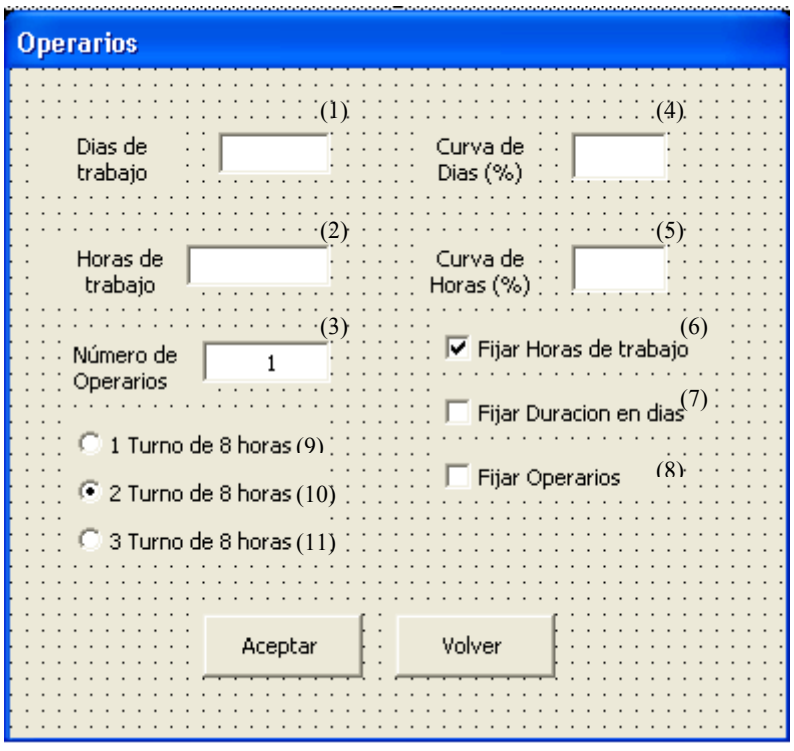
DESCRIPCIÓN: descargamos el formulario e indicamos (variable modo) que salimos del submenú de subestaciones.

CÓDIGO:

```
Private Sub Volver_Click()
 GestFrmSubestaciones.modo = 0
 Unload Me
 End Sub
```

FrmOperarios

INTERFAZ GRÁFICA



(1),(2),(3),(4),(5): TBDT, TBHT,TBOp,TBCD,TBCH (TextBox)

(6),(7),(8): CbFH,CbFD,CbFOp (CheckBox)

(9),(10),(11): OB1,OB2,OB3 (OptionButton)

VARIABLES

NEst

TIPO: Integer

DESCRIPCIÓN: número de estación.

NSubEst

TIPO: Integer

DESCRIPCIÓN: número de subestación.

modo

TIPO: Integer

DESCRIPCIÓN: nos permite distinguir si la información pertenece a una estación (modo=0) o una subestación (modo=1).

pg

TIPO: Page

DESCRIPCIÓN: control en el que tenemos los datos de la estación o subestación en la que estamos.

turnoActual

TIPO: Integer

DESCRIPCIÓN: valor entre 1,2 ó 3 que indica el número de turnos de los operarios.

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: presenta la vantana de operarios en función de la ventana precedente.

LLAMADAS ENTRANTES:

FrmEstaciones.BotonOperarios_Click()

FrmSubEstaciones.BOperarios_Click()

CÓDIGO:

```
Private Sub UserForm_Initialize()
 Dim st As Pages
 Dim OPE As Frame
 Dim CurvDias As TextBox
 Set st = FrmEstaciones.Multiestacion.Pages
 NEst = FrmEstaciones.Multiestacion.Value
 NSubest = 0
 If GestFrmSubestaciones.modo = 1 Then
```

```
 modo = 1
 Else
  modo = 0
 End If
 If modo = 0 Then
  FrmOperarios.Caption = " Operarios del avión " & GestFrmEstaciones.av.NumeroAvion & " Estación "
& NEst + 1
  Set pg = FrmEstaciones.Multiestacion.Pages.Item(NEst)
  Set OPE = pg.Controls.Item(12)
  Set CurvDias = pg.Controls.Item(7)
 Else
  NSubest = FrmSubEstaciones.MultiSubestacion.Value + 1
  Set pg = FrmSubEstaciones.MultiSubestacion.Pages.Item(NSubest - 1)
  FrmOperarios.Caption = " Operarios del avión " & GestFrmEstaciones.av.NumeroAvion & " Estación "
& NEst + 1 & " Subestación " & NSubest
  If NSubest = 1 Then
  Set OPE = pg.Controls.Item(10)
  Set CurvDias = pg.Controls.Item(5)
  Else
   Set OPE = pg.Controls.Item(13)
   Set CurvDias = pg.Controls.Item(8)
  End If
 End If
TBDT.Text = pg.Controls.Item(3).Text
TBCD.Text = CurvDias.Text
TBHT.Text = OPE.Controls.Item(1).Text
TBCH.Text = OPE.Controls.Item(3).Text
OB1.Value = OPE.Controls.Item(6).Value
OB2.Value = OPE.Controls.Item(7).Value
OB3.Value = OPE.Controls.Item(8).Value
TBOp.Text = OPE.Controls.Item(5).Text
turnoActual = 2
If OB1.Value = True Then
 turnoActual = 1
End If
If OB2.Value = True Then
 turnoActual = 2
End If
If OB3.Value = True Then
 turnoActual = 3
```

```
End If
Aceptar.SetFocus
End Sub
```

Aceptar_Click

DESCRIPCIÓN: se actualizan en la ventana de la estación o subestación correspondiente los nuevos valores fijados de duraciones y operarios.

CÓDIGO:

```
Private Sub Aceptar_Click()
On Error GoTo fin:
Dim TBDias As TextBox
Dim CurvDias As TextBox
Dim TBHoras As TextBox
Dim TBCHO As TextBox
Dim OBP1 As OptionButton
Dim OBP2 As OptionButton
Dim OBP3 As OptionButton
Dim TBOpO As TextBox
Dim TurnoEst As Integer
Dim OPE As Frame
If modo = 0 Then
Set OPE = pg.Controls.Item(12)
Set CurvDias = pg.Controls.Item(7)
Else
If NSubest = 1 Then
  Set OPE = pg.Controls.Item(10)
  Set CurvDias = pg.Controls.Item(5)
 Else
  Set OPE = pg.Controls.Item(13)
  Set CurvDias = pg.Controls.Item(8)
 End If
End If
Set TBDias = pg.Controls.Item(3)
Set TBHoras = OPE.Controls.Item(1)
Set TBCHO = OPE.Controls.Item(3)
Set OBP1 = OPE.Controls.Item(6)
Set OBP2 = OPE.Controls.Item(7)
Set OBP3 = OPE.Controls.Item(8)
Set TBOpO = OPE.Controls.Item(5)
TBDias.Text = TBDT.Text
```

```
CurvDias.Text = TBCD.Text
TBHoras.Text = TBHT.Text
TBCHO.Text = TBCH.Text
OBP1.Value = OB1.Value
OBP2.Value = OB2.Value
OBP3.Value = OB3.Value
TBOpO.Text = TBOp.Text
Exit Sub
fin:
  MsgBox "Dato/s Incorrecto/s"
End Sub
```

CbFD_Change

CÓDIGO:

```
Private Sub CbFD_Change()
  If CbFD.Value = True Then
  CbFH.Value = False
  CbFOp.Value = False
 End If
End Sub
```

CbFH_Change

CÓDIGO:

```
Private Sub CbFH_Change()
 If CbFH.Value = True Then
  CbFD.Value = False
  CbFOp.Value = False
 End If
End Sub
```

CbFOp_Change

CÓDIGO:

```
Private Sub CbFOp_Change()
 If CbFOp.Value = True Then
  CbFH.Value = False
  CbFD.Value = False
 End If
End Sub
```

OB1_Click

CÓDIGO:

```
Private Sub OB1_Click()
 On Error GoTo fin:
 Dim Operarios As Single
 Dim horas As Single
 Dim Dias As Single
 CbFD.Enabled = True
 CbFH.Enabled = True
 CbFOp.Enabled = False
 If CbFH.Value = False And CbFD.Value = False Then
  CbFH.Value = True
 End If
 Operarios = TBOp.Text
 Dias = TBDT.Text
 horas = TBHT.Text
 If OB1.Value = True Then
  turnoActual = 1
 End If
 If CbFH.Value = True Then
  Dias = horas / (Operarios * turnoActual * 8)
  TBDT.Text = Dias
 End If
 If CbFD.Value = True Then
  horas = Dias * (Operarios * turnoActual * 8)
  TBHT.Text = horas
 End If
Exit Sub
fin: MsgBox "Division por cero!!"
End Sub
```

OB2_Click

CÓDIGO:

```
Private Sub OB2_Click()
 On Error GoTo fin:
 Dim Operarios As Single
 Dim horas As Single
 Dim Dias As Single
 CbFD.Enabled = True
 CbFH.Enabled = True
 CbFOp.Enabled = False
```

```
 If CbFH.Value = False And CbFD.Value = False Then
  CbFH.Value = True
 End If
 Operarios = TBOp.Text
 Dias = TBDT.Text
 horas = TBHT.Text
 If OB2.Value = True Then
turnoActual = 2
 End If
 If CbFH.Value = True Then
  Dias = horas / (Operarios * turnoActual * 8)
  TBDT.Text = Dias
 End If
 If CbFD.Value = True Then
  horas = Dias * (Operarios * turnoActual * 8)
  TBHT.Text = horas
 End If
 Exit Sub
fin: MsgBox "Division por cero!!"
End Sub
```

OB3_Click

CÓDIGO:

```
Private Sub OB3_Click()
On Error GoTo fin:
Dim Operarios As Single
Dim horas As Single
Dim Dias As Single
 CbFD.Enabled = True
 CbFH.Enabled = True
 CbFOp.Enabled = False
 If CbFH.Value = False And CbFD.Value = False Then
  CbFH.Value = True
 End If
 Operarios = TBOp.Text
 Dias = TBDT.Text
 horas = TBHT.Text
 If OB3.Value = True Then
turnoActual = 3
 End If
```

```
 If CbFH.Value = True Then
 Dias = horas / (Operarios * turnoActual * 8)
 TBDT.Text = Dias
End If
If CbFD.Value = True Then
 horas = Dias * (Operarios * turnoActual * 8)
 TBHT.Text = horas
End If
Exit Sub
fin: MsgBox "Division por cero!!"
End Sub
```

TBDT_AfterUpdate

CÓDIGO:

```
Private Sub TBDT_AfterUpdate()
 On Error GoTo fin:
 Dim horas As Single
Dim Dias As Single
Dim Operarios As Single
 Dias = TBDT.Text
horas = TBHT.Text
Operarios = TBOp.Text * turnoActual
 If CbFH.Value = True Then
Operarios = horas / (Dias * 8)
TBOp.Text = Operarios
Else
horas = Dias * Operarios * 8
TBHT.Text = horas
End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
End Sub
```

TBDT_Enter

CÓDIGO:

```
Private Sub TBDT_Enter()
CbFD.Enabled = False
CbFH.Enabled = True
CbFOp.Enabled = True
If CbFH.Value = False And CbFOp.Value = False Then
```

```
 CbFH.Value = True
 End If
End Sub
```

TBHT_AfterUpdate

CÓDIGO:

```
Private Sub TBHT_AfterUpdate()
On Error GoTo fin:
 Dim horas As Single
 Dim Dias As Single
 Dim Operarios As Single
 Dias = TBDT.Text
 horas = TBHT.Text
 Operarios = TBOp.Text * turnoActual
 If CbFD.Value = True Then
 Operarios = horas / (Dias * 8)
 TBOp.Text = Operarios
 Else
 Dias = horas / (Operarios * 8)
 TBDT.Text = Dias
 End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
End Sub
```

TBHT_Enter

CÓDIGO:

```
Private Sub TBHT_Enter()
 CbFD.Enabled = True
 CbFH.Enabled = False
 CbFOp.Enabled = True
 If CbFD.Value = False And CbFOp.Value = False Then
  CbFD.Value = True
 End If
 End Sub
```

TBOp_AfterUpdate

CÓDIGO:

```
Private Sub TBOp_AfterUpdate()
 On Error GoTo fin:
```

```
 Dim horas As Single
 Dim Dias As Single
 Dim Operarios As Single
 Dias = TBDT.Text
 horas = TBHT.Text
 Operarios = TBOp.Text * turnoActual
 If CbFH.Value = True Then
 Dias = horas / (Operarios * 8)
 TBDT.Text = Dias
 Else
 horas = Dias * Operarios * 8
 TBHT.Text = horas
 End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
End Sub
```

TBOp_Enter

CÓDIGO:

```
Private Sub TBOp_Enter()
 CbFD.Enabled = True
 CbFH.Enabled = True
 CbFOp.Enabled = False
 If CbFH.Value = False And CbFD.Value = False Then
  CbFH.Value = True
 End If
End Sub
```

Volver_Click

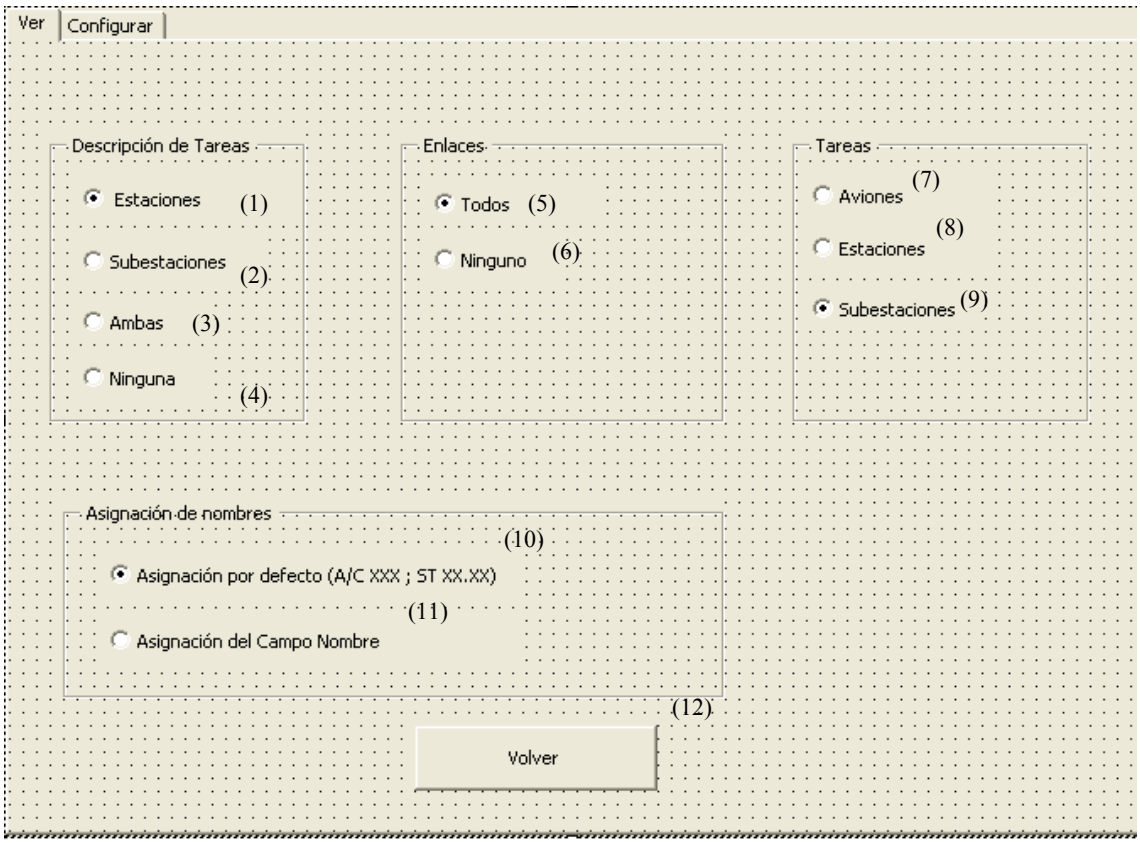
CÓDIGO:

```
Private Sub Volver_Click()
 Unload Me
End Sub
```

MENÚ OPCIONES (FrmOpciones)

INTERFAZ GRÁFICA

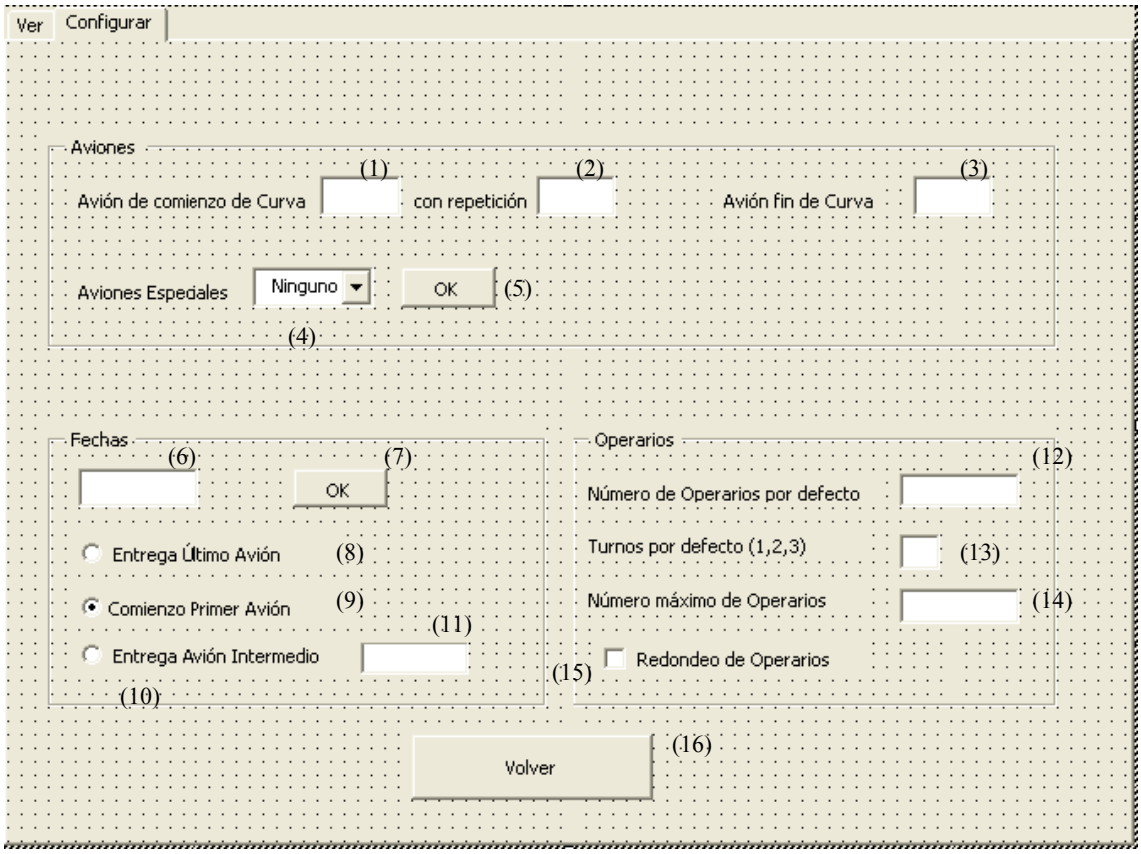
MENÚ OPCIONES.VER



(1),(2),(3),(4),(5),(6),(7),(8),(9),(10),(11),(12):DescEstaciones,DescSubestaciones, DescAmbas,DescNinguna,EnlaceTodos,EnlaceNinguno,TareaAviones,Tarea-Subestaciones,AsignaDefecto,AsignaCampo. (OptionButton)

(12): Aceptar. (CommandButton)

MENÚ OPCIONES.CONFIGURAR



(1),(2),(3),(6),(11),(12),(13),(14): CC,NRCC,TBAFC, TextFecha, NAI,TBOpDef, TBOpMax. (TextBox)

(5),(7),(16): OKButton,OKFecha,CommandButton1. (CommandButton)

(8),(9),(10): CUA,CPA,CAI. (OptionButton)

(15): CBRedondeo. (CheckBox)

(4): AvionesEspeciales. (ComboBox)

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: se inicializan los componentes con los valores obtenidos en el módulo de inicialización.

CÓDIGO:

```
Private Sub UserForm_Initialize()
Dim L As ComboBox
Dim i As Integer
Set L = FrmOpciones.AvionesEspeciales
```

```
CC.Text = planning.AvCC
NRCC.Text = planning.NumeroRepAvCC
TBOpDef = planning.OpDefecto
TBTDef.Text = planning.TurnoDefecto
TBOpMax = planning.OpMax
TBAVFC.Value = planning.AvFC
If GestAv.NumeroAvEspeciales > 0 Then
For i = 0 To UBound(GestAv.AvEspeciales) - 1
 L.AddItem (GestAv.AvEspeciales(i))
Next
End If
TextFecha.Text = Inicializacion.FFecha
 If Inicializacion.AFecha = 1 Then
   CPA.Value = True
 Else
   CUA.Value = True
 End If
If CAI.Value Then
   NAI.Visible = True
 Else
   NAI.Visible = False
 End If
End Sub
```

UserForm_QueryClose

DESCRIPCIÓN: al ser el formulario de opciones, un formulario siempre residente en memoria, hay que evitar que no se muestre en pantalla ,con excepción de cuando se solicita.

CÓDIGO:

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
If CloseMode = vbFormControlMenu Then
 FrmOpciones.Enabled = False
End If
End Sub
```

Aceptar_Click

DESCRIPCIÓN: no se han reservado variables en memoria para guardar las opciones. Por lo tanto,no se descarga (unload), sino que se oculta (hide), para poder tener accesible sus controles.

CÓDIGO:

```
Private Sub Aceptar_Click()
hide
```

End Sub

CC_AfterUpdate

DESCRIPCIÓN: se actualizan variables y se cumprueba que el dato introducido es válido.El avión de comienzo de curva no puede ser ni especial ni estar fuera de rango.

LLAMADAS SALIENTES:

isEspecial(CC.Text)

GestAv.CambioAvionReferencia(planning.RepRef, planning.NumeroRepAvCC, CC.Text)

CÓDIGO:

```
Private Sub CC_AfterUpdate()
On Error GoTo fin
If isEspecial(CC.Text) Then
  MsgBox "Este avion esta marcado como especial"
  CC.Text = planning.AvCC
  Exit Sub
End If
If GestAv.CambioAvionReferencia(planning.RepRef, planning.NumeroRepAvCC, CC.Text) And CC.Text > 0 Then
 Else
   MsgBox " No puede ser superior al número de aviones!! "
  CC.Text = planning.AvCC
End If
Exit Sub
fin:  MsgBox "  Dato incorrecto "
  CC.Text = planning.AvCC
End Sub
```

CommandButton1_Click

DESCRIPCIÓN: ocultamos sin descargar el formulario, para preservar los datos.

CÓDIGO:

```
Private Sub CommandButton1_Click()
 hide
End Sub
```

NRCC_AfterUpdate

DESCRIPCIÓN: cambiamos el número de repetición del avión dew comienzo de curva si el dato introducido es válido.

LLAMADAS SALIENTES:

GestAv.CambioAvionReferencia(planning.RepRef, NRCC.Text, planning.AvCC)

CÓDIGO:

```
Private Sub NRCC_AfterUpdate()
On Error GoTo fin
If GestAv.CambioAvionReferencia(planning.RepRef, NRCC.Text, planning.AvCC) And NRCC > 0 Then
```

```
Else
  'MsgBox " La repetición del avión de comienzo de curvar debe ser menor que la de referencia y estar en un margen válido", vbOKOnly
  'NRCC.Text = planning.NumeroRepAvCC
End If
Exit Sub
fin:
    MsgBox "    Dato incorrecto"
    NRCC.Text = planning.NumeroRepAvCC
End Sub
```

CAI_Change

DESCRIPCIÓN: control para posible opción añadida del programa que permitiria fijar la fecha de una avión intermedoi de la curva. Opción incompatible con la fijación de la fecha de otro avión.

CÓDIGO:

```
Private Sub CAI_Change()
 If CAI.Value Then
   NAI.Visible = True
 Else
   NAI.Visible = False
 End If
End Sub
```

OKButton_Click

DESCRIPCIÓN: tras comprobar que el avión puede ser especial, se añade a la lista de aviones especiales y se inicializan sus variables. Posteriormente mostramos el formulario de geetión de aviones especiales.

LLAMADAS SALIENTES:

GestAv.AñadirAvionEspecial(val)

GestAv.OrdenarAvionesEspeciales v

Control.EnlacesDefecto GestAv.aviones(ave + 1);Control.EnlacesDefecto GestAv.aviones(ave)

FrmAvEspecial.Show

CÓDIGO:

```
Private Sub OKButton_Click()
Dim ave As Integer
Dim a As Boolean
Dim v() As Integer
Dim L As ComboBox
Dim i As Integer
Dim val As Integer
```

```
On Error GoTo fin
ave = AvionesEspeciales.Value
If Not FrmOpciones.Enabled Then
  Exit Sub
End If
 If ave > 0 And ave <= planning.NumeroAviones Then
 Set L = FrmOpciones.AvionesEspeciales
 val = L.Value
 If val = AvRR.NumeroAvion Then
   MsgBox "El avion no puede ser especial puesto que es el de referencia de la curva"
   Exit Sub
 End If
 a = GestAv.AñadirAvionEspecial(val)
 If a Then
 L.AddItem val
 ReDim v(L.ListCount - 1)
 For i = 0 To UBound(v)
   v(i) = L.list(i, 0)
 Next
 GestAv.OrdenarAvionesEspeciales v
 For i = 0 To UBound(v)
  L.list(i, 0) = v(i)
 Next
 L.Value = val
If ave + 1 <= planning.NumeroAviones Then
 If Not isEspecial(ave + 1) Then
  GestAv.aviones(ave + 1).EE = True
  GestAv.aviones(ave + 1).NumeroAvion = ave + 1
  GestAv.aviones(ave + 1).NumeroEstaciones = AvRR.NumeroEstaciones
  GestAv.aviones(ave + 1).NumeroMaximoSubestaciones = AvRR.NumeroMaximoSubestaciones
  GestAv.aviones(ave + 1).Estaciones = AvRR.Estaciones
  GestAv.aviones(ave + 1).EnlaceEstaciones = AvRR.EnlaceEstaciones
  GestAv.aviones(ave + 1).EnlaceSubestaciones = AvRR.EnlaceSubestaciones
  Control.EnlacesDefecto GestAv.aviones(ave + 1)
  End If
End If
Control.EnlacesDefecto GestAv.aviones(ave)
 End If
FrmAvEspecial.Caption = "Avión Especial " & ave
FrmAvEspecial.Show
```

```
Else
  MsgBox "Avion especial incorrecto. Debe estar en un margen correcto", vbOKOnly
  Exit Sub
End If
If Not FrmOpciones.Enabled Then
  Unload FrmAvEspecial
 End If
Exit Sub
fin:
   MsgBox "No se ha Introducido un dato correcto", vbOKOnly
End Sub
```

OKFecha_Click

DESCRIPCIÓN: en función de la opción elegida de fecha (primer o últimmo avión)inicializamos la variable correspondiente.

CÓDIGO:

```
Private Sub OKFecha_Click()
 On Error GoTo fin:
  Inicializacion.FFecha = TextFecha.Text
  If CUA Then
   Inicializacion.AFecha = planning.NumeroAviones
  Else
   Inicializacion.AFecha = 1
  End If
 Exit Sub
fin:
    MsgBox " Formato de fecha incorrecto!!"
    TextFecha.Text = Inicializacion.FFecha
End Sub
```

TBAVFC_AfterUpdate

CÓDIGO:

```
Private Sub TBAVFC_AfterUpdate()
On Error GoTo fin
Dim i As Integer
i = TBAVFC.Value
If i <= 0 Or i < planning.AvCC Then
 GoTo fin
End If
planning.AvFC = i
```

```
Exit Sub
fin: MsgBox " Dato incorrecto"
TBAVFC.Value = planning.AvFC
End Sub
```

TBOpDef_AfterUpdate

CÓDIGO:

```
Private Sub TBOpDef_AfterUpdate()
 On Error GoTo fin:
  If TBOpDef.Text > planning.OpMax Or TBOpDef.Text < 0 Then
   GoTo fin
  Else
   planning.OpDefecto = TBOpDef.Text
  End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
    TBOpDef.Text = planning.OpDefecto
End Sub
```

TBOpMax_AfterUpdate

CÓDIGO:

```
Private Sub TBOpMax_AfterUpdate()
 On Error GoTo fin:
  If TBOpMax.Text < planning.OpDefecto Or TBOpMax.Text < 0 Then
   GoTo fin
  Else
   planning.OpMax = TBOpMax.Text
  End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
    TBOpMax.Text = planning.OpMax
End Sub
```

TBTDef_Change

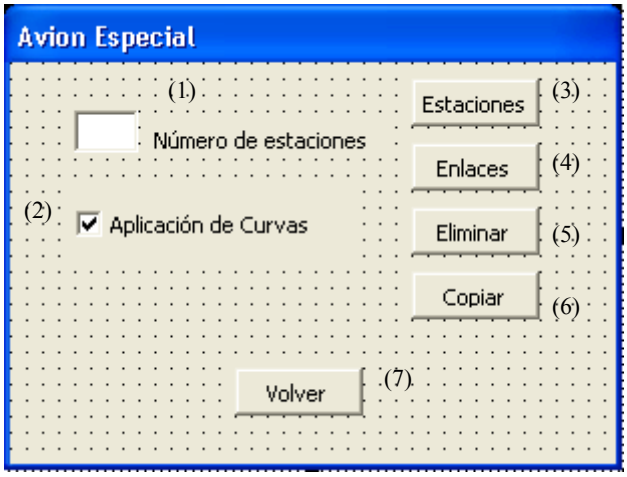
CÓDIGO:

```
Private Sub TBTDef_Change()
On Error GoTo fin:
  If TBTDef.Text > 3 Or TBOpDef.Text < 0 Then
   GoTo fin
  Else
```

```
  planning.TurnoDefecto = TBTDef.Text
 End If
 Exit Sub
fin:  MsgBox " Dato incorrecto"
    TBTDef.Text = planning.TurnoDefecto
End Sub
```

FrmAvEspecial

INTERFAZ GRÁFICA



(3),(4),(5),(6),(7): Estaciones, EnlacesEspeciales, Eliminar, Copiar, Volver. (CommandButton).

(1): Nest. (TextBox)

(2): CbCurva. (CheckBox)

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: comprobamos el número de avión y si se le aplica o no curvas. Preparamos el formulario de forma correcta.

LLAMADAS ENTRANTES:

FrmOpciones.OKButton_Click

LAMADAS SALIENTES:

isEspecial(ave)

CÓDIGO:

```
Private Sub UserForm_Initialize()
Dim ave As Integer
On Error GoTo fallo
ave = FrmOpciones.AvionesEspeciales.Value
CbCurva.Visible = True
If ave > planning.AvCC Then
   CbCurva.Value = aviones(ave).AplicaCurva
  Else
   'CbCurva.Value = False
   'aviones(ave).AplicaCurva = False
   CbCurva.Visible = False
End If
If isEspecial(ave) Then
   NEst.Text = aviones(ave).NumeroEstaciones
Else
 NEst.Text = 1
End If
Exit Sub
fallo: MsgBox "No se ha Introducido un dato correcto", vbOKOnly
End Sub
```

Volver_Click

CÓDIGO:

```
Private Sub Volver_Click()
Unload Me
End Sub
```

CbCurva_Click

DESCRIPCIÓN: especificamos si al avión se le aplica curva o no.

CÓDIGO:

```
Private Sub CbCurva_Click()
Dim ave As Integer
ave = FrmOpciones.AvionesEspeciales.Value
  If CbCurva.Value = True Then
   GestAv.aviones(ave).AplicaCurva = True
  Else
   GestAv.aviones(ave).AplicaCurva = False
  End If
```

End Sub

Copiar_Click

LLAMADAS SALIENTES:

FrmCopiar.UserFormInitialize()

CÓDIGO:

Private Sub Copiar_Click()

FrmCopiar.Show

End Sub

EnlacesEspeciales_Click

LLAMADAS SALIENTES:

frmEnlacesEspeciales.UserFormInitialize()

CÓDIGO:

Private Sub EnlacesEspeciales_Click()

frmEnlacesEspeciales.Show

End Sub

Estaciones_Click

DESCRIPCIÓN: mostramos el formulario de estaciones con los datos del avión especial. Para ello el avión se copia en la variable del módulo que gestiona el formulario de estaciones.

LLAMADAS SALIENTES:

FrmEstaciones.UserFormInitialize()

CÓDIGO:

Private Sub Estaciones_Click()

Dim ave As Integer

ave = FrmOpciones.AvionesEspeciales.Value

Set GestFrmEstaciones.av = aviones(ave)

FrmEstaciones.Show

End Sub

Eliminar_Click

DESCRIPCIÓN: se elimina el avión de todas lo variables que lo indicaban como especial.

LLAMADAS SALIENTES:

GestAv.EliminarAvionEspecial(ave)

GestAv.OrdenarAvionesEspeciales v

isEspecial(ave + 1)

CÓDIGO:

Private Sub Eliminar_Click()

Dim a As Boolean

```
Dim v() As Integer
Dim L As ComboBox
Dim i As Integer
Dim indice As Integer
Dim ave As Integer
Set L = FrmOpciones.AvionesEspeciales
ave = L.Value
a = GestAv.EliminarAvionEspecial(ave)
If a Then
For indice = 0 To L.ListCount
 If L.Value = L.list(indice, 0) Then
   Exit For
 End If
Next
L.RemoveItem indice
 If L.ListCount > 0 Then
 ReDim v(L.ListCount - 1)
 For i = 0 To UBound(v)
   v(i) = L.list(i, 0)
 Next
 GestAv.OrdenarAvionesEspeciales v
 For i = 0 To UBound(v)
  L.list(i, 0) = v(i)
 Next
 L.Value = v(0)
 Else
  L.Value = "Ninguno"
 End If
 Unload FrmAvEspecial
If ave + 1 < planning.NumeroAviones Then
 If Not isEspecial(ave + 1) Then
  GestAv.aviones(ave + 1).EE = False
  End If
End If
Else
 ' Nunca se deberia de dar este caso
 MsgBox " El avión NO está marcado como especial", vbOKOnly
End If
End Sub
```

NEst_Change

LLAMADAS SALIENTES:

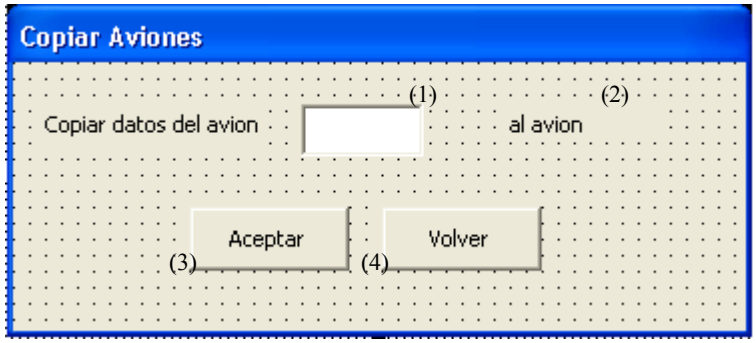
isEspecial(ave)

CÓDIGO:

```
Private Sub NEst_Change()
Dim ave As Integer
Dim buc As Integer
On Error GoTo fin
ave = FrmOpciones.AvionesEspeciales.Value
If NEst > 0 Then
 If Not isEspecial(ave) Then
 If NEst.Enabled = True Then
  MsgBox " El avión  debe estar indicado como especial!!", vbOKOnly
  NEst.Enabled = False
  NEst.Text = 1
  End If
 NEst.Enabled = True
 Exit Sub
 Else
   aviones(ave).NumeroEstaciones = NEst.Text
 End If
Else
fin:   MsgBox "El número de estaciones debe ser positivo", vbOKOnly
   NEst.Text = aviones(ave).NumeroEstaciones
End If
End Sub
```

FrmCopiar

INTERFAZ GRÁFICA



(1): LA. (ListBox)

(2): Lavion. (Label)

(3),(4): Aceptar, CommandButton1. (CommandButton)

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: se carga el formulario con la opción de copiar del avión de referencia y de los especiales. Se muestran los elementos del listbox ordenados.

LLAMADAS ENTRANTES:

FrmAvEspeciales.Copiar_Click

LLAMADAS SALIENTES:

GestAv.OrdenarAvionesEspeciales v

CÓDIGO:

```
Private Sub UserForm_Initialize()
Dim ave As Integer
Dim i As Integer
Dim v() As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Lavion.Caption = ave
LA.AddItem GestAv.AvRR.NumeroAvion
For i = 0 To GestAv.NumeroAvEspeciales - 1
If ave <> GestAv.AvEspeciales(i) Then
  LA.AddItem GestAv.AvEspeciales(i)
 End If
Next
ReDim v(LA.ListCount - 1)
For i = 0 To UBound(v)
   v(i) = LA.list(i, 0)
 Next
 GestAv.OrdenarAvionesEspeciales v
 For i = 0 To UBound(v)
  LA.list(i, 0) = v(i)
 Next
End Sub
```

Aceptar_Click

DESCRIPCIÓN: se copia el avión y se controlan sus enlaces y los enlaces del avión posterior.

LLAMADAS SALIENTES:

GestAv.CopiarAviones Origen, Destino

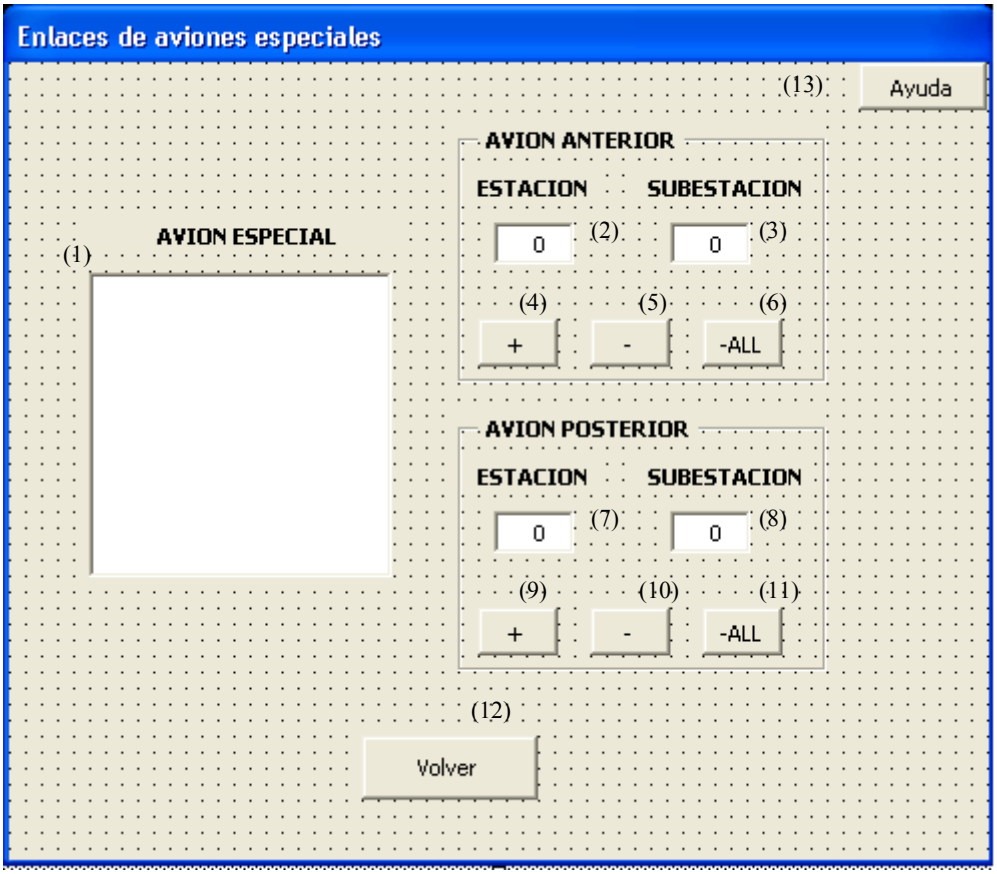
Control.EnlacesDefecto GestAv.aviones(Destino.NumeroAvion + 1);Control.EnlacesDefecto Destino

CÓDIGO:

```
Private Sub Aceptar_Click()
On Error GoTo fin:
Dim Origen As Avion
Dim Destino As Avion
If IsNull(LA.Value) Then
 MsgBox "Debe selecionar un valor haciendo click en el valor"
 Exit Sub
End If
If LA.Value = GestAv.AvRR.NumeroAvion Then
 Set Origen = AvRR
Else
 Set Origen = aviones(LA.Value)
End If
    Set Destino = aviones(FrmOpciones.AvionesEspeciales.Value)
GestAv.CopiarAviones Origen, Destino
If Destino.NumeroAvion + 1 <= planning.NumeroAviones And Origen.NumeroAvion + 1 <=
planning.NumeroAviones Then
  If GestAv.aviones(Destino.NumeroAvion + 1).Especial = False And Origen.NumeroAvion <>
AvRR.NumeroAvion And GestAv.aviones(Origen.NumeroAvion + 1).Especial = False Then
  GestAv.aviones(Destino.NumeroAvion + 1).EnlaceEstaciones = GestAv.aviones(Origen.NumeroAvion
+ 1).EnlaceEstaciones
  GestAv.aviones(Destino.NumeroAvion+1).EnlaceSubestaciones=GestAv.aviones(Origen.NumeroAvion
+ 1).EnlaceSubestaciones
  End If
End If
    If Origen.NumeroAvion = AvRR.NumeroAvion And Destino.NumeroAvion+1 <=
    planning.NumeroAniones  Then
  Control.EnlacesDefecto GestAv.aviones(Destino.NumeroAvion + 1)
End If
If Origen.NumeroAvion = AvRR.NumeroAvion Then
  Control.EnlacesDefecto Destino
End If
FrmAvEspecial.NEst.Text = Destino.NumeroEstaciones
Unload Me
Exit Sub
fin: MsgBox " No se pudo copiar!!!"
End Sub
```

frmEnlacesEspeciales

INTERFAZ GRÁFICA



(4),(5),(6),(9),(10),(11),(12),(13): OKAA, ELEA, AMenosALL, OKAP, ELEP, PMenosALL, Volver, CommandButton1. (CommandButon)

(2),(3),(7),(8): TBEA, TBSA, TBEP, TBSP. (TextBox)

(1): LBA. (ListBox)

MÉTODOS

UserForm_Initialize

DESCRIPCIÓN: se prepara y se muestran los enlaces del avión. Se contruye el ListBox con la información del avión. Si el avión no tiene enlaces se muestra un enlace como el de referencia.

LLAMADAS ENTRANTES:

FrmAvEspeciales.EnlacesEspeciales_Click

LLAMADAS SALIENTES:

GestAv.CopiarAviones GestAv.AvRR, GestAv.aviones(ave + 1)

CÓDIGO:

```
Private Sub UserForm_Initialize()
Dim ave As Integer
Dim av As Avion
Dim i As Integer
Dim j As Integer
Dim Str As String
ave = FrmOpciones.AvionesEspeciales.Value
frmEnlacesEspeciales.Caption = "ENLACES DEL AVION " & ave
If ave = 1 Then
 Anterior.Visible = False
End If
If ave = planning.NumeroAviones Then
 Posterior.Visible = False
End If
Set av = GestAv.aviones(ave)
 If IsEmpty(GestAv.aviones(ave + 1).EnlaceEstaciones) Then
  GestAv.CopiarAviones GestAv.AvRR, GestAv.aviones(ave + 1)
  End If
For i = 0 To av.NumeroEstaciones - 1
 If av.Estaciones(i) = 0 Then
  If av.Nombre(i, 0) = "" Then
   Str = "ST " & i + 1
    Else
    Str = av.Nombre(i, 0)
   End If
   LBA.AddItem Str
 Else
 For j = 1 To av.Estaciones(i)
   If av.Nombre(i, j) = "" Then
   Str = "ST " & i + 1 & " / " & j
    Else
    Str = av.Nombre(i, j)
   End If
   LBA.AddItem Str
  Next
   End If
Next
  LBA.Selected(0) = True
```

```
End Sub
```

Volver_Click

CÓDIGO:

```
Private Sub Volver_Click()
 Unload Me
End Sub
```

AMenosAll_Click

DESCRIPCIÓN: borramos todos los enlaces con el avión anterior.

CÓDIGO:

```
Private Sub AMenosAll_Click()
On Error GoTo fin:
Dim ave As Integer
Dim avAnt As Avion
Dim i As Integer
Dim j As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Set avAnt = GestAv.aviones(ave)
For i = 0 To avAnt.NumeroEstaciones - 1
 For j = 0 To avAnt.NumeroMaximoSubestaciones
 avAnt.setEnlaceEstaciones i, j, 0
 avAnt.setEnlaceSubestaciones i, j, 0
 Next
Next
Exit Sub
fin:
 MsgBox " Accion Incorrecta!!"
End Sub
```

CommandButton1_Click

DESCRIPCIÓN: se muestra información de como hay que realizar los enlaces.

CÓDIGO:

```
Private Sub CommandButton1_Click()
 FrmAyudaEnlaces.Show
End Sub
```

ELEA_Click

DESCRIPCIÓN: borramos el enlace que se presenta en las cajas de texto.

LLAMADAS SALIENTES:

Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)

CÓDIGO:

```
Private Sub ELEA_Click()
On Error GoTo fin
Dim ave As Integer
Dim Est As Integer
Dim Subest As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Est = Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Subest = Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)
GestAv.aviones(ave).setEnlaceEstaciones Est - 1, Subest, 0
GestAv.aviones(ave).setEnlaceSubestaciones Est - 1, Subest, 0
TBEA.Value = 0
TBSA.Value = 0
Exit Sub
fin:
 MsgBox " Enlace Incorrecto"
End Sub
```

ELEP_Click

DESCRIPCIÓN: borramos el enlace del avión posterior con el especial.

LLAMADAS SALIENTES:

Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.indiceEstacionSubestacion GestAv.aviones(ave + 1), Est, Subest

CÓDIGO:

```
Private Sub ELEP_Click()
On Error GoTo fin
Dim ave As Integer
Dim Est As Integer
Dim Subest As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Est = Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Subest = Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Control.indiceEstacionSubestacion GestAv.aviones(ave + 1), Est, Subest
If Est >= 0 Then
GestAv.aviones(ave + 1).setEnlaceEstaciones Est, Subest, 0
GestAv.aviones(ave + 1).setEnlaceSubestaciones Est, Subest, 0
Else
```

```
 GoTo fin
End If
TBEP.Value = 0
TBSP.Value = 0
Exit Sub
fin:
 MsgBox " Enlace incorrecto"
End Sub
```

LBA_Change

DESCRIPCIÓN: en función del elemento de ListBox seleccionado mostramos los enlaces con el avión anterior y posterior.

LLAMADAS SALIENTES:

Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.indiceEstacionSubestacion GestAv.aviones(ave + 1), Est, Subest

CÓDIGO:

```
Private Sub LBA_Change()
On Error GoTo fin:
Dim ave As Integer
Dim Est As Integer
Dim Subest As Integer
Dim Estreal As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Est = Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Subest = Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)
TBEA.Value = GestAv.aviones(ave).EnlaceEstaciones(Est - 1, Subest)
TBSA.Value = GestAv.aviones(ave).EnlaceSubestaciones(Est - 1, Subest)
If ave < planning.NumeroAviones Then
Control.indiceEstacionSubestacion GestAv.aviones(ave + 1), Est, Subest
TBEP.Value = Est + 1
TBSP.Value = Subest
End If
Exit Sub
fin: MsgBox "Error inesperado"
End Sub
```

OKAA_Click

DESCRIPCIÓN: se almacenan como enlaces los valores de los TextBox.

LLAMADAS SALIENTES:

Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.ControlEnlacesValidos(GestAv.aviones(ave), TBEA.Value, TBSA.Value)

CÓDIGO:

```
Private Sub OKAA_Click()
On Error GoTo fin
Dim ave As Integer
Dim Est As Integer
Dim Subest As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Est = Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Subest = Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)
If Control.ControlEnlacesValidos(GestAv.aviones(ave), TBEA.Value, TBSA.Value) = False Then
 GoTo fin:
End If
GestAv.aviones(ave).setEnlaceEstaciones Est - 1, Subest, TBEA.Value
GestAv.aviones(ave).setEnlaceSubestaciones Est - 1, Subest, TBSA.Value
Exit Sub
fin:
 MsgBox " Enlace Incorrecto"
End Sub
```

OKAP_Click

DESCRIPCIÓN: se almacenan como enlaces los valores de los TextBox.

LLAMADAS SALIENTES:

Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)

Control.ControlEnlacesValidos(GestAv.aviones(ave + 1), TBEP.Value, TBSP.Value, opc:=1)

CÓDIGO:

```
Private Sub OKAP_Click()
On Error GoTo fin
Dim ave As Integer
Dim Est As Integer
Dim Subest As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Est = Control.RecuperarEstacion(GestAv.aviones(ave), LBA.ListIndex + 1)
Subest = Control.RecuperarSubestacion(GestAv.aviones(ave), LBA.ListIndex + 1)
If Control.ControlEnlacesValidos(GestAv.aviones(ave + 1), TBEP.Value, TBSP.Value, opc:=1) = False
Then
 GoTo fin:
```

```
End If
GestAv.aviones(ave + 1).setEnlaceEstaciones TBEP.Value - 1, TBSP.Value, Est
GestAv.aviones(ave + 1).setEnlaceSubestaciones TBEP.Value - 1, TBSP.Value, Subest
Exit Sub
fin:
 MsgBox " Enlace incorrecto"
End Sub
```

PMenosAll_Click

DESCRIPCIÓN: se borran todos los enlaces del avión posterior (enlaces =0,0).

CÓDIGO:

```
Private Sub PMenosAll_Click()
On Error GoTo fin
Dim ave As Integer
Dim avPos As Avion
Dim i As Integer
Dim j As Integer
Dim NE As Integer
Dim Ns As Integer
ave = FrmOpciones.AvionesEspeciales.Value
Set avPos = GestAv.aviones(ave + 1)
 NE = avPos.NumeroEstaciones
 Ns = avPos.NumeroMaximoSubestaciones
For i = 0 To NE - 1
 For j = 0 To Ns
 avPos.setEnlaceEstaciones i, j, 0
 avPos.setEnlaceSubestaciones i, j, 0
 Next
Next
Exit Sub
fin:
 MsgBox " Accion Incorrecta!!"
End Sub
```

MÓDULOS

Inicio

Sub inicio()
FrmGeneral.Show
End Sub

Inicialización

VARIABLES

NOMBRE: FFecha
DESCRIPCIÓN: variable tipo DATE en la que se guarda la fecha de inicio o finalización del proyecto.
SINTAXIS: Public FFecha As Date

NOMBRE: AFecha
DESCRIPCIÓN: indica si la fecha Ffecha es de comienzo (=1) o de fin del proyecto(=0).
SINTAXIS: Public AFecha As Integer

MÉTODOS

RecuperarDatos

DESCRIPCIÓN: método principal de inicialización, en el que se "leen" todas las taras de Project buscando información y en función de lo que se quiera recuperar se llama a métodos más especificos. La recuperación principal de datos se realiza a partir del avión de comienzo de curva, para posteriormente calcular los datos que corresponden al de referencia. No tiene parámetros de entrada.
LLAMADAS ENTRANTES:
FrmGeneral.Userform_Initialize
LLAMADAS SALIENTES:
RecuperarDatosAvion ACCC;RecuperarDatosAvion aviones(IntAvion)
GestAv.CopiarAviones ACCC, AvRR
RecuperarDatosRef ACCC, AvRR; RecuperarDatosRef aviones(IntAvion), aviones(IntAvion)
RecuperarDatosRefHoras ACCC, AvRR;RecuperarDatosRefHoras aviones(IntAvion), aviones(IntAvion)
RecuperarEnlacesVerticales aviones(IntAvion), ObjAvionAnt

RecursoOperarios e; valoresRecursoOperarios True

valoresRecursoOperarios

CÓDIGO:

```
Sub RecuperarDatos()
  Dim i As Integer
  Dim ObjPlanificacion As Project
  Dim ObjAvion As Task
  Dim ObjAvionAnt As Task
  Dim IntNumAvion As Integer
  Dim IntAvion As Integer
  Dim e As Boolean
  planning.OpDefecto = 10
  planning.OpMax = 25
  planning.TurnoDefecto = 2
  planning.NumeroAviones = 1
  planning.AvFC = 1
  planning.AvCC = 1
  planning.NumeroRepAvCC = 1
  planning.RepRef = 1
  NumeroAvEspeciales = 0
  ReDim AvEspeciales(0)
  ReDim aviones(1 To 1)
  Set AvRR = New Avion
  Set ACCC = New Avion
  Set ObjPlanificacion = Application.ActiveProject
  FFecha = DateValue(ActiveProject.ProjectStart)
  AFecha = 1
  If ObjPlanificacion.NumberOfTasks = 0 Then
  AvRR.Inicializa
  AvRR.setCurvas 0, 0, 100
  AvRR.setPosiciones 0, 0, 1
  AvRR.setOperarios 0, 0, planning.OpDefecto
  AvRR.setCurvasHoras 0, 0, 100
  AvRR.setTurnos 0, 0, planning.TurnoDefecto
   Exit Sub
    End If
  IntNumAvion = ObjPlanificacion.OutlineChildren.Count
  IntAvion = 1
 ReDim aviones(1 To IntNumAvion)
  Set ObjAvionAnt = Nothing
```

```
 For Each ObjAvion In ObjPlanificacion.OutlineChildren
 If ObjAvion.Id = 1 Then
     FFecha = DateValue(ObjAvion.Start)
   End If
  If ObjAvion.Flag1 Then
      planning.AvCC = IntAvion
      planning.NumeroRepAvCC = ObjAvion.Number6
      planning.RepRef = ObjAvion.Number7
      planning.AvFC = ObjAvion.Number8
     Set ACCC.TaskAvion = ObjAvion
      AvRR.NumeroAvion = planning.AvCC + planning.RepRef - planning.NumeroRepAvCC
      ACCC.NumeroAvion = IntAvion
      RecuperarDatosAvion ACCC
      GestAv.CopiarAviones ACCC, AvRR
      RecuperarDatosRef ACCC, AvRR
      RecuperarDatosRefHoras ACCC, AvRR
   End If
   IntAvion = IntAvion + 1
Next
 IntAvion = 1
  For Each ObjAvion In ObjPlanificacion.OutlineChildren
  If ObjAvion.Flag2 Then  ' Av especial
   AvEspeciales(NumeroAvEspeciales) = IntAvion
    NumeroAvEspeciales = NumeroAvEspeciales + 1
    ReDim Preserve AvEspeciales(NumeroAvEspeciales)
    Set aviones(IntAvion) = New Avion
    Set aviones(IntAvion).TaskAvion = ObjAvion
    aviones(IntAvion).Especial = True
    aviones(IntAvion).EE = True
    aviones(IntAvion).NumeroAvion = IntAvion
    RecuperarDatosAvion aviones(IntAvion)
    If ObjAvion.Flag3 Then
       aviones(IntAvion).AplicaCurva = True
       RecuperarDatosRef aviones(IntAvion), aviones(IntAvion)
       RecuperarDatosRefHoras aviones(IntAvion), aviones(IntAvion)
     End If
    RecuperarEnlacesVerticales aviones(IntAvion), ObjAvionAnt
    End If
    If ObjAvion.Flag4 And ObjAvion.Flag2 = False Then ' EE
    Set aviones(IntAvion) = New Avion
```

```
      aviones(IntAvion).NumeroAvion = IntAvion
      aviones(IntAvion).EE = True
      Set aviones(IntAvion).TaskAvion = ObjAvion
      aviones(IntAvion).NumeroEstaciones = AvRR.NumeroEstaciones
      aviones(IntAvion).NumeroMaximoSubestaciones = AvRR.NumeroMaximoSubestaciones
      aviones(IntAvion).EnlaceEstaciones = AvRR.EnlaceEstaciones
      aviones(IntAvion).EnlaceSubestaciones = AvRR.EnlaceSubestaciones
      RecuperarEnlacesVerticales aviones(IntAvion), ObjAvionAnt
      End If
     IntAvion = IntAvion + 1
     Set ObjAvionAnt = ObjAvion
  Next
  planning.NumeroAviones = IntAvion - 1
  RecursoOperarios e
  If e = False Then
   valoresRecursoOperarios
  Else
   valoresRecursoOperarios True
  End If
End Sub
```

RecuperarDatosAvion

DESCRIPCIÓN: método que recupera las principales propiedades de un avión: duraciones, curvas, operarios, etcétera. El avión se pasa como parámetro de entrada.

LLAMADAS ENTRANTES:

RecuperarDatos

LLAMADAS SALIENTES:

Duracion.Redondeo(ObjEstacion.Number1, 2); Duracion.Redondeo(ObjSubEstacion.Number1, 2)

GestAv.ValoresPorDefecto Curva, 100, 2; GestAv.ValoresPorDefecto Posicion, 1, 2

Control.CalculoOperarios(HorasT(i, j), Dur(i, j))

CÓDIGO:

```
Sub RecuperarDatosAvion(av As Avion)
    Dim i As Integer
    Dim j As Integer
    Dim Objav As Task
    Dim ObjEstacion As Task
    Dim ObjAntEstacion As Task
    Dim ObjSubEstacion As Task
    Dim ObjAntSubEstacion As Task
    Dim Numest As Integer
```

```
    Dim MaxNumSubEst As Integer
    Dim Est As Integer
    Dim Subest As Integer
    Dim AntSubEst As Integer
    Dim NumSubEst As Integer
    Dim Cadena As String
    Dim Estacion() As Integer
    Dim Dur() As Single
    Dim HorasT() As Single
    Dim Curva() As Single
    Dim CurvaHoras() As Single
    Dim Posicion() As Integer
    Dim turnos() As Integer
    Dim Predecesora() As Integer
    Dim Descripcion() As String
    Dim Nombre() As String
    Dim e() As Integer
    Dim Op() As Single
    Dim d As Single
    Set Objav = av.TaskAvion
    Numest = Objav.OutlineChildren.Count
    ReDim Estacion(Numest - 1)
  MaxNumSubEst = 0
  Est = 0
    For Each ObjEstacion In Objav.OutlineChildren
     NumSubEst = ObjEstacion.OutlineChildren.Count
     Estacion(Est) = NumSubEst
      If NumSubEst >= MaxNumSubEst Then
          MaxNumSubEst = NumSubEst
          ReDim Preserve Dur(Numest - 1, NumSubEst)
          ReDim Preserve HorasT(Numest - 1, NumSubEst)
          ReDim Preserve Curva(Numest - 1, NumSubEst)
          ReDim Preserve CurvaHoras(Numest - 1, NumSubEst)
          ReDim Preserve Descripcion(Numest - 1, NumSubEst)
          ReDim Preserve Nombre(Numest - 1, NumSubEst)
          ReDim Preserve Posicion(Numest - 1, NumSubEst)
          ReDim Preserve turnos(Numest - 1, NumSubEst)
        End If
      Descripcion(Est, 0) = ObjEstacion.Text2
      Nombre(Est, 0) = ObjEstacion.Text1
```

```
      If NumSubEst = 0 Then
        Dur(Est, 0) = (ObjEstacion.Duration / 60) / 8
        HorasT(Est, 0) = Duracion.Redondeo(ObjEstacion.Number1, 2)
        Curva(Est, 0) = ObjEstacion.Number2
        CurvaHoras(Est, 0) = ObjEstacion.Number3
        turnos(Est, 0) = ObjEstacion.Number5
      End If
        Posicion(Est, 0) = ObjEstacion.Number4
        Subest = 1
        For Each ObjSubEstacion In ObjEstacion.OutlineChildren
          Dur(Est, Subest) = (ObjSubEstacion.Duration / 60) / 8
          HorasT(Est, Subest) = Duracion.Redondeo(ObjSubEstacion.Number1, 2)
          Curva(Est, Subest) = ObjSubEstacion.Number2
          CurvaHoras(Est, Subest) = ObjSubEstacion.Number3
          Posicion(Est, Subest) = ObjSubEstacion.Number4
          turnos(Est, Subest) = ObjSubEstacion.Number5
          Descripcion(Est, Subest) = ObjSubEstacion.Text2
          Nombre(Est, Subest) = ObjSubEstacion.Text1
          Subest = Subest + 1
        Next ObjSubEstacion
    Est = Est + 1
    Next ObjEstacion
  Set ObjSubEstacion = Nothing
  Set ObjAntSubEstacion = Nothing
  If MaxNumSubEst = 0 Then
   ReDim Predecesora(Numest - 1, 0, 0)
  Else
  ReDim Predecesora(Numest - 1, MaxNumSubEst, MaxNumSubEst - 1)
  End If
  'CC Depende si queremos empezar subestaciones en paralelo al mismo tiempo
  Est = 0
  For Each ObjEstacion In Objav.OutlineChildren
     Subest = 0
    For Each ObjSubEstacion In ObjEstacion.OutlineChildren
      AntSubEst = 0
      For Each ObjAntSubEstacion In ObjEstacion.OutlineChildren
       Cadena = CStr(ObjAntSubEstacion.Id) & "CC"
       If ObjAntSubEstacion = ObjSubEstacion Then
          Exit For
```

```
      ElseIf (InStr(ObjSubEstacion.Predecessors, CStr(ObjAntSubEstacion.Id)) > 0 And
InStr(ObjSubEstacion.Predecessors, Cadena) = 0) Then  Predecesora(Est, Subest + 1, AntSubEst + 1) = 1
      ElseIf (InStr(ObjSubEstacion.Predecessors, "CC") > 0 Or ObjSubEstacion.Predecessors = "") And
AntSubEst = 0 Then  Predecesora(Est, Subest + 1, AntSubEst) = 1
       End If
      AntSubEst = AntSubEst + 1
      Next ObjAntSubEstacion
      Set ObjAntSubEstacion = ObjSubEstacion
      Subest = Subest + 1
    Next ObjSubEstacion
    Set ObjAntEstacion = ObjEstacion
    Est = Est + 1
  Next ObjEstacion
  GestAv.ValoresPorDefecto Curva, 100, 2
  GestAv.ValoresPorDefecto Posicion, 1, 2
  ReDim e(Numest - 1, MaxNumSubEst)
  ReDim Op(Numest - 1, MaxNumSubEst)
  av.NumeroEstaciones = Numest
  av.NumeroMaximoSubestaciones = MaxNumSubEst
  av.Estaciones = Estacion
  av.Duraciones = Dur
  av.Curvas = Curva
  av.Posiciones = Posicion
  av.Predecesoras = Predecesora
  av.Descripcion = Descripcion
  av.Nombre = Nombre
  av.EnlaceEstaciones = e
  av.EnlaceSubestaciones = e
  av.turnos = turnos
  av.HorasTrabajo = HorasT
  av.CurvasHoras = CurvaHoras
  For i = 0 To Numest - 1
   For j = 0 To MaxNumSubEst
    Op(i, j) = Control.CalculoOperarios(HorasT(i, j), Dur(i, j))
   Next
  Next
  av.Operarios = Op
End Sub
```

RecuperarDatosRef

DESCRIPCIÓN: calculamos los datos (duraciones en días) del avión de referencia a partir de las duraciones del avión de comienzo de curva. Este método es genérico con lo que el "calculo inverso" en la curva de aprendizaje se puede hacer para cualquier avión. Los parámetros de entrada son av1 y av2 (aviones), av1 es el avión de dato conocido y av2 el que se calcula a partir de av1.

LLAMADAS ENTRANTES:

RecuperarDatos

LLAMADAS SALIENTES:

Duracion.Redondeo((*Duracion.CalculoInverso*(TT, av1.Duraciones(Est, 0) * 8 * 60, c1) / 60) / 8, 2)

Duracion.Redondeo((*Duracion.CalculoInverso*(TT, av1.Duraciones(Est, Subest) * 8 * 60, c2) / 60) / 8, 2)

CÓDIGO:

```
Sub RecuperarDatosRef(av1 As Avion, av2 As Avion)
Dim Est As Integer
Dim Subest As Integer
Dim c1 As Integer
Dim c2 As Integer
Dim TT As Integer
 If av1.NumeroAvion <= planning.AvCC Then
  TT = planning.AvCC
 Else
  TT = av1.NumeroAvion
 End If
For Est = 0 To av2.NumeroEstaciones - 1
 c1 = av1.Curvas(Est, 0)
 If c1 = 0 Then
  c1 = 100
 End If
 av2.setDuraciones Est, 0, Duracion.Redondeo((Duracion.CalculoInverso(TT, av1.Duraciones(Est, 0) * 8
* 60, c1) / 60) / 8, 2)
 For Subest = 1 To av2.NumeroMaximoSubestaciones
 c2 = av1.Curvas(Est, Subest)
 If c2 = 0 Then
 c2 = 100
 End If
 av2.setDuraciones Est, Subest, Duracion.Redondeo((Duracion.CalculoInverso(TT, av1.Duraciones(Est,
Subest) * 8 * 60, c2) / 60) / 8, 2)
 Next
Next
End Sub
```

RecuperarDatosRefHoras

DESCRIPCIÓN: método parecido al anterior, cuyo objetivo es recuperar las horas de trabajo a partir de otro avión "entrando" de forma inversa en la curva de aprendizaje.

LLAMADAS ENTRANTES:

RecuperarDatos

LLAMADAS SALIENTES:

Duracion.Redondeo((Duracion.CalculoInverso(TT,CDbl(av1.HorasTrabajo(Est, 0)), c1)), 2)

Duracion.Redondeo((Duracion.CalculoInverso(TT,CDbl(av1.HorasTrabajo(Est, Subest)), c2)), 2)

CÓDIGO:

```
Sub RecuperarDatosRefHoras(av1 As Avion, av2 As Avion)
Dim Est As Integer
Dim Subest As Integer
Dim c1 As Integer
Dim c2 As Integer
Dim TT As Integer
 If av1.NumeroAvion <= planning.AvCC Then
  TT = planning.AvCC
 Else
  TT = av1.NumeroAvion
 End If
For Est = 0 To av2.NumeroEstaciones - 1
 c1 = av1.CurvasHoras(Est, 0)
 If c1 = 0 Then
  c1 = 100
 End If
 av2.setHorasTrabajo Est, 0,
Duracion.Redondeo((Duracion.CalculoInverso(TT,CDbl(av1.HorasTrabajo(Est, 0)), c1)), 2)
 For Subest = 1 To av2.NumeroMaximoSubestaciones
 c2 = av1.CurvasHoras(Est, Subest)
 If c2 = 0 Then
 c2 = 100
 End If
 av2.setHorasTrabajo Est, Subest,
Duracion.Redondeo((Duracion.CalculoInverso(TT,CDbl(av1.HorasTrabajo(Est, Subest)), c2)), 2)
 Next
Next
End Sub
```

RecuperarEnlacesVerticales

DESCRIPCIÓN: método principal de la recuperación de la vinculación ente distintos aviones, que llama a los métodos encargados de extraer la información. Parámetros de entrada av (avión que queremos saber su vinculación), TaskavAnt (objeto tarea que representa el avión anterior vinculado).

LLAMADAS ENTRANTES:

RecuperarDatos

LLAMADAS SALIENTES:

obtencionVinculo(ObjEstacion.Predecessors)

Control.RecuperarEstacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)

Control.RecuperarSubestacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)

CÓDIGO:

```
Sub RecuperarEnlacesVerticales(av As Avion, TaskavAnt As Task)
Dim avAnt As Avion
Dim ObjEstacion As Task
Dim ObjSubEstacion As Task
Dim vinc As Variant
Dim i As Integer
Dim EstEn As Integer
Dim SubestEn As Integer
Dim Est As Integer
Dim Subest As Integer
 If TaskavAnt Is Nothing Then
  Exit Sub
 End If
 If TaskavAnt.Flag2 = False Then
  Set avAnt = AvRR
  Else
   Set avAnt = aviones(av.NumeroAvion - 1)
 End If
  Est = 0
  For Each ObjEstacion In av.TaskAvion.OutlineChildren
   vinc = obtencionVinculo(ObjEstacion.Predecessors)
   For i = 0 To UBound(vinc) - 1
    If vinc(i) < av.TaskAvion.Id Then
      EstEn = Control.RecuperarEstacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)
      SubestEn = Control.RecuperarSubestacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)
      av.setEnlaceEstaciones Est, 0, EstEn
      av.setEnlaceSubestaciones Est, 0, SubestEn
     End If
   Next
```

```
  Subest = 1
 For Each ObjSubEstacion In ObjEstacion.OutlineChildren
  vinc = obtencionVinculo(ObjSubEstacion.Predecessors)
 For i = 0 To UBound(vinc) - 1
  If vinc(i) < av.TaskAvion.Id Then
    EstEn = Control.RecuperarEstacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)
    SubestEn = Control.RecuperarSubestacion(avAnt, vinc(i) - TaskavAnt.Id, opc:=1)
    av.setEnlaceEstaciones Est, Subest, EstEn
    av.setEnlaceSubestaciones Est, Subest, SubestEn
   End If
 Next
 Subest = Subest + 1
 Next
 Est = Est + 1
Next
End Sub
```

obtencionVinculo

DESCRIPCIÓN: función que a partir de la información de vinculación de una tarea que nos da Project (String), distingue cúal es una vinculación vertical, y devuelve un vector de enteros que representa la tarea del avión predecesor.

LLAMADAS ENTRANTES:

RecuperarEnlacesVerticales

CÓDIGO:

```
Function obtencionVinculo(predec As String) As Variant
Dim posdiv As Long
Dim Nenlaces As Integer
Dim n() As Integer
Dim s() As String
Dim i As Integer
Dim j As Integer
 posdiv = 1
 Nenlaces = 0
 j = 0
 ReDim n(0)
 ReDim s(0)
 If predec <> "" Then
 posdiv = InStr(predec, ";")
 While posdiv <> 0
 Nenlaces = Nenlaces + 1
```

```
 ReDim Preserve s(Nenlaces)
 s(Nenlaces - 1) = Left(predec, posdiv - 1)
 predec = Mid(predec, posdiv + 1)
 posdiv = InStr(predec, ";")
 Wend
 s(Nenlaces) = predec
 s(Nenlaces) = predec
 For i = 0 To Nenlaces
  If InStr(s(i), "C") = 0 Then
   j = j + 1
   ReDim Preserve n(j)
   n(j - 1) = CInt(s(i))
  End If
 Next
 End If
 obtencionVinculo = n
End Function
```

RecursoOperarios

DESCRIPCIÓN: función que sirve tanto para añadir en Project el recurso operarios, como para saber si ya existen operarios y contarlos en ese caso.

LLAMADAS ENTRANTES:

RecuperarDatos

CÓDIGO:

```
Function RecursoOperarios(Optional existe As Boolean, Optional Noañadir As Boolean) As Integer
Dim RecursosP As Resources
Dim SoloRecurso As Resource
Dim n As String
Dim i As Integer
existe = False
Set RecursosP = ActiveProject.Resources
For i = 1 To RecursosP.Count
 Set SoloRecurso = RecursosP.Item(i)
 n = SoloRecurso.GetField(pjResourceName)
 If n = "Operarios" Then
   existe = True
   Exit For
 End If
Next
If existe = False And Noañadir = False Then
```

```
 RecursosP.Add "Operarios"
 i = RecursosP.Count
End If
RecursoOperarios = i
End Function
```

valoresRecursoOperarios

DESCRIPCIÓN: procedimiento que sirve para leer y escribir desde y en las variables del programa y Project los campos relacionados con los operarios.

LLAMADAS ENTRANTES:

RecuperarDatos

Interno.Ejecutar

CÓDIGO:

```
Sub valoresRecursoOperarios(Optional leer As Boolean)
Dim i As Integer
Dim e As Boolean
Dim RR As Resources
Dim r As Resource
Set RR = ActiveProject.Resources
i = RecursoOperarios(e)
Set r = RR.Item(i)
If leer = False Then
r.SetField pjResourceInitials, "Op"
r.SetField pjResourceMaxUnits, planning.OpMax
r.SetField pjResourceNumber1, planning.OpDefecto
r.SetField pjResourceNumber2, planning.TurnoDefecto
Else
 planning.OpMax = r.GetField(pjResourceMaxUnits)
 planning.OpDefecto = r.GetField(pjResourceNumber1)
 planning.TurnoDefecto = r.GetField(pjResourceNumber2)
 If planning.OpDefecto = 0 Then
  planning.OpDefecto = 10
 End If
End If
End Sub
```

GestFrmEstaciones

VARIABLES

NOMBRE: pgs

DESCRIPCIÓN: array de páginas correspondientes a las del formulario estaciones.

SINTAXIS: Public pgs() As Page

NOMBRE: av

DESCRIPCIÓN: variable que representa al avión que en ese momento ha accedido al formulario de estaciones. Puede ser un avión especial como el de referencia.

SINTAXIS: Public av As New Avion

MÉTODOS

añadirEstación

DESCRIPCIÓN: se añade un nueva estación en la posición correcta del formulario FrmEstaciones, copiando los controles del formulario FEstaciones.

LLAMADAS ENTRANTES:

FrmEstaciones.UserForm_Initialize

FrmEstaciones.AñadirEstacion_Click

CÓDIGO:

```
Sub añadirEstación()
  Dim n As Long
  Dim estacionx As Long, i As Integer
  Dim NumEstaciones As Integer
  Dim Nombre As String
  Dim pg As Page
  Dim stations As Pages
  Dim controles As Controls
  Set stations = FrmEstaciones.Multiestacion.Pages
  estacionx = FrmEstaciones.Multiestacion.Value
  n = estacionx + 2
  Nombre = "Estación" & n
  Set pg = stations.Add(Nombre, Nombre, n - 1)
  NumEstaciones = FrmEstaciones.Multiestacion.Count
  ReDim Preserve pgs(NumEstaciones - 1)
    For i = 0 To (NumEstaciones - 1)
```

```
    Set pgs(i) = stations.Item(i)
    Next
    For i = n To (NumEstaciones - 1)
    pgs(i).Caption = "Estación" & i + 1
   Next
 ' Copiar controles
  Set controles = FEstaciones.Controls
  controles.SelectAll
  controles.Copy
  pg.Paste
  pg.Repaint
 End Sub
```

EliminarEstacion

DESCRIPCIÓN: Se borra la estación correspondiente del formulario de estaciones y se reordena este.

LLAMADAS ENTRANTES:

FrmEstaciones.EliminarEstacion_Click

CÓDIGO:

```
Sub EliminarEstacion()
  Dim NumEstaciones As Integer, i As Integer
  Dim estacionx As Long
  Dim stations As Pages
  Set stations = FrmEstaciones.Multiestacion.Pages
  estacionx = FrmEstaciones.Multiestacion.Value
  stations.Remove (estacionx)
  NumEstaciones = FrmEstaciones.Multiestacion.Count
  ReDim Preserve pgs(NumEstaciones - 1)
  For i = 0 To (NumEstaciones - 1)
    Set pgs(i) = stations.Item(i)
  Next
  For i = estacionx To (NumEstaciones - 1)
    pgs(i).Caption = "Estación" & i + 1
  Next
End Sub
```

obtenerDatosEstacion

DESCRIPCIÓN: se muestra el forumario de estaciones con la información correcta de cada avión. Los valores de operarios los gestiona otro método.

LLAMADAS ENTRANTES:

FrmEstaciones.UserForm_Initialize

FrmEstaciones.AñadirEstacion_Click

LLAMADAS SALIENTES:

ObtenerFrmOperarios Op, av, n

CÓDIGO:

```
Sub obtenerDatosEstacion(n As Integer, i As Integer)
 ' n : Número de Estacion
 ' i: Página del Multipage en la que se van a modificar los campos
 Dim boxNsub As TextBox
 Dim boxDur As TextBox
 Dim boxCurva As TextBox
 Dim boxPosiciones As TextBox
 Dim boxDesc As TextBox
 Dim boxNombre As TextBox
 Dim LD As Label
 Dim LC As Label
 Dim Nsub As Integer
 Dim DurEst As Single
 Dim ValCurva As Single
 Dim av As Avion
 Dim Op As Frame
 Set av = FrmEstaciones.AvCopia
 Nsub = av.Estaciones(n - 1)
  ' Actualizamos maximo número de subestaciones
   If (Nsub > av.NumeroMaximoSubestaciones) Then
   av.NumeroMaximoSubestaciones = Nsub
   End If
  Set boxNsub = pgs(i).Controls.Item(5)
  Set boxDur = pgs(i).Controls.Item(3)
  Set boxCurva = pgs(i).Controls.Item(7)
  Set boxDesc = pgs(i).Controls.Item(1)
  Set boxPosiciones = pgs(i).Controls.Item(9)
  Set LD = pgs(i).Controls.Item(2)
  Set LC = pgs(i).Controls.Item(6)
  Set Op = pgs(i).Controls.Item(12)
   If i = 0 Then
    Set boxNombre = pgs(i).Controls.Item(11)
   Else
    Set boxNombre = pgs(i).Controls.Item(10)
  End If
  boxNsub.Text = Nsub
```

```
  boxPosiciones.Text = av.Posiciones(n - 1, 0)
  boxDesc.Text = av.Descripcion(n - 1, 0)
  boxNombre.Text = av.Nombre(n - 1, 0)
  DurEst = av.Duraciones(n - 1, 0)
  ValCurva = av.Curvas(n - 1, 0)
If Nsub = 0 Then
    LD.Visible = True
    boxDur.Visible = True
    boxDur.Text = DurEst
    LC.Visible = True
    boxCurva.Visible = True
    boxCurva.Text = ValCurva
    Op.Visible = True
 Else
    LD.Visible = False
    boxDur.Visible = False
    LC.Visible = False
    boxCurva.Visible = False
   Op.Visible = False
  End If
   ObtenerFrmOperarios Op, av, n
End Sub
```

insertarDatosEstacion

DESCRIPCIÓN: actualizamos las variables del avión a partir de los datos del formulario.Los operarios se gestionan con otro método.

LLAMADAS ENTRANTES:

FrmEstaciones.SubEstaciones_Click

FrmEstaciones.ActualizarEstacion_Click

LLAMADAS SALIENTES:

av.Actualiza2D av.NumeroEstaciones - 1, Nsub

av.Actualiza3D av.NumeroEstaciones - 1, Nsub

InsertarFrmOperarios Op, av, n

CÓDIGO:

```
Sub insertarDatosEstacion(n As Integer)
' Actualizamos las variables del avion con los datos dados por el usuario
' n : Estación a modificar
 On Error GoTo fin:
 Dim boxNsub As TextBox
 Dim boxDur As TextBox
```

```
Dim boxCurva As TextBox
Dim boxPosiciones As TextBox
Dim boxDesc As TextBox
Dim boxNombre As TextBox
Dim Nsub As Integer
Dim Dur As Single
Dim ValCurva As Single
Dim LD As Label
Dim LC As Label
Dim av As Avion
Dim Op As Frame
 Set av = FrmEstaciones.AvCopia
 Set boxNsub = pgs(n - 1).Controls.Item(5)
 Set boxDur = pgs(n - 1).Controls.Item(3)
 Set boxCurva = pgs(n - 1).Controls.Item(7)
 Set boxPosiciones = pgs(n - 1).Controls.Item(9)
 Set boxDesc = pgs(n - 1).Controls.Item(1)
 Set LD = pgs(n - 1).Controls.Item(2)
 Set LC = pgs(n - 1).Controls.Item(6)
 Set Op = pgs(n - 1).Controls.Item(12)
  If pgs(n - 1).Name = "Page1" Then
 Set boxNombre = pgs(n - 1).Controls.Item(11)
 Else
 Set boxNombre = pgs(n - 1).Controls.Item(10)
 End If
 Nsub = boxNsub.Text
 av.setPosiciones n - 1, 0, boxPosiciones.Text
 av.setDescripcion n - 1, 0, boxDesc.Text
 av.setNombre n - 1, 0, boxNombre.Text
 av.setEstaciones n - 1, Nsub
 ' Datos de Duración
 If (Nsub >= av.NumeroMaximoSubestaciones) Then
  av.NumeroMaximoSubestaciones = Nsub
  av.Actualiza2D av.NumeroEstaciones - 1, Nsub
  av.Actualiza3D av.NumeroEstaciones - 1, Nsub
 End If
 If (Nsub = 0 And boxDur.Visible) Then
 Dur = boxDur.Text
 ValCurva = boxCurva.Text
 av.setDuraciones n - 1, 0, Dur
```

```
 av.setCurvas n - 1, 0, ValCurva
 End If

 If (Nsub = 0 And Not boxDur.Visible) Then
 LD.Visible = True
 boxDur.Visible = True
 LC.Visible = True
 boxCurva.Visible = True
 Op.Visible = True
 End If
 If Nsub <> 0 Then
   LD.Visible = False
   boxDur.Visible = False
   LC.Visible = False
   boxCurva.Visible = False
   Op.Visible = False
   End If
 InsertarFrmOperarios Op, av, n
Exit Sub
fin: MsgBox " Dato/s Incorrecto/s"
End Sub
```

InsertarFrmOperarios

DESCRIPCIÓN: se actualizan las variables del avión con la información suministrada por el formulario.

LLAMADAS ENTRANTES:

GestFrmEstaciones.InsertarDatosEstacion

CÓDIGO:

```
Sub InsertarFrmOperarios(Op As Frame, av As Avion, n As Integer)
Dim TBHoras As TextBox
Dim TBCH As TextBox
Dim OB1 As OptionButton
Dim OB2 As OptionButton
Dim OB3 As OptionButton
Dim TBOp As TextBox
Dim TurnoEst As Integer
Dim OpTotal As Single
Set TBHoras = Op.Controls.Item(1)
Set TBCH = Op.Controls.Item(3)
Set OB1 = Op.Controls.Item(6)
Set OB2 = Op.Controls.Item(7)
```

```
Set OB3 = Op.Controls.Item(8)
Set TBOp = Op.Controls.Item(5)
av.setHorasTrabajo n - 1, 0, TBHoras.Text
av.setCurvasHoras n - 1, 0, TBCH.Text
TurnoEst = 1
If OB1.Value = True Then
 TurnoEst = 1
End If
If OB2.Value = True Then
 TurnoEst = 2
End If
If OB3.Value = True Then
 TurnoEst = 3
End If
av.setTurnos n - 1, 0, TurnoEst
OpTotal = TBOp.Text * TurnoEst
av.setOperarios n - 1, 0, OpTotal
End Sub
```

ObtenerFrmOperarios

DESCRIPCIÓN: se muestra la información en el formualario a partir de las variables del avión.

LLAMADAS ENTRANTES:

GestFrmEstaciones.ObtenerDatosEstacion

GestFrmEstaciones.ActualizarFrmOperariosEstaciones

CÓDIGO:

```
Sub ObtenerFrmOperarios(Op As Frame, av As Avion, n As Integer)
Dim TBHoras As TextBox
Dim TBCH As TextBox
Dim OB1 As OptionButton
Dim OB2 As OptionButton
Dim OB3 As OptionButton
Dim TBOp As TextBox
Dim TurnoEst As Integer
Set TBHoras = Op.Controls.Item(1)
Set TBCH = Op.Controls.Item(3)
Set OB1 = Op.Controls.Item(6)
Set OB2 = Op.Controls.Item(7)
Set OB3 = Op.Controls.Item(8)
Set TBOp = Op.Controls.Item(5)
TBHoras.Text = av.HorasTrabajo(n - 1, 0)
```

```
TBCH.Text = av.CurvasHoras(n - 1, 0)
If av.turnos(n - 1, 0) = 1 Then
 OB1.Value = True
End If
If av.turnos(n - 1, 0) = 2 Or av.turnos(n - 1, 0) = 0 Then
 OB2.Value = True
End If
If av.turnos(n - 1, 0) = 3 Then
 OB3.Value = True
End If
If av.turnos(n - 1, 0) <> 0 Then
TBOp.Text = av.Operarios(n - 1, 0) / av.turnos(n - 1, 0)
End If
End Sub
```

ActualizarFrmOperariosEstaciones

DESCRIPCIÓN: Las variables del avión referentes a operarios se actualizan para todas las estaciones

LLAMADAS ENTRANTES:

FrmEstaciones.ActualizarEstacion_Click

LLAMADAS SALIENTES:

GestAv.CalcularHorasTrabajo av

GestAv.CalcularOperarios av

GestFrmEstaciones.ObtenerFrmOperarios Op, av, n

CÓDIGO:

```
Sub ActualizarFrmOperariosEstaciones(av As Avion)
 Dim st As Pages
 Dim numpag As Long
 Dim n As Integer
 Dim Op As Frame
 GestAv.CalcularHorasTrabajo av
 GestAv.CalcularOperarios av
  Set st = FrmEstaciones.Multiestacion.Pages
 numpag = st.Count
 For n = 1 To numpag
   Set Op = st(n - 1).Controls.Item(12)
   GestFrmEstaciones.ObtenerFrmOperarios Op, av, n
 Next
End Sub
```

GestFrmSubestaciones

VARIABLES

NOMBRE: pgs

DESCRIPCIÓN: variable en la que se almacenan las subestaciones (páginas) de una estación.

SINTAXIS: Public pgs() As Page

NOMBRE: modo

DESCRIPCIÓN: distingue ente modo estaciones(=0) y subestaciones(=1). Sirve para indicar a otros método en que formulario estamos.

SINTAXIS: Public modo As Integer

MÉTODOS

añadirSubEstación

DESCRIPCIÓN: añadimos subestaciones al formulario de subestaciones, copiando los controles de FsubEstaciones.

LLAMADAS ENTRANTES:

FrmSubestaciones.UserForm_initalize

CÓDIGO:

```
Sub añadirSubEstación()
  Dim n As Long
  Dim subestacionx As Long, i As Integer, numsubestaciones As Integer
  Dim Nombre As String
  Dim pg As Page
  Dim substations As Pages
  Dim controles As Controls
  Set substations = FrmSubEstaciones.MultiSubestacion.Pages
  subestacionx = FrmSubEstaciones.MultiSubestacion.Value
  n = subestacionx + 2
  Nombre = "subestación" & n
  Set pg = substations.Add(Nombre, Nombre, n - 1)
  numsubestaciones = FrmSubEstaciones.MultiSubestacion.Count
  ReDim pgs(numsubestaciones - 1)
    For i = 0 To (numsubestaciones - 1)
     Set pgs(i) = substations.Item(i)
     Next
```

```
   For i = n To (numsubestaciones - 1)
    pgs(i).Caption = "subestación" & i + 1
   Next
 ' Copiar controles
  Set controles = FSubEstaciones.Controls
  controles.SelectAll
  controles.Copy
  pg.Paste
  pg.Repaint
  End Sub
```

eliminarsubEstacion

DESCRIPCIÓN: se elimiman páginas del formulario de subestaciones.

LLAMADAS ENTRANTES:

LLAMADAS SALIENTES:

CÓDIGO:

```
Sub eliminarsubEstacion()
  Dim numsubestaciones As Integer, i As Integer
  Dim subestacionx As Long
  Dim substations As Pages
  Set substations = FrmSubEstaciones.MultiSubestacion.Pages
  subestacionx = FrmSubEstaciones.MultiSubestacion.Value
  substations.Remove (subestacionx)
  numsubestaciones = FrmSubEstaciones.MultiSubestacion.Count
  ReDim pgs(numsubestaciones - 1)
  For i = 0 To (numsubestaciones - 1)
     Set pgs(i) = substations.Item(i)
  Next
  For i = subestacionx To (numsubestaciones - 1)
     pgs(i).Caption = "subestación" & i + 1
  Next
End Sub
```

obtenerDatosSubestacion

DESCRIPCIÓN: se muestra la información del formulario al usuario en función de las variables internas del avión. Como parámetros de entrada se da la página (subestación), y su posición dentro del formlurario(multipage).

LLAMADAS ENTRANTES:

FrmSubestaciones.UserForm_initalize

LLAMADAS SALIENTES:

ObtenerFrmSubOperarios Op, av, pagEst, n

CÓDIGO:

```
Sub obtenerDatosSubestacion(pagEst As Integer, n As Integer, i As Integer)
 Dim boxDur As TextBox
 Dim boxCurva As TextBox
 Dim boxPosiciones As TextBox
 Dim boxDesc As TextBox
 Dim boxNombre As TextBox
 Dim xsub As Single
 Dim ValCurva As Single
 Dim ValDesc As String
 Dim Op As Frame
 Dim av As Avion
 Set av = FrmEstaciones.AvCopia
 xsub = av.Duraciones(pagEst, n)
 ValCurva = av.Curvas(pagEst, n)
 ValDesc = av.Descripcion(pagEst, n)
 Set boxDur = pgs(i).Controls.Item(3)
 Set boxDesc = pgs(i).Controls.Item(1)
 boxDur.Text = xsub
 boxDesc.Text = ValDesc
 Dim lista As ListBox
 Dim cbox As CheckBox
 Dim xpos As Integer
 Dim k As Integer
 If i <> 0 Then
 Set lista = pgs(i).Controls.Item(5)
 Set cbox = pgs(i).Controls.Item(6)
 Set boxCurva = pgs(i).Controls.Item(8)
 Set boxPosiciones = pgs(i).Controls.Item(10)
 Set boxNombre = pgs(i).Controls.Item(12)
 Set Op = pgs(i).Controls.Item(13)
 boxPosiciones.Text = av.Posiciones(pagEst, n)
 boxCurva.Text = ValCurva
 boxNombre.Text = av.Nombre(pagEst, n)
 xpos = av.Predecesoras(pagEst, n, 0)
If xpos = 1 Then
  cbox.Value = True
Else
  cbox.Value = False
```

```
For k = 1 To n - 1
  xpos = av.Predecesoras(pagEst, n, k)
  If xpos = 1 Then
   Dim b As Boolean
   b = lista.MultiSelect
   lista.Selected(k - 1) = True
  End If
Next k
End If
Else
  Set boxCurva = pgs(i).Controls.Item(5)
  boxCurva.Text = ValCurva
  Set boxPosiciones = pgs(i).Controls.Item(7)
  boxPosiciones.Text = av.Posiciones(pagEst, n)
  Set boxNombre = pgs(i).Controls.Item(9)
  boxNombre.Text = av.Nombre(pagEst, n)
  Set Op = pgs(i).Controls.Item(10)
End If
 ObtenerFrmSubOperarios Op, av, pagEst, n
End Sub
```

insertarDatosSubestacion

DESCRIPCIÓN: Se actualizan las variables del avión con los datos aportados por el usuario.

LLAMADAS ENTRANTES:

FrmSubestaciones.ActualizarSubestación_Click

LLAMADAS SALIENTES:

InsertarFrmSubOperarios Op, av, pagEst, n

CÓDIGO:

```
Sub insertarDatosSubestacion(pagEst As Integer, n As Integer)
 On Error GoTo fin:
 Dim boxDur As TextBox
 Dim boxCurva As TextBox
 Dim boxPosiciones As TextBox
 Dim boxDesc As TextBox
 Dim boxNombre As TextBox
 Dim xsub As Single
 Dim ValCurva As Single
 Dim i As Integer
 Dim Op As Frame
 Dim av As Avion
```

```
  Set av = FrmEstaciones.AvCopia
 i = n - 1
 Set boxDur = pgs(i).Controls.Item(3)
 Set boxDesc = pgs(i).Controls.Item(1)
 xsub = boxDur.Text
 av.setDuraciones pagEst, n, xsub
 av.setDescripcion pagEst, n, boxDesc.Text
 Dim lista As ListBox
 Dim cbox As CheckBox
 Dim xpos As Integer
 Dim k As Integer
 Dim sel As Boolean
  If i <> 0 Then
Set lista = pgs(i).Controls.Item(5)
 Set cbox = pgs(i).Controls.Item(6)
 Set boxCurva = pgs(i).Controls.Item(8)
 Set boxPosiciones = pgs(i).Controls.Item(10)
 Set boxNombre = pgs(i).Controls.Item(12)
 Set Op = pgs(i).Controls.Item(13)
 ValCurva = boxCurva.Text
 av.setCurvas pagEst, n, ValCurva
 av.setPosiciones pagEst, n, boxPosiciones.Text
 av.setNombre pagEst, n, boxNombre.Text
If cbox.Value Then
   av.setPredecesoras pagEst, n, 0, 1
  For k = 1 To n - 1
     av.setPredecesoras pagEst, n, k, 0
  Next k
Else
For k = 1 To n - 1
  sel = lista.Selected(k - 1)
  If sel Then
   av.setPredecesoras pagEst, n, k, 1
   av.setPredecesoras pagEst, n, 0, 0
   Else
   av.setPredecesoras pagEst, n, k, 0
  End If
Next k
End If
Else
```

```
 Set boxCurva = pgs(i).Controls.Item(5)
 Set boxPosiciones = pgs(i).Controls.Item(7)
 Set boxNombre = pgs(i).Controls.Item(9)
 Set Op = pgs(i).Controls.Item(10)
 ValCurva = boxCurva.Text
 av.setCurvas pagEst, n, ValCurva
 av.setPosiciones pagEst, n, boxPosiciones.Text
 av.setPredecesoras pagEst, n, 0, 1
 av.setNombre pagEst, n, boxNombre.Text
End If
 InsertarFrmSubOperarios Op, av, pagEst, n
Exit Sub
fin: MsgBox " Dato/s Incorrecto/s"
End Sub
```

InsertarFrmSubOperarios

DESCRIPCIÓN: con los datos de operario del formulario (Op), se actualizan las variables del avión (Av) de la estación (Est) y subestación (Subest).

LLAMADAS ENTRANTES:

GestFrmSubestaciones.insertarDatosSubestacion

CÓDIGO:

```
Sub InsertarFrmSubOperarios(Op As Frame, av As Avion, Est As Integer, Subest As Integer)
Dim TBHoras As TextBox
Dim TBCH As TextBox
Dim OB1 As OptionButton
Dim OB2 As OptionButton
Dim OB3 As OptionButton
Dim TBOp As TextBox
Dim TurnoEst As Integer
Dim OpTotal As Single
Set TBHoras = Op.Controls.Item(1)
Set TBCH = Op.Controls.Item(3)
Set OB1 = Op.Controls.Item(6)
Set OB2 = Op.Controls.Item(7)
Set OB3 = Op.Controls.Item(8)
Set TBOp = Op.Controls.Item(5)
av.setHorasTrabajo Est, Subest, TBHoras.Text
av.setCurvasHoras Est, Subest, TBCH.Text
av.setOperarios Est, Subest, TBOp.Text
TurnoEst = 1
```

```
If OB1.Value = True Then
 TurnoEst = 1
End If
If OB2.Value = True Then
 TurnoEst = 2
End If
If OB3.Value = True Then
 TurnoEst = 3
End If
av.setTurnos Est, Subest, TurnoEst
OpTotal = TBOp.Text * TurnoEst
av.setOperarios Est, Subest, OpTotal
End Sub
```

ObtenerFrmSubOperarios

DESCRIPCIÓN: proceso contrario al de insertar, en este caso el formulario se carga con los valores del avión.

LLAMADAS ENTRANTES:

GestFrmSubestaciones.obtenerDatosSubestacion

GestFrmSubestaciones.ActualizarFrmOperariosEstacion

CÓDIGO:

```
Sub ObtenerFrmSubOperarios(Op As Frame, av As Avion, Est As Integer, Subest As Integer)
Dim TBHoras As TextBox
Dim TBCH As TextBox
Dim OB1 As OptionButton
Dim OB2 As OptionButton
Dim OB3 As OptionButton
Dim TBOp As TextBox
Dim TurnoEst As Integer
Set TBHoras = Op.Controls.Item(1)
Set TBCH = Op.Controls.Item(3)
Set OB1 = Op.Controls.Item(6)
Set OB2 = Op.Controls.Item(7)
Set OB3 = Op.Controls.Item(8)
Set TBOp = Op.Controls.Item(5)
TBHoras.Text = av.HorasTrabajo(Est, Subest)
TBCH.Text = av.CurvasHoras(Est, Subest)
If av.turnos(Est, Subest) = 1 Then
 OB1.Value = True
End If
```

```
If av.turnos(Est, Subest) = 2 Or av.turnos(Est, Subest) = 0 Then
 OB2.Value = True
End If
If av.turnos(Est, Subest) = 3 Then
 OB3.Value = True
End If
If av.turnos(Est, Subest) <> 0 Then
TBOp.Text = av.Operarios(Est, Subest) / av.turnos(Est, Subest)
End If
End Sub
```

ActualizarFrmOperariosEstaciones

DESCRIPCIÓN: se actualizan las variables del avión referentes a operaris en aquellos casos que la estación tenga subestación.

LLAMADAS ENTRANTES:

FrmSubestaciones.ActualizarSubestación_Click

LLAMADAS SALIENTES:

GestAv.CalcularHorasTrabajo av

GestAv.CalcularOperarios av

GestFrmSubestaciones.ObtenerFrmSubOperarios Op, av, Est, Subest

CÓDIGO:

```
Sub ActualizarFrmOperariosEstaciones(av As Avion, Est As Integer, Subest As Integer)
Dim Op As Frame
Dim pgs As Pages
Set pgs = FrmSubEstaciones.MultiSubestacion.Pages
If Subest <> 1 Then
 Set Op = pgs(Subest - 1).Controls.Item(13)
 Else
  Set Op = pgs(Subest - 1).Controls.Item(10)
 End If
GestAv.CalcularHorasTrabajo av
GestAv.CalcularOperarios av
GestFrmSubestaciones.ObtenerFrmSubOperarios Op, av, Est, Subest
End Sub
```

GestAv

VARIABLES

NOMBRE: AvEspeciales

DESCRIPCIÓN: array de enteros, donde almacenamos las posiciones en la escalera de los avoines especiales. Siempre tiene que estar ordenado.

SINTAXIS: Public AvEspeciales() As Integer

NOMBRE: NumeroAvEspeciales

DESCRIPCIÓN: variable que tiene que estar actualizada con el número de aviones especiales que hay en la escalera.

SINTAXIS: Public NumeroAvEspeciales As Integer

NOMBRE: aviones

DESCRIPCIÓN: array de aviiones en la que cada elemento es un avión especial, su ary "indice" es AvEspeciales.

SINTAXIS: Public aviones() As New Avion

NOMBRE: AvRR

DESCRIPCIÓN: objeto Avion que representa el avión de referencia.

SINTAXIS: Public AvRR As New Avion

NOMBRE: ACCC

DESCRIPCIÓN: objeto Avion que representa el avión de rcomienzo de curva.

SINTAXIS: Public ACCC As New Avion

MÉTODOS

CambioAvionReferencia

DESCRIPCIÓN: método que gestiona la modoficación de todas las variables involucradas en un cambio del número (posición en la escalera) del avión de referencia. Devuleve True si el cambio es correcto y se ha producido correctamente.

LLAMADAS ENTRANTES:

FrmGeneral.RR_AfterUpdate

FrmOpciones.CC_AfterUpdate

FrmOpciones.NRCC_AfterUpdate

Demo.PlanificacionDemo

LLAMADAS SALIENTES:

GestAv.isEspecial

CÓDIGO:

```
Function CambioAvionReferencia(RRef As Integer, NRAC As Integer, NAVCC As Integer) As Boolean
 Dim NAvRr As Integer
 NAvRr = RRef - NRAC + NAVCC
  If isEspecial(NAvRr) Then
   MsgBox " El Nuevo avión de referencia esta marcado como especial "
   FrmGeneral.RR.Text = planning.RepRef
   FrmOpciones.NRCC.Text = planning.NumeroRepAvCC
   FrmOpciones.CC.Text = planning.AvCC
  CambioAvionReferencia = False
   Exit Function
  End If
  If NAVCC <= planning.NumeroAviones Then
  AvRR.NumeroAvion = NAvRr
  planning.RepRef = RRef
  planning.NumeroRepAvCC = NRAC
  planning.AvCC = NAVCC
  CambioAvionReferencia = True
 Else
  CambioAvionReferencia = False
 End If
End Function
```

CalcularHorasTrabajo

DESCRIPCIÓN: se calculan las horas de trabajo a partir del número de operarios y las duraciones.

LLAMADAS ENTRANTES:

GestFrmEstaciones.ActualizarFrmOperariosEstaciones

GestFrmSubestaciones. ActualizarFrmOperariosEstaciones

CÓDIGO:

```
Sub CalcularHorasTrabajo(av As Avion)
Dim i As Integer
Dim j As Integer
Dim horas As Single
 For i = 0 To av.NumeroEstaciones - 1
  For j = 0 To av.NumeroMaximoSubestaciones
   horas = av.Duraciones(i, j) * (av.Operarios(i, j) * 8)
```

```
  av.setHorasTrabajo i, j, horas
 Next
Next
End Sub
```

CalcularOperarios

DESCRIPCIÓN: se calcula el número de operarios a partir de las horas de trabajo y las duraciones.

LLAMADAS ENTRANTES:

GestFrmEstaciones.ActualizarFrmOperariosEstaciones

GestFrmSubestaciones. ActualizarFrmOperariosEstaciones

CÓDIGO:

```
Sub CalcularOperarios(av As Avion)
Dim i As Integer
Dim j As Integer
Dim Operarios As Single
 For i = 0 To av.NumeroEstaciones - 1
  For j = 0 To av.NumeroMaximoSubestaciones
   Operarios = av.HorasTrabajo(i, j) / (av.Duraciones(i, j) * 8)
   av.setOperarios i, j, Operarios
  Next
 Next
End Sub
```

CalcularDias

DESCRIPCIÓN: se calculan los las duraciones a partir de los operarios y las horas de trabajo.

CÓDIGO:

```
Sub CalcularDias(av As Avion)
Dim i As Integer
Dim j As Integer
Dim Dias As Single
 For i = 0 To av.NumeroEstaciones - 1
  For j = 0 To av.NumeroMaximoSubestaciones
   Dias = av.HorasTrabajo(i, j) / (av.Operarios(i, j) * 8)
   av.setDuraciones i, j, Dias
  Next
 Next
End Sub
```

AñadirAvionEspecial

DESCRIPCIÓN: se añade (si es correcto) un elemento más en la posición adecuada a las variables avEspeciales y aviones.

LLAMADAS ENTRANTES:

FrmOpciones.OKButton_Click

LLAMADAS SALIENTES:

isEspecial(NA)

OrdenarAvionesEspeciales AvEspeciales

CÓDIGO:

```
Function AñadirAvionEspecial(NA As Integer) As Boolean
 If isEspecial(NA) Then
   AñadirAvionEspecial = False
   Exit Function
 End If
 Set aviones(NA) = New Avion
 aviones(NA).Inicializa
 aviones(NA).setCurvas 0, 0, 100
 aviones(NA).setPosiciones 0, 0, 1
 aviones(NA).Especial = True
 aviones(NA).EE = True
 aviones(NA).NumeroAvion = NA
  If NA > planning.AvCC Then
   aviones(NA).AplicaCurva = True
  Else
    aviones(NA).AplicaCurva = False
  End If
  AvEspeciales(NumeroAvEspeciales) = NA
  OrdenarAvionesEspeciales AvEspeciales
  NumeroAvEspeciales = NumeroAvEspeciales + 1
  ReDim Preserve AvEspeciales(NumeroAvEspeciales)
  AñadirAvionEspecial = True
End Function
```

EliminarAvionEspecial

DESCRIPCIÓN: se elimina un avion especial y se "readecuan" las variables que gestionan los avoines especiales.

LLAMADAS ENTRANTES:

FrmGeneral.NA_AfterUpdate

FrmAvEspecial.Eliminar_Click

LLAMADAS SALIENTES:

CÓDIGO:

```
Function EliminarAvionEspecial(NA As Integer) As Boolean
Dim i As Integer
Dim j As Integer
For i = 0 To NumeroAvEspeciales - 1
  If NA = AvEspeciales(i) Then
     For j = i + 1 To NumeroAvEspeciales - 1
      AvEspeciales(j - 1) = AvEspeciales(j)
     Next
  ReDim Preserve AvEspeciales(NumeroAvEspeciales - 1)
  NumeroAvEspeciales = NumeroAvEspeciales - 1
  EliminarAvionEspecial = True
  Set aviones(NA) = Nothing
  Exit Function
  End If
Next
  EliminarAvionEspecial = False
End Function
```

OrdenarAvionesEspeciales

DESCRIPCIÓN: Los aviones especiales tienen que estar ordenados de menor a mayor posición en la escalera de aviones. El algoritmo utilizado es el de ordenamiento por el método de la burbuja.

LLAMADAS ENTRANTES:

FrmOpciones.OKButton_Click

FrmAvEspecial.Eliminar_Click

FrmCopiar.UserForm_Initialize

GestAv.AñadirAvionEspecial

CÓDIGO:

```
Sub OrdenarAvionesEspeciales(v() As Integer)
' Ordenamiento de menor a mayor por el metodo de la burbuja
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim buffer As Integer
k = UBound(v)
For i = 0 To k
 For j = k To i + 1 Step -1
  If v(i) > v(j) Then
    buffer = v(i)
    v(i) = v(j)
```

```
    v(j) = buffer
  End If
 Next
Next
End Sub
```

isEspecial

DESCRIPCIÓN: informa si en la posición de la escalera pasada como parámetro tenemos un avión especial.

LLAMADAS ENTRANTES:

FrmOpciones.CC_AfterUpdate

FrmAvEspecial.UserFormInitialize

FrmAvEspecial.Eliminar_Click

FrmAvEspecial.Nest_Change

GestAv.CambioAvionReferencia

GestAv.AñadirAvionEspecial

CÓDIGO:

```
Function isEspecial(NA As Integer) As Boolean
Dim i As Integer
isEspecial = False
For i = 0 To NumeroAvEspeciales - 1
  If NA = AvEspeciales(i) Then
   isEspecial = True
   Exit Function
  End If
Next
End Function
```

CopiarAviones

DESCRIPCIÓN: Se copian las variables de un avión en otro.

LLAMADAS ENTRANTES:

RecuperarDatos

FrmEstaciones.UserForm_Initialize

FrmEstaciones.ActualizarEstaciones_Click

FrmCopiar.Aceptar_Click

frmEnlacesEspeciales.UserForm_Initialize

Demo.PlanificacionDemo

CÓDIGO:

```
Sub CopiarAviones(Origen As Avion, Destino As Avion)
Destino.NumeroEstaciones = Origen.NumeroEstaciones
```

```
Destino.NumeroMaximoSubestaciones = Origen.NumeroMaximoSubestaciones
Destino.Estaciones = Origen.Estaciones
Destino.Duraciones = Origen.Duraciones
Destino.Curvas = Origen.Curvas
Destino.Posiciones = Origen.Posiciones
Destino.Predecesoras = Origen.Predecesoras
Destino.Descripcion = Origen.Descripcion
Destino.Nombre = Origen.Nombre
Destino.EnlaceEstaciones = Origen.EnlaceEstaciones
Destino.EnlaceSubestaciones = Origen.EnlaceSubestaciones
Destino.CurvasHoras = Origen.CurvasHoras
Destino.Operarios = Origen.Operarios
Destino.turnos = Origen.turnos
Destino.HorasTrabajo = Origen.HorasTrabajo
End Sub
```

ValoresPorDefecto

DESCRIPCIÓN: existen variables que no interesan que esten a cero cuando se inicializan o reajustan su tamaño (matrices), pr lo que se utiliza este método para corregirlo.

LLAMADAS ENTRANTES:

RecuperarDatosAvion

Demo.PlanificacionDemo

CÓDIGO:

```
Sub ValoresPorDefecto(c As Variant, valor As Variant, Optional Dimension As Integer)
On Error GoTo fin:
 Dim i As Integer
 Dim j As Integer
 Dim k As Integer
 Dim x As Integer
 Dim y As Integer
 Dim z As Integer
 Dim e As Variant
 e = c
 If Dimension = 0 Then
  c = valor
  Exit Sub
 End If
 If Dimension = 1 Then
  x = UBound(c, 1)
  For i = 0 To x
```

```
  If c(i) = 0 Then
  c(i) = valor
  End If
 Next
 Exit Sub
 End If
 If Dimension = 2 Then
  x = UBound(c, 1)
  y = UBound(c, 2)
 For i = 0 To x
  For j = 0 To y
   If c(i, j) = 0 Then
   c(i, j) = valor
   End If
  Next
 Next
 End If
 If Dimension = 3 Then
  x = UBound(c, 1)
  y = UBound(c, 2)
  z = UBound(c, 3)
 For i = 0 To x
  For j = 0 To y
   For k = 0 To z
   If c(i, j, k) = 0 Then
   c(i, j, k) = valor
   End If
  Next
 Next
 Next
End If
Exit Sub
fin:
  c = e
End Sub
```

Control

MÉTODOS

PreparacionProgreso

DESCRIPCIÓN: ocultamos formulario general mostrando únicamente (en la parte superior izquierda de la pantalla) la parte que nos muestra el pogreso de la ejecución

LLAMADAS ENTRANTES:

Interno.Ejecutar

CÓDIGO:

```
Sub PreparacionProgreso()
FrmGeneral.Frame1.Visible = False
FrmGeneral.Image1.Visible = False
FrmGeneral.Ejecutar.Visible = False
FrmGeneral.Salir.Visible = False
FrmGeneral.Estaciones.Visible = False
FrmGeneral.Opciones.Visible = False
FrmGeneral.DemoButton.Visible = False
FrmGeneral.Label1.Visible = False
FrmGeneral.Label2.Visible = False
FrmGeneral.Label3.Visible = False
FrmGeneral.NA.Visible = False
FrmGeneral.NE.Visible = False
FrmGeneral.RR.Visible = False
FrmGeneral.Frame2.Top = 0
FrmGeneral.Frame2.Left = 0
FrmGeneral.Height = 200
FrmGeneral.Width = 300
FrmGeneral.Frame2.Visible = True
FrmGeneral.Repaint
FrmGeneral.Lnav = planning.NumeroAviones
FrmGeneral.Repaint
End Sub
```

RecuperarGeneral

DESCRIPCIÓN: método que contrarresta al anterior.

CÓDIGO:

```
Sub RecuperarGeneral()
FrmGeneral.Height = 406.5
```

```
FrmGeneral.Width = 555
FrmGeneral.Frame1.Visible = True
FrmGeneral.Image1.Visible = True
FrmGeneral.Ejecutar.Visible = True
FrmGeneral.Salir.Visible = True
FrmGeneral.Estaciones.Visible = True
FrmGeneral.Opciones.Visible = True
FrmGeneral.DemoButton.Visible = True
FrmGeneral.Label1.Visible = True
FrmGeneral.Label2.Visible = True
FrmGeneral.Label3.Visible = True
FrmGeneral.NA.Visible = True
FrmGeneral.NE.Visible = True
FrmGeneral.RR.Visible = True
FrmGeneral.Frame2.Visible = False
FrmGeneral.Repaint
End Sub
```

RecuperarEstacion

DESCRIPCIÓN: método que a partir de la posición de un elemento de un ListBox (opc=0) o de una tarea del proyecto (opc=1), con respecto a su origen (comienzo del ListBox o tarea ción), calcula el número de estación dentro del proceso de ensamblaje.

LLAMADAS ENTRANTES:

RecuperarEnlacesVerticales
frmEnlacesEspeciales.ELEA_Click
frmEnlacesEspeciales.ELEP_Click
frmEnlacesEspeciales.LBA_Change
frmEnlacesEspeciales.OKAA_Click
frmEnlacesEspeciales.OKAP_Click

CÓDIGO:

```
Function RecuperarEstacion(av As Avion, n As Integer, Optional opc As Integer) As Integer
' Valor devuelto [1, NE]
' n posicion [1,-]
Dim NE As Integer
Dim k As Integer
k = 0
 For NE = 1 To av.NumeroEstaciones
  If opc = 0 Then
   If av.Estaciones(NE - 1) = 0 Then
    k = k + 1
```

```
    If k = n Then
     GoTo fin:
    End If
   Else
     For j = 1 To av.Estaciones(NE - 1)
       k = k + 1
       If k = n Then
        GoTo fin:
       End If
     Next
   End If
   Else
    For j = 0 To av.Estaciones(NE - 1)
       k = k + 1
       If k = n Then
        GoTo fin:
       End If
     Next
   End If
 Next
fin:
   RecuperarEstacion = NE
End Function
```

RecuperarSubestacion

DESCRIPCIÓN: método que a partir de la posición de un elemento de un ListBox (opc=0) o de una tarea del proyecto (opc=1), con respecto a su origen (comienzo del ListBox o tarea avión), calcula el número de subestación dentro del proceso de ensamblaje. Si no pertenece a ninguna subestación devuelve 0.

LLAMADAS ENTRANTES:

RecuperarEnlacesVerticales
frmEnlacesEspeciales.ELEA_Click
frmEnlacesEspeciales.ELEP_Click
frmEnlacesEspeciales.LBA_Change
frmEnlacesEspeciales.OKAA_Click
frmEnlacesEspeciales.OKAP_Click

CÓDIGO:

```
Function RecuperarSubestacion(av As Avion, n As Integer, Optional opc As Integer) As Integer
' Valor devuelto [0, NSubest]
Dim NE As Integer
```

```
Dim k As Integer
k = 0
 For NE = 1 To av.NumeroEstaciones
  If opc = 0 Then
   If av.Estaciones(NE - 1) = 0 Then
    j = 0
    k = k + 1
    If k = n Then
     GoTo fin:
    End If
   Else
     For j = 1 To av.Estaciones(NE - 1)
      k = k + 1
      If k = n Then
       GoTo fin:
      End If
     Next
   End If
  Else
     For j = 0 To av.Estaciones(NE - 1)
      k = k + 1
      If k = n Then
       GoTo fin:
      End If
     Next
   End If
 Next
fin:
  RecuperarSubestacion = j
End Function
```

indiceEstacionSubestacion

DESCRIPCIÓN: nos devuelve los indices (estación y subestación) de la variables EnlaceEstaciones y EnlaceSubestaciones que tienen los valores de enlace coincidentes con los valores pasados como parámetros.

LLAMADAS ENTRANTES:

frmEnlacesEspeciales.ELEP_Click

frmEnlacesEspeciales.LBA_Change

CÓDIGO:

```
Sub indiceEstacionSubestacion(av As Avion, e As Integer, s As Integer)
```

```
' Valor devuelto [0, NE-1],[0,NS]
Dim i As Integer
Dim j As Integer
 For i = 0 To UBound(av.EnlaceEstaciones, 1)
  For j = 0 To UBound(av.EnlaceEstaciones, 2)
   If av.EnlaceEstaciones(i, j) = e And av.EnlaceSubestaciones(i, j) = s Then
    e = i
    s = j
   Exit Sub
   End If
  Next
 Next
e = -1
s = 0
End Sub
```

comparaAviones

DESCRIPCIÓN: método que compueba de forma rápida si dos aviones tienen el mismo número de estacoines y subestaciones.

LLAMADAS ENTRANTES:

Vinculacion.VinculacionVertical

CÓDIGO:

```
Function comparaAviones(av1 As Avion, av2 As Avion) As Boolean
Dim i As Integer
If av1.NumeroEstaciones = av2.NumeroEstaciones And av1.NumeroMaximoSubestaciones =
av2.NumeroMaximoSubestaciones Then
  comparaAviones = True
   For i = 0 To av1.NumeroEstaciones - 1
    If av1.Estaciones(i) <> av2.Estaciones(i) Then
     comparaAviones = False
     Exit Function
    End If
   Next
 Else
  comparaAviones = False
 End If
End Function
```

ControlDatosValidos

DESCRIPCIÓN: método de comprobación rápida de variables correctas.

LLAMADAS ENTRANTES:

FrmGeneral.Userform_Initialize

FrmEstaciones.ActualizaEstacion_Click

CÓDIGO:

```
Function ControlDatosValidos(av As Avion) As Boolean
 Dim i As Integer
 Dim j As Integer
  ControlDatosValidos = True
 If av.NumeroEstaciones = 0 Then
  ControlDatosValidos = False
  Exit Function
 End If
  For i = 0 To av.NumeroEstaciones - 1
  For j = 0 To av.Estaciones(i)
   If av.Curvas(i, j) < 0 Or av.Duraciones(i, j) < 0 Or av.Posiciones(i, j) <= 0 Then
   ControlDatosValidos = False
   End If
  Next
 Next
End Function
```

ControlEnlacesValidos

DESCRIPCIÓN: método que nos asegura que no enlazamos con una estación o subestación inexistente del avión anterior.

LLAMADAS ENTRANTES:

frmEnlacesEspeciales.OKAA_Click

frmEnlacesEspeciales.OKAP_Click

CÓDIGO:

```
Function ControlEnlacesValidos(av As Avion, Een As Integer, Sen As Integer, Optional opc As Integer)
As Boolean
Dim avAnt As Avion
 ControlEnlacesValidos = True
If Een <= 0 Or Sen < 0 Then
   ControlEnlacesValidos = False
    Exit Function
End If
If opc = 0 Then
 If GestAv.aviones(av.NumeroAvion - 1).EE = False Then
     Set avAnt = AvRR
 Else
```

```
    Set avAnt = GestAv.aviones(av.NumeroAvion - 1)
 End If
Else
   Set avAnt = av
End If
If Een > avAnt.NumeroEstaciones Or Sen > avAnt.NumeroMaximoSubestaciones Then
  ControlEnlacesValidos = False
End If
End Function
```

EnlacesDefecto

DESCRIPCIÓN: método que facilita la creación de enlaces con el avión anterior, para ello y utilizando como referencia el avión de referencia y en función del número de estaciones y subestaciones del avión establece unos enlaces que si bien no tienen porque ser los correctos se garantiza que son no váildos (enlace inexistente).

LLAMADAS ENTRANTES:

FrmEstaciones.ActualizaEstacion_Click

FrmOpciones.OKButton_Click

FrmCopiar.Aceptar_Click

CÓDIGO:

```
Sub EnlacesDefecto(av As Avion)
Dim avAnt As Avion
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim z As Integer
If av.NumeroAvion <= 1 Then
 Exit Sub
End If
If av.NumeroAvion <= planning.NumeroAviones Then
 If GestAv.aviones(av.NumeroAvion - 1).Especial = False Then
  Set avAnt = AvRR
 Else
   Set avAnt = GestAv.aviones(av.NumeroAvion - 1)
 End If
Else
  Set avAnt = AvRR
End If
If avAnt.NumeroEstaciones < av.NumeroEstaciones Then
 k = avAnt.NumeroEstaciones
```

```
Else
 k = av.NumeroEstaciones
End If
For i = 0 To k - 1
 If avAnt.Estaciones(i) < av.Estaciones(i) Then
 z = avAnt.Estaciones(i)
 Else
  z = av.Estaciones(i)
 End If
 For j = 0 To z
  av.setEnlaceEstaciones i, j, i + 1
  av.setEnlaceSubestaciones i, j, j
 Next
Next
End Sub
```

CalculoHoras

DESCRIPCIÓN: dados como parámetros los días y los operarios se calculan las horas de trabajo.

LLAMADAS ENTRANTES:

Demo.PlanificacionDemo

CÓDIGO:

```
Function CalculoHoras(d As Single, o As Single) As Single
 CalculoHoras = d * o * 8
End Function
```

CalculoDias

DESCRIPCIÓN: dados como parámetros las horas de trabajo y los operarios se calculan los días.

LLAMADAS ENTRANTES:

CÓDIGO:

```
Function CalculoDias(h As Single, o As Single) As Single
 If o = 0 Then
  CalculoDias = 0
  Exit Function
 End If
 CalculoDias = h / (o * 8)
End Function
```

CalculoOperarios

DESCRIPCIÓN: dados como parámetros las horas de trabajo y los días se calculan los operarios.

LLAMADAS ENTRANTES:

Inicialización.RecuperarDatosAvion

Duracion.CalculoOperarios

CÓDIGO:

```
Function CalculoOperarios(h As Single, d As Single) As Single
 If d = 0 Then
  CalculoOperarios = 0
  Exit Function
 End If
CalculoOperarios = h / (d * 8)
End Function
```

Demo

MÉTODOS

PlanificacionDemo

DESCRIPCIÓN: método que inicializa las variables del avión de referencia con unos valores demostrativos de una FAL. Puede ser utilizado como punto de partida para una planificación realista.

LLAMADAS ENTRANTES:

FrmGeneral.DemoButton_Click

LLAMADAS SALIENTES:

GestAv.ValoresPorDefecto Op, 10, 2; GestAv.ValoresPorDefecto e, 10, 2

Control.CalculoHoras(Duracion(i, j), Op(i, j))

GestAv.CopiarAviones avDemo, AvRR

GestAv.CambioAvionReferencia planning.RepRef, planning.NumeroRepAvCC, planning.AvCC

CÓDIGO:

```
Sub PlanificacionDemo()
 Dim avDemo As New Avion
 Dim i As Integer
 Dim j As Integer
 Dim Estaciones() As Integer
 Dim Curva(6, 3) As Single
 Dim Desc(6, 3) As String
 Dim Nombre(6, 3) As String
 Dim Duracion(6, 3) As Single
 Dim Posiciones(6, 3) As Integer
 Dim Predecesora(6, 3, 2) As Integer
 Dim e() As Integer
 Dim Op() As Single
```

```
Dim Ht() As Single
Dim NumAviones As Integer
Dim NumEstaciones As Integer
Dim RR As Integer
Dim CC As Integer
Dim RCC As Integer
Dim NumMaxSub As Integer
NumEstaciones = 7
NumMaxSub = 3
NumAviones = 10
RR = 1
CC = 4
RCC = 1
ReDim Estacion(NumEstaciones - 1)
Estacion(0) = 3
Estacion(1) = 0
Estacion(2) = 3
Estacion(3) = 0
Estacion(4) = 0
Estacion(5) = 0
Estacion(6) = 2
Duracion(0, 0) = 0
Duracion(0, 1) = 6.5
Duracion(0, 2) = 6.5
Duracion(0, 3) = 6.5
Duracion(1, 0) = 13
Duracion(2, 0) = 0
Duracion(2, 1) = 13
Duracion(2, 2) = 6.5
Duracion(2, 3) = 4
Duracion(3, 0) = 6.5
Duracion(4, 0) = 8
Duracion(5, 0) = 6.5
Duracion(6, 0) = 0
Duracion(6, 1) = 1.5
Duracion(6, 2) = 7
Predecesora(0, 1, 0) = 1
Predecesora(0, 2, 0) = 1
Predecesora(0, 3, 0) = 1
Predecesora(2, 1, 0) = 1
```

Predecesora(2, 2, 1) = 1
Predecesora(2, 3, 1) = 1
Predecesora(6, 1, 0) = 1
Predecesora(6, 2, 1) = 1
Curva(0, 0) = 0
Curva(0, 1) = 85
Curva(0, 2) = 85
Curva(0, 3) = 85
Curva(1, 0) = 85
Curva(2, 0) = 0
Curva(2, 1) = 85
Curva(2, 2) = 85
Curva(2, 3) = 85
Curva(3, 0) = 85
Curva(4, 0) = 85
Curva(5, 0) = 85
Curva(6, 0) = 0
Curva(6, 1) = 85
Curva(6, 2) = 85
Posiciones(0, 0) = 1
Posiciones(0, 1) = 1
Posiciones(0, 2) = 1
Posiciones(0, 3) = 1
Posiciones(1, 0) = 1
Posiciones(2, 0) = 1
Posiciones(2, 1) = 1
Posiciones(2, 2) = 1
Posiciones(2, 3) = 1
Posiciones(3, 0) = 1
Posiciones(4, 0) = 1
Posiciones(5, 0) = 1
Posiciones(6, 0) = 1
Posiciones(6, 1) = 1
Posiciones(6, 2) = 1
Desc(0, 0) = "Join up"
Desc(0, 1) = "Fuselage join up"
Desc(0, 2) = "VTP-HTP join up"
Desc(0, 3) = "Wing join up"
Desc(1, 0) = "Aircraft integration & equipment"
Desc(2, 0) = "System Tests"

```
 Desc(2, 1) = "System Tests Indoors"
 Desc(2, 2) = "System Tests Outdoor"
 Desc(2, 3) = "Engines & Propeller equipment"
 Desc(3, 0) = "Powerplant install.&Interior Furnishing"
 Desc(4, 0) = "Flight Line"
 Desc(5, 0) = "Painting"
 Desc(6, 0) = "Handover"
 Desc(6, 1) = "w.d. Handover to AMC"
 Desc(6, 2) = "Handover to Customer"
 Nombre(0, 0) = "Estación 1"
 Nombre(0, 1) = "Estación 1.1"
 Nombre(0, 2) = "Estación 1.2"
 Nombre(0, 3) = "Estación 1.3"
 Nombre(1, 0) = "Estación 2&3"
 Nombre(2, 0) = "Estación 4"
 Nombre(2, 1) = "Estación 4.1"
 Nombre(2, 2) = "Estación 4.2"
 Nombre(2, 3) = "Estación 4.3"
 Nombre(3, 0) = "Estación 5"
 Nombre(4, 0) = "Estación 6"
 Nombre(5, 0) = "Estación 7"
 Nombre(6, 0) = "Estación 8"
 Nombre(6, 1) = "Estación 8.1"
 Nombre(6, 2) = "Estación 8.2"
planning.NumeroAviones = NumAviones
planning.RepRef = RR
planning.AvCC = CC
planning.AvFC = NumAviones
planning.NumeroRepAvCC = RCC
NumeroAvEspeciales = 0
ReDim AvEspeciales(0)
ReDim aviones(1 To NumAviones)
  FrmGeneral.NA.Text = NumAviones
  FrmGeneral.NE.Text = NumEstaciones
  FrmGeneral.RR.Text = RR
  FrmOpciones.CC.Text = CC
  FrmOpciones.NRCC.Text = RCC
  ReDim e(NumEstaciones - 1, NumMaxSub)
  ReDim Op(NumEstaciones - 1, NumMaxSub)
  ReDim Ht(NumEstaciones - 1, NumMaxSub)
```

```
  avDemo.NumeroEstaciones = NumEstaciones
  avDemo.NumeroMaximoSubestaciones = NumMaxSub
  avDemo.Estaciones = Estacion
  avDemo.Duraciones = Duracion
  avDemo.Curvas = Curva
  avDemo.Posiciones = Posiciones
  avDemo.Predecesoras = Predecesora
  avDemo.Descripcion = Desc
  avDemo.Nombre = Nombre
  avDemo.EnlaceEstaciones = e
  avDemo.EnlaceSubestaciones = e
  GestAv.ValoresPorDefecto e, 2, 2
  avDemo.turnos = e
  avDemo.CurvasHoras = Curva
 GestAv.ValoresPorDefecto Op, 10, 2
  avDemo.Operarios = Op
  For i = 0 To NumEstaciones - 1
   For j = 0 To NumMaxSub
    Ht(i, j) = Control.CalculoHoras(Duracion(i, j), Op(i, j))
   Next
  Next
  avDemo.HorasTrabajo = Ht
  GestAv.CopiarAviones avDemo, AvRR
  GestAv.CambioAvionReferencia planning.RepRef, planning.NumeroRepAvCC, planning.AvCC
End Sub
```

Interno

MÉTODOS

Ejecutar

DESCRIPCIÓN: método que controla la ejecución del programa. Modifica el formulario de progreso y llama a los métodos encargados de las distintas tareas de ejecución.

LLAMADAS ENTRANTES:

FrmGeneral.Ejecutar_Click

LLAMADAS SALIENTES:

Control.PreparacionProgreso

Inicializacion.valoresRecursoOperarios False

Tareas.AsignarTareas

Duracion.AsignarDuracion

Vinculacion.AsignarVinculacion

CÓDIGO:

```
Sub Ejecutar()
Application.SelectAll
Application.RowDelete
Control.PreparacionProgreso
Inicializacion.valoresRecursoOperarios False
FrmGeneral.CheckBox1.Visible = True
FrmGeneral.CheckBox1.ForeColor = vbRed
FrmGeneral.Repaint
Tareas.AsignarTareas
FrmGeneral.CheckBox1 = True
FrmGeneral.CheckBox1.ForeColor = vbBlack
FrmGeneral.Repaint
FrmGeneral.CheckBox2.Visible = True
FrmGeneral.CheckBox2.ForeColor = vbRed
FrmGeneral.Repaint
Duracion.AsignarDuracion
FrmGeneral.CheckBox2 = True
FrmGeneral.CheckBox2.ForeColor = vbBlack
FrmGeneral.Repaint
FrmGeneral.CheckBox3.Visible = True
FrmGeneral.CheckBox3.ForeColor = vbRed
FrmGeneral.Repaint
Vinculacion.AsignarVinculacion
FrmGeneral.CheckBox3 = True
FrmGeneral.CheckBox3.ForeColor = vbBlack
FrmGeneral.Repaint
 If FrmOpciones.EnlaceTodos Then
    Application.GanttBarLinks pjToTop
  Else
    Application.GanttBarLinks
 End If
If FrmOpciones.TareaAviones Then
Application.SelectAll
OutlineHideSubTasks
Application.SelectCell 1, 3, False
End If
If FrmOpciones.TareaEstaciones Then
```

```
Application.SelectAll
OutlineHideSubTasks
OutlineShowSubTasks
Application.SelectCell 1, 3, False
End If
End Sub
```

Tareas

MÉTODOS

AsignarTareas

DESCRIPCIÓN: escribe las tareas de la escalera y las jerarquiza.

LLAMADAS ENTRANTES:

Interno.Ejecutar

LLAMADAS SALIENTES:

Tareas.BarraTareas

CÓDIGO:

```
Sub AsignarTareas()
 Dim ObjTasks As Tasks
 Dim ObjTask As Task
 Dim IntAvion As Integer
 Dim IntEstacion As Integer
 Dim IntSubestacion As Integer
 Dim StrTarea As String
Dim i As Integer
 For i = 1 To planning.NumeroAviones
   If Not aviones(i).Especial Then
     aviones(i).NumeroEstaciones = AvRR.NumeroEstaciones
     aviones(i).NumeroMaximoSubestaciones = AvRR.NumeroMaximoSubestaciones
     aviones(i).Estaciones = AvRR.Estaciones
     aviones(i).Duraciones = AvRR.Duraciones
     aviones(i).Curvas = AvRR.Curvas
     aviones(i).Posiciones = AvRR.Posiciones
     aviones(i).Predecesoras = AvRR.Predecesoras
     aviones(i).Nombre = AvRR.Nombre
     aviones(i).Descripcion = AvRR.Descripcion
     aviones(i).CurvasHoras = AvRR.CurvasHoras
     aviones(i).turnos = AvRR.turnos
```

```
    aviones(i).Operarios = AvRR.Operarios
    aviones(i).HorasTrabajo = AvRR.HorasTrabajo
   End If
Next
 Application.SelectCell 1, 3, False
 Set ObjTasks = ActiveProject.Tasks
  ObjTasks.Add "A/C " & Format(1, "000")
 Set ObjTask = ActiveCell.Task
 aviones(1).NumeroAvion = 1
 Set aviones(1).TaskAvion = ObjTask
 aviones(1).RDTask
  For IntAvion = 1 To planning.NumeroAviones
   FrmGeneral.Lav.Caption = IntAvion
   FrmGeneral.Repaint
   If ObjTask.OutlineLevel = 1 Then
    GoTo LnEstaciones
   End If
   If ObjTask.OutlineLevel > 1 Then
     GoTo LnAviones
    End If

LnFb:   Next IntAvion
          BarraTareas
          Exit Sub

LnAviones:
  Application.SelectCellDown
  StrTarea = "A/C " & Format(IntAvion, "000")
  ObjTasks.Add StrTarea
  Set ObjTask = ActiveCell.Task
  aviones(IntAvion).NumeroAvion = IntAvion
  Set aviones(IntAvion).TaskAvion = ObjTask
  aviones(IntAvion).RDTask
  While ObjTask.OutlineLevel > 1
  ObjTask.OutlineOutdent
  Wend
 GoTo LnEstaciones

LnEstaciones:
   For IntEstacion = 1 To aviones(IntAvion).NumeroEstaciones
```

```
    Application.SelectCellDown
     If FrmOpciones.AsignaDefecto Then
      StrTarea = "A/C " & Format(IntAvion, "000") & " ST " & Format(IntEstacion, "00")
     Else
      StrTarea = "A/C "& Format(IntAvion, "000") & " " & aviones(IntAvion).Nombre(IntEstacion - 1, 0)
     End If
     ObjTasks.Add StrTarea
     Set ObjTask = ActiveCell.Task
     aviones(IntAvion).setTakGlobal IntEstacion - 1, 0, ObjTask
     If ObjTask.OutlineLevel = 1 Then
      ObjTask.OutlineIndent
     End If
     If ObjTask.OutlineLevel = 3 Then
      ObjTask.OutlineOutdent
     End If
     GoTo LnSubestaciones

lnIE:   Next IntEstacion
        GoTo LnFb

LnSubestaciones:
     For IntSubestacion = 1 To aviones(IntAvion).Estaciones(IntEstacion - 1)
     Application.SelectCellDown
     If FrmOpciones.AsignaDefecto Then
      StrTarea = "A/C " & Format(IntAvion, "000") & " ST " & Format(IntEstacion, "00") & "." &
Format(IntSubestacion, "00")
     Else
       StrTarea = "A/C " & Format(IntAvion, "000") & " " & aviones(IntAvion).Nombre(IntEstacion - 1,
IntSubestacion)
      End If
     ObjTasks.Add StrTarea
     Set ObjTask = ActiveCell.Task
     aviones(IntAvion).setTakGlobal IntEstacion - 1, IntSubestacion, ObjTask
     If ObjTask.OutlineLevel = 2 Then
       ObjTask.OutlineIndent
     End If
    Next IntSubestacion
    GoTo lnIE
End Sub
```

BarraTareas

DESCRIPCIÓN: guarda en los campos de tareas de project la información que posteriormete permitirá arancar la alpicación con los datos de usuario correctos. También se tienen en cuenta las opciones de visualización escogidas por el usuario en el menú de opciones.

LLAMADAS ENTRANTES:

Tareas.AsignarTareas

CÓDIGO:

```
Sub BarraTareas()
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim c As Single
Dim d As Single
Application.GanttBarSize (pjBarSize14)
Application.SelectAll
Application.ColumnAlignment pjLeft
Application.SelectCell 1, 3, False
For i = 1 To planning.NumeroAviones
 FrmGeneral.Lav.Caption = i
 FrmGeneral.Repaint
 If aviones(i).Especial Then
  aviones(i).TaskAvion.Flag2 = True
  aviones(i).TaskAvion.Flag3 = aviones(i).AplicaCurva
 End If
 If aviones(i).EE Then
 aviones(i).TaskAvion.Flag4 = True
 End If
 If i = planning.AvCC Then
   aviones(i).TaskAvion.Flag1 = True
   aviones(i).TaskAvion.Number6 = planning.NumeroRepAvCC
   aviones(i).TaskAvion.Number7 = planning.RepRef
   aviones(i).TaskAvion.Number8 = planning.AvFC
 End If
 For j = 0 To aviones(i).NumeroEstaciones - 1
   For k = 0 To aviones(i).Estaciones(j)
    aviones(i).TaskGlobal(j, k).Number2 = aviones(i).Curvas(j, k)
    aviones(i).TaskGlobal(j, k).Number3 = aviones(i).CurvasHoras(j, k)
    aviones(i).TaskGlobal(j, k).Number4 = aviones(i).Posiciones(j, k)
    aviones(i).TaskGlobal(j, k).Number5 = aviones(i).turnos(j, k)
    aviones(i).TaskGlobal(j, k).Text1 = aviones(i).Nombre(j, k)
```

```
    aviones(i).TaskGlobal(j, k).Text2 = aviones(i).Descripcion(j, k)
    If aviones(i).Estaciones(j) = 0 Then
     If FrmOpciones.DescAmbas Or FrmOpciones.DescEstaciones Then
      Application.GanttBarFormat TaskId:=aviones(i).TaskGlobal(j, k).Id, StartShape:=0, StartType:=0,
StartColor:=1, MiddleShape:=1, MiddlePattern:=3, MiddleColor:=1, EndShape:=0, EndType:=0,
EndColor:=0, rightText:="Texto2"
     Else
     Application.GanttBarFormat TaskId:=aviones(i).TaskGlobal(j, k).Id, StartShape:=0, StartType:=0,
StartColor:=1, MiddleShape:=1, MiddlePattern:=3, MiddleColor:=1, EndShape:=0, EndType:=0,
EndColor:=0
     End If
    End If
    If k = 0 And aviones(i).Estaciones(j) > 0 Then
    Application.SelectCell aviones(i).TaskGlobal(j, k).Id, 3, False
     If FrmOpciones.DescAmbas Or FrmOpciones.DescEstaciones Then
      Application.GanttBarFormat StartShape:=3, StartType:=0, StartColor:=1, MiddleShape:=1,
MiddlePattern:=1, MiddleColor:=1, EndShape:=3, EndType:=0, EndColor:=1, rightText:="texto2"
     Else
      Application.GanttBarFormat StartShape:=3, StartType:=0, StartColor:=1, MiddleShape:=1,
MiddlePattern:=1, MiddleColor:=1, EndShape:=3, EndType:=0, EndColor:=1
     End If
    End If
    If k <> 0 And (FrmOpciones.DescAmbas Or FrmOpciones.DescSubestaciones) Then
     Application.GanttBarFormat TaskId:=aviones(i).TaskGlobal(j, k).Id, rightText:="texto2"
    End If
   Next
  Next
 Next
Application.SelectAll
OutlineHideSubTasks
Application.GanttBarFormat StartShape:=2, StartType:=1, StartColor:=0, MiddleShape:=1,
MiddlePattern:=3, MiddleColor:=5, EndShape:=2, EndType:=1, EndColor:=0, LeftText:="Nombre"
'Application.GanttBarFormat LeftText:="Nombre"
OutlineShowSubTasks
OutlineShowSubTasks
Application.SelectCell 1, 3, False
End Sub
```

Duracion

MÉTODOS

AsignarDuracion

DESCRIPCIÓN: método principal de duración. Se controlan y secuencian las llamadas a otros métodos que introducen en Project los valores de duraciones, horas de trabajo y operarios con los parámetros adecuados.

LLAMADAS ENTRANTES:

Interno.Ejecutar

LLAMADAS SALIENTES:

DuracionCurva(ai, h, ch); DuracionCurva(ai, d, c)

IntroducirOperarios i, j, k, Valdur / (60 * 8), NHoras

CÓDIGO:

```
Sub AsignarDuracion()
Dim ai As Integer
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim c As Single
Dim ch As Single
Dim NHoras As Single
Dim Valdur As Single
Dim d As Single
Dim h As Single
For i = 1 To planning.NumeroAviones
 FrmGeneral.Lav.Caption = i
 FrmGeneral.Repaint
  ai = i
 If i < planning.AvCC Then
  If aviones(i).Especial = False Then
   ai = planning.AvCC
  Else
    If aviones(i).AplicaCurva Then
      ai = planning.AvCC
    Else
      ai = i
    End If
  End If
```

```
 End If
  If i >= planning.AvFC Then
  ai = planning.AvFC
 End If
 For j = 0 To aviones(i).NumeroEstaciones - 1
   For k = 0 To aviones(i).Estaciones(j)
     d = aviones(i).Duraciones(j, k)
     h = aviones(i).HorasTrabajo(j, k)
     c = aviones(i).Curvas(j, k)
     ch = aviones(i).CurvasHoras(j, k)
     If c = 0 Then
      c = 100
     End If
     If ch = 0 Then
      ch = 100
     End If
    If Not (aviones(i).Estaciones(j) > 0 And k = 0) Then
     Valdur = DuracionCurva(ai, d, c)
     aviones(i).TaskGlobal(j, k).Duration = Valdur
     NHoras = DuracionCurva(ai, h, ch) / (60 * 8)
    IntroducirOperarios i, j, k, Valdur / (60 * 8), NHoras
     aviones(i).TaskGlobal(j, k).Number1 = NHoras
    End If
   Next
  Next
 Next
End Sub
```

DuracionCurva

DESCRIPCIÓN: método que calcula de duración teniendo en cuenta la reducción de tiempos debido a la curva de aprendizaje. Nav es la posición del avión (no repetición!) en la escalera. Duración y ValorCurva son la duración y el factor de reducción de la estación o subestación equivalente del avión de comienzo de curva.

LLAMADAS ENTRANTES:

Duracion.AsignarDuracion

CÓDIGO:

```
Function DuracionCurva(Nav As Integer, Duracion As Single, ValorCurva As Single) As Double
Dim Log10 As Double
Log10 = Log(CDbl(ValorCurva)) / Log(10#)
If Nav >= planning.AvCC And aviones(Nav).Especial = False Then
```

```
DuracionCurva = 8 * 60 * Duracion * ((Nav - planning.AvCC + planning.NumeroRepAvCC) /
planning.RepRef) ^ ((Log10 - 2) / 0.30103)
Else
  If aviones(Nav).Especial And aviones(Nav).AplicaCurva Then
  DuracionCurva = 60 * 8 * Duracion * ((Nav - planning.AvCC + planning.NumeroRepAvCC) /
planning.RepRef) ^ ((Log10 - 2) / 0.30103)
  Else
  DuracionCurva = Duracion * 60 * 8
 End If
End If
End Function
```

CalculoInverso

DESCRIPCIÓN: Realiza una entrada en la curva de forma "inversa", es decir, se calcula la duración del avión de referencia a partir de un avión cualquiera de la curva.

LLAMADAS ENTRANTES:

RecuperarDatosRef

RecuperarDatosRefHoras

CÓDIGO:

```
Function CalculoInverso(Nav As Integer, DuracionCurva As Double, ValorCurva As Integer) As Double
Dim Log10 As Double
Log10 = Log(CDbl(ValorCurva)) / Log(10#)
CalculoInverso = DuracionCurva * ((Nav - planning.AvCC + planning.NumeroRepAvCC) /
planning.RepRef) ^ (-(Log10 - 2) / 0.30103)
End Function
```

Redondeo

DESCRIPCIÓN: función "genérica" que redondea un número "x" con la precisión decimal "p"

LLAMADAS ENTRANTES:

Inicialización.RecuperarDatosAvion

Inicialización.RecuperarDatosRef

Inicialización.RecuperarDatosRefHoras

Duracion.IntroducirOperarios

CÓDIGO:

```
Function Redondeo(x As Single, p As Integer) As Single
Dim ax As Integer
Dim m As Long
Dim rx As Single
ax = Int(x)
rx = x - ax
```

```
m = 10 ^ p
Redondeo = ax + (CInt(rx * m) / m)
End Function
```

IntroducirOperarios

DESCRIPCIÓN: introducimos en Project el número de operarios en la tarea correspondiente. Si no se quieren decimales el redondeo siempre se produce al entero próximo superior.

LLAMADAS ENTRANTES:

Duracion.AsignarDuracion

LLAMADAS SALIENTES:

Control.CalculoOperarios(h, d)

Redondeo(Op, 2)

CÓDIGO:

```
Sub IntroducirOperarios(i As Integer, j As Integer, k As Integer, d As Single, h As Single)
Dim Op As Single
Dim Str As String
Dim Opr As Single
 Op = Control.CalculoOperarios(h, d)
 If FrmOpciones.CBRedondeo = True Then
   Opr = Fix(Op)
    If Opr <> Op Then
     Op = Opr + 1
    End If
 End If
 Op = Redondeo(Op, 2)
 Str = "Operarios[" & Op & "]"
 aviones(i).TaskGlobal(j, k).SetField pjTaskResourceNames, Str
End Sub
```

Vinculacion

VARIABLES

NOMBRE: ECR

DESCRIPCIÓN: entero que se corresponde con la estación crítica

SINTAXIS: Public ECR As Integer

NOMBRE: SCR

DESCRIPCIÓN: entero que se corresponde con la subestación crítica

SINTAXIS: Public SCR As Integer

MÉTODOS

AsignarVinculacion

DESCRIPCIÓN: método principal de vinculación, se encarga de llamar a otros método con tareas más específicas de la forma y en el orden correctos.Para agilizar la ejecución trabajamos con la hipotesis de no cruce de curvas en cálculo de la tarea crítica, es decir suponemos que la reducción de las duraciones no hace cambiar la tarea crítica de la escalera.

LLAMADAS ENTRANTES:

Interno.Ejecutar

LLAMADAS SALIENTES:

CalcularCritica AvRR, ECR, SCR

VinculacionSubestaciones i

VinculacionHorizontal i

VinculacionVertical i, opc:=1; VinculacionVertical i

PegarAviones i, opc:=1; PegarAviones i

DelimitarFecha

CÓDIGO:

```
Sub AsignarVinculacion()
Dim i As Integer
Dim j As Integer
Dim Normal As Boolean
CalcularCritica AvRR, ECR, SCR
For i = 1 To planning.NumeroAviones
FrmGeneral.Lav.Caption = i
FrmGeneral.Repaint
 If aviones(i).EE Then
  VinculacionSubestaciones i
  VinculacionHorizontal i
  VinculacionVertical i, opc:=1
  PegarAviones i, opc:=1
    Else
  VinculacionSubestaciones i
  VinculacionHorizontal i
   VinculacionVertical i
   PegarAviones i
  End If
Next
```

DelimitarFecha
End Sub

VinculacionHorizontal

DESCRIPCIÓN: vinculamos las tareas pertenecientes a un mismo avión.
LLAMADAS ENTRANTES:
Vinculacion.AsignarVinculacion
CÓDIGO:

```
Sub VinculacionHorizontal(i As Integer)
Dim j As Integer
For j = 0 To aviones(i).NumeroEstaciones - 2
  aviones(i).TaskGlobal(j, 0).LinkSuccessors aviones(i).TaskGlobal(j + 1, 0)
Next
End Sub
```

VinculacionVertical

DESCRIPCIÓN: se vinculan las tareas entre los distintos aviones. La vinculación es distinta en función si el avión es especial o no. También hay que tener en cuenta si el avión al cual se va a enlazar es especial
LLAMADAS ENTRANTES:
Vinculacion.AsignarVinculacion
LLAMADAS SALIENTES:
Control.comparaAviones(GestAv.aviones(i), GestAv.aviones(i - 1))
CÓDIGO:

```
Sub VinculacionVertical(i As Integer, Optional opc As Integer)
Dim j As Integer
Dim k As Integer
Dim p As Integer
Dim iguales As Boolean
If i = 1 Then
 Exit Sub
End If
If opc <> 0 Then
 iguales = Control.comparaAviones(GestAv.aviones(i), GestAv.aviones(i - 1))
 If iguales Then
  GoTo Igual:
 End If
End If
For j = 0 To aviones(i).NumeroEstaciones - 1
 If opc = 0 Then
 If aviones(i).Estaciones(j) = 0 Then
```

```
 k = 0
 p = aviones(i).Posiciones(j, k)
 If i - p > 0 Then
  If aviones(i - p).Especial = False Then
   aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - p).TaskGlobal(j, k)
  End If
 End If
 Else
  For k = 1 To aviones(i).Estaciones(j)
  p = aviones(i).Posiciones(j, k) * aviones(i).Posiciones(j, 0)
   If i - p > 0 Then
   If aviones(i - p).Especial = False Then
    aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - p).TaskGlobal(j, k)
   End If
  End If
 Next
 End If
 Else
  If aviones(i).Estaciones(j) = 0 Then
   k = 0
  If aviones(i).EnlaceEstaciones(j, k) <> 0 Then
   aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - 1).TaskGlobal(aviones(i).EnlaceEstaciones(j,
k) - 1, aviones(i).EnlaceSubestaciones(j, k))
  End If
  Else
   For k = 1 To aviones(i).Estaciones(j)
    If aviones(i).EnlaceEstaciones(j, k) <> 0 Then
    aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - 1).TaskGlobal(aviones(i).EnlaceEstaciones(j,
k) - 1, aviones(i).EnlaceSubestaciones(j, k))
     End If
   Next
  End If
 End If
Next
Exit Sub
Igual:
  For j = 0 To aviones(i).NumeroEstaciones - 1
   If aviones(i).Estaciones(j) = 0 Then
   k = 0
   aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - 1).TaskGlobal(j, k)
```

```
  Else
   For k = 1 To aviones(i).Estaciones(j)
   aviones(i).TaskGlobal(j, k).LinkPredecessors aviones(i - 1).TaskGlobal(j, k)
   Next
  End If
Next
End Sub
```

VinculacionSubestaciones

DESCRIPCIÓN: vinculamos las subestacoines dentro de un mismo avión. Aen este caso los enlaces puede ser distintos en función de la variable Predecesoras.

LLAMADAS ENTRANTES:

Vinculacion.AsignarVinculacion

CÓDIGO:

```
Sub VinculacionSubestaciones(i As Integer)
Dim j As Integer
Dim k As Integer
Dim z As Integer
For j = 0 To aviones(i).NumeroEstaciones - 1
 For k = 2 To aviones(i).Estaciones(j)
  For z = 1 To k - 1
   If aviones(i).Predecesoras(j, k, z) = 1 Then
   aviones(i).TaskGlobal(j, z).LinkSuccessors aviones(i).TaskGlobal(j, k)
  End If
  Next
 Next
Next
End Sub
```

PegarAviones

DESCRIPCIÓN: en este método se evita el efecto “troceo”de los aviones. Para ello, las tareas anteriores a la crítica empiezan lo más tarde posible y las posteriores lo antes posible.

LLAMADAS ENTRANTES:

Vinculacion.AsignarVinculacion

LLAMADAS SALIENTES:

CalcularCritica aviones(i), EC, SC

CÓDIGO:

```
Sub PegarAviones(i As Integer, Optional opc As Integer)
Dim EC As Integer
Dim SC As Integer
```

```
Dim k As Integer
Dim j As Integer
If opc = 0 Then
  EC = ECR
  SC = SCR
Else
   CalcularCritica aviones(i), EC, SC
End If
For j = 0 To aviones(i).NumeroEstaciones - 1
If j < EC Then
   If aviones(i).Estaciones(j) = 0 Then
   aviones(i).TaskGlobal(j, 0).ConstraintType = pjALAP
   Else
   For k = 1 To aviones(i).Estaciones(j)
     aviones(i).TaskGlobal(j, k).ConstraintType = pjALAP
   Next
   End If
End If
If j = EC Then
 If aviones(i).Estaciones(j) > 0 Then
   For k = 1 To SC - 1
    aviones(i).TaskGlobal(j, k).ConstraintType = pjALAP
   Next
   For k = SC To aviones(i).Estaciones(j)
    aviones(i).TaskGlobal(j, k).ConstraintType = pjASAP
  Next
 Else
   aviones(i).TaskGlobal(j, 0).ConstraintType = pjASAP
 End If
End If
If j > EC Then
   If aviones(i).Estaciones(j) = 0 Then
   aviones(i).TaskGlobal(j, 0).ConstraintType = pjASAP
   Else
   For k = 1 To aviones(i).Estaciones(j)
     aviones(i).TaskGlobal(j, k).ConstraintType = pjASAP
   Next
   End If
End If
Next
```

```
End Sub
```

CalcularCritica

DESCRIPCIÓN: calculamos la estación- subestación crítica de un avión teniendo en cuenta el número de posiciones.

LLAMADAS ENTRANTES:

Vinculacion.AsignarVinculacion

Vinculacion.PegarAviones

CÓDIGO:

```
Sub CalcularCritica(av As Avion, x As Integer, y As Integer)
 Dim Est As Integer
 Dim Subest As Integer
 Dim DuracionMax As Single
 Dim Estacion As Variant
 Dim Duracion As Variant
 Dim Posicion As Variant
 DuracionMax = 0
 Estacion = av.Estaciones
 Duracion = av.Duraciones
 Posicion = av.Posiciones
 For Est = 0 To av.NumeroEstaciones - 1
   For Subest = 0 To Estacion(Est)
     If Posicion(Est, Subest) <> 0 Then
      If DuracionMax < (Duracion(Est, Subest) / Posicion(Est, Subest)) Then
         DuracionMax = (Duracion(Est, Subest) / Posicion(Est, Subest))
         x = Est
         y = Subest
      End If
     End If
   Next
 Next
End Sub
```

DelimitarFecha

DESCRIPCIÓN: forzamos la fijación del comienzo o fin de la escalera a una fecha concreta.

LLAMADAS ENTRANTES:

Vinculacion.AsignarVinculacion

LLAMADAS SALIENTES:

CÓDIGO:

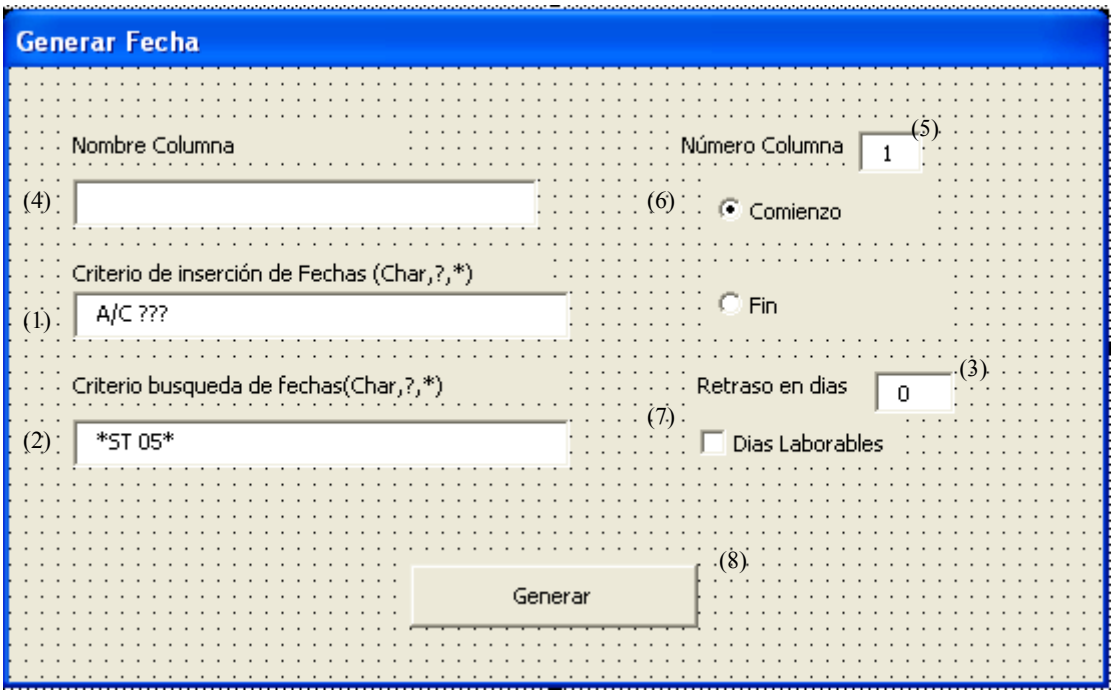
```
Sub DelimitarFecha()
```

```
Dim NE As Integer
Dim nsu As Integer
 NE = aviones(planning.NumeroAviones).NumeroEstaciones
 nsu = aviones(planning.NumeroAviones).Estaciones(NE - 1)
 If FrmOpciones.CPA Then
  SetTaskField Field:="fecha de delimitación", Value:=Inicializacion.FFecha,
TaskId:=aviones(1).TaskAvion.Id
 End If
 If FrmOpciones.CUA Then
  SetTaskField Field:="fin", Value:=Inicializacion.FFecha,
TaskId:=aviones(planning.NumeroAviones).TaskGlobal(NE - 1, nsu).Id
 End If
End Sub
```

ANEXO III

CALCULO DE FECHAS

FORMULARIO



(1),(2),(3),(4),(5): TextBox1, TextBox2, TextBox3, TextBox4, TextBox5 (TextBox)

(6): OptionButton1 (OptionButton)

(7): CheckBox1 (CheckBox)

(8): Generar (CommandButton)

MÉTODO

Generar_Click

DESCRIPCIÓN: gestiona posibles errores y llama al método principal.

```
Private Sub Generar_Click()
On Error GoTo err
If (TextBox1.Text = "" Or TextBox2.Text = "" Or TextBox4.Text = "") Then
```

```
  MsgBox "Debe rellenar todos los campos!!", vbOKOnly
  Exit Sub
End If
GeneraFecha.CreaColumna
Exit Sub
err:
   MsgBox " Incompatibilidad interna con los datos introducidos", vbOKOnly
End Sub
```

MÓDULOS

MacroColumnaFecha

DESCRIPCIÓN: método que hace accesible la macro. Es el que hay que ejecutar para "arrancar" la aplicación.

CÓDIGO:

```
Sub MacroColumnaFecha()
FrmGeneraFecha.Show
End Sub
```

GeneraFecha

MÉTODOS

CreaColumna

DESCRIPCIÓN: creamos la nueva columna (en la posición número columna indicada por el usuario) y realizamos la busqueda de fecha con el cirterio de busqueda especificado, la inserción se hace en aquellas tareas que cumplan con el criterio de inserción. Para el cálculo de retraso en días se llama a otro método.

LLAMADA SALIENTE:

obtenerRetraso

CÓDIGO:

```
Sub CreaColumna()
Dim objTarea As Task
Dim objInsercion() As Task
Dim objReferencias() As Task
Dim i As Integer
Dim j As Integer
Dim k As Integer
```

```
Dim patron As String
Dim insercion As String
Dim nombreColumna As String
Dim retraso As Integer
Dim NC As Integer
Dim Snc As String
Dim Comienzo As Boolean

nombreColumna = FrmGeneraFecha.TextBox4.Text
insercion = FrmGeneraFecha.TextBox1.Text
patron = FrmGeneraFecha.TextBox2.Text
Comienzo = FrmGeneraFecha.OptionButton1.Value
NC = FrmGeneraFecha.TextBox5.Text
Snc = "Fecha" & NC
retraso = FrmGeneraFecha.TextBox3.Text

TableEdit Name:="&Entrada", TaskTable:=True, NewName:="", FieldName:="", NewFieldName:=Snc,
Title:=nombreColumna, Width:=10, Align:=pjCenter, ShowInMenu:=True, LockFirstColumn:=True,
DateFormat:=255, RowHeight:=1, ColumnPosition:=-1, AlignTitle:=1
TableApply Name:="&Entrada"
SelectTaskColumn Column:=Snc
EditClear

i = 0
j = 0
ReDim objInsercion(i)
ReDim objReferencias(j)

For Each objTarea In ActiveProject.Tasks
 If (objTarea.Name Like insercion) Then
   Set objInsercion(i) = objTarea
   i = i + 1
   ReDim Preserve objInsercion(i)
 End If
 If (objTarea.Name Like patron) Then
    Set objReferencias(j) = objTarea
   j = j + 1
   ReDim Preserve objReferencias(j)
   End If
 Next
```

```
 If i > j Then
   MsgBox " Advertencia: El número de Filas de inserción de fechas es mayor que el de origen",
vbOKOnly
   ReDim Preserve objInsercion(j)
  i = j
  End If
 If i < j Then
   MsgBox " Advertencia: El número de Filas de obtención de fechas es mayor que el de inserción",
vbOKOnly
   ReDim Preserve objReferencias(i)
   j = i
  End If
For k = 0 To i - 1
 Select Case NC
   Case Is = 1
    GoTo d1
   Case Is = 2
    GoTo d2
   Case Is = 3
    GoTo d3
   Case Is = 4
    GoTo d4
   Case Is = 5
    GoTo d1
   Case Is = 6
    GoTo d6
   Case Is = 7
     GoTo d7
   Case Is = 8
     GoTo d8
   Case Is = 9
     GoTo d9
   Case Is = 10
     GoTo d10
End Select
finbucle: Next
 Exit Sub
d1:
    If Comienzo Then
     objInsercion(k).Date1 = obtenerRetraso(objReferencias(k).Start, retraso)
```

```
  Else
   objInsercion(k).Date1 = obtenerRetraso(objReferencias(k).Finish, retraso)
    End If
 GoTo finbucle
d2:   If Comienzo Then
   objInsercion(k).Date2 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date2 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
d3:   If Comienzo Then
   objInsercion(k).Date3 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date3 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
d4:   If Comienzo Then
   objInsercion(k).Date4 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date4 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
d5:   If Comienzo Then
   objInsercion(k).Date5 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date5 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
d6:   If Comienzo Then
   objInsercion(k).Date6 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date6 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
d7:   If Comienzo Then
   objInsercion(k).Date7 = obtenerRetraso(objReferencias(k).Start, retraso)
  Else
   objInsercion(k).Date7 = obtenerRetraso(objReferencias(k).Finish, retraso)
   End If
GoTo finbucle
```

```
d8:    If Comienzo Then
    objInsercion(k).Date8 = obtenerRetraso(objReferencias(k).Start, retraso)
   Else
    objInsercion(k).Date8 = obtenerRetraso(objReferencias(k).Finish, retraso)
    End If
GoTo finbucle
d9:    If Comienzo Then
     objInsercion(k).Date9 = obtenerRetraso(objReferencias(k).Start, retraso)
   Else
    objInsercion(k).Date9 = obtenerRetraso(objReferencias(k).Finish, retraso)
     End If
 GoTo finbucle
d10:    If Comienzo Then
    objInsercion(k).Date10 = obtenerRetraso(objReferencias(k).Start, retraso)
   Else
    objInsercion(k).Date10 = obtenerRetraso(objReferencias(k).Finish, retraso)
    End If
GoTo finbucle
End Sub
```

obtenerRetraso

DESCRIPCIÓN: método que calcula la fecha en función de los días de adelanto o retrado especificados y con la posibilidad de realizarlo o no teniendo en cuenta los días marcados como festivos en el calendario laboral de Project.

CÓDIGO:

```
Function obtenerRetraso(f As Date, r As Integer) As Date
Dim d As Period
Dim rr As Integer
Set d = ActiveProject.Calendar.Period(f)
rr = r
If FrmGeneraFecha.CheckBox1 Then
If rr < 0 Then
While rr < 0
   f = f + 1
   Set d = ActiveProject.Calendar.Period(f)
  If d.Working Then
   rr = rr + 1
  End If
Wend
 obtenerRetraso = f
```

```
 Exit Function
End If
If rr = 0 Then
 obtenerRetraso = f
 Exit Function
End If
While rr > 0
 f = f - 1
 Set d = ActiveProject.Calendar.Period(f)
  If d.Working Then
   rr = rr - 1
 End If
Wend
obtenerRetraso = f
Else
  f = f - r
  obtenerRetraso = f
End If
End Function
```

REFERECIAS

REFERENCIAS

- CHARTE OJEDA, Francisco: *Programación profesional con Visual Basic 4.0*. Anaya Multimedia. Madrid,1997.

- *Microsoft Project 98*. Ediciones ENI, colección Pasaporte. Barcelona, 1998.

- Página oficial de Artemis, empresa dedicada a la gestión de recursos y proyectos http://www.aisc.com/us/lang_en/solutions/project_management

- Página con información sobre los sistemas CAD y PDM http://www.cadvisionsl.com/pdm.php

- Página oficial de EADS http://www.eads.net/eads

- Página con información sobre SAPR/3 http://www.sapmania.com

- Página oficial de Tecnomatix, empresa dedicada a las soluciones MPM http://www.tecnomatix.com

- R.DUNCAN, William: *A Guide to the Project Management Body of knowledge* http://www.pmi.org/prod/groups/public/documents/info/PP_PMBOKGuide2000Excerpts.pdf

- RAMÍREZ CRUZADO: *Tecnología de fabricación aeronáutica*. Documento interno de la empresa Construcciones Aeronáuticas, S.A. Factoría San Pablo, Sevilla.