Especificación de requerimientos software.

1. Introducción.

1.1 Propósito.

El propósito es definir cuales son los requerimientos que debe tener un programa que gestione el sistema de control de un aeropuerto.

La aplicación de control aéreo ha sido encargada por la empresa NOVUELA para gestionar el control aéreo y terrestre y sustituir a los ineficientes controladores aéreos.

1.2 Ámbito.

El producto que vamos a describir es un programa que desempeñará el papel de los controladores aéreos.

Este producto debe ser capaz de adaptarse a cualquier aeropuerto, o sea, según el número de pistas, terminales, puertas, etc. El programa deberá funcionar de igual forma.

Además el programa debe ser capaz de informarnos del estado global del aeropuerto en cualquier instante de tiempo, advirtiéndonos de todos los sucesos que han ocurrido en ese día.

La precisión de este programa será de minutos, por lo que, en cada minuto deberá comprobar el estado del sistema.

1.3 Definiciones, acrónimos y abreviaturas.

Aeropuerto : conjunto de instalaciones para el transporte aéreo.
Pistas de aterrizaje y despegue: lugar por el que los aviones aterrizan y despegan en/del aeropuerto.
Terminales: Zona de un aeropuerto por donde se puede acceder a las pistas.
Puertas: Puertas situadas en las terminales.
Pistas de acceso: Unen las pistas de aterrizaje y despegue con una serie de puertas y viceversa.
Aviones: Pueden ser de cuatro tipos: de pasajeros, comerciales, aerotaxis y avionetas.
Vehículos terrestres: Dan servicio al aeropuerto, como autobuses, portamaletas, ambulancias y furgonetas.
Aterrizar: un avión toma tierra y ocupa una pista.
Despegar: un avión se eleva y libera una pista.

1.4 Referencias.

[1] ANSI/IEEE Std. 830-1984 Guía del IEEE para la Especificación de Requerimientos Software.

2. Descripción general.

2.1 Perspectiva del producto.

La aplicación AEROGESTION debe aumentar la eficacia en la gestión del control aéreo y reducir la tasa de errores que se provocan debido a los operadores humanos.

2.2 Funciones del producto.

Las funciones que debe realizar el producto la podemos clasificar en varios bloques:

A) Almacenamiento de datos.

a1) Almacenamiento de los datos del aeropuerto:

- Almacenar en memoria el número de pistas, terminales, puertas y pistas de acceso, puertas de cada pista de acceso.

- Asignar a cada pista el tiempo de acceso, que representará, el tiempo que le cuesta, a un avión cualquiera, desplazarse desde el inicio al final de la misma.

a2) Almacenamiento de los datos del día:

- Almacenar en memoria el número de aviones junto con su tipo (de pasajeros, comercial, aerotaxis y avionetas privadas) que despegan o aterrizan ese día.
- Añadir todos los datos de cada avión (hora de salida/llegada, origen/destino, número de personas de la tripulación, la capacidad máxima de queroseno (en litros) del depósito del avión, el consumo de queroseno por minuto de vuelo, el tiempo total (en minutos) del trayecto y la puerta que tienen asignada para despegar /aterrizar.

a3) Almacenamiento de los datos de los vehículos terrestres:

- Almacenar en memoria el número de vehículos existentes, junto con su tipo y su situación.

B) Situación del aeropuerto.

b1) Situación de los aviones.

En cada momento el programa nos debe de informar acerca de la situación en que se encuentra cada avión: aparcado, esperando despegue, esperando aterrizar...

b2) Situación particular del avión.

En cada momento nos tiene que dar información sobre las condiciones de cada avión, como son la hora de despegue/aterrizaje, situación de los depósitos de queroseno, número de pasajeros, prioridad...

2.3 Características del usuario.

Este producto sólo lo utilizarán aquellas personas del aeropuerto que deseen comprobar el funcionamiento del aeropuerto. Además deberá de haber algún encargado de introducir los datos del avión, junto con los horarios de despegue y de aterrizaje.

Dado el entorno de ventanas de la aplicación sería conveniente que el usuario tuviera conocimientos de Windows a nivel usuario, además de un alto conocimiento del funcionamiento del aeropuerto.

2.4 Restricciones generales.

La aplicación se realizará con un lenguaje de programación Orientado a Objetos: el Object Pascal. Al ser una simulación en tiempo no real no necesitaremos un hardware específicamente potente para que lo mantenga en tiempo real.

3.Requerimientos específicos.

3.1 Requerimientos funcionales.

3.1.1 Inicialización de la aplicación.

3.1.1.1 Introducción.

Al principio del día la aplicación se conecta a un ordenador central del que recibe toda la información del aeropuerto.

3.1.1.2 Entradas.

En primer lugar, se introduce la estructura del aeropuerto. Esta sólo se introducirá una vez.

Cabe señalar que para cada pista de acceso y de aterrizaje y despegue hay un tiempo asignado, que representará el tiempo que a cada tipo de avión le cuesta desplazarse desde el inicio hasta el final de la pista.

Después se introducen los datos de los aviones que llegan ese día, y los que despegan, junto con la puerta de la terminal donde están estacionados y los vehículos terrestres junto con su situación.

3.1.1.3 Proceso.

El sistema almacena en su memoria toda la información para poderla consultar y actualizar a lo largo del día. Además envía toda la información a un ordenador central .

3.1.1.4 Salida.

Envío de la información al ordenador central y actualización de la consola.

3.1.2 Alta de aviones que llegan.

3.1.2.1 Introducción.

Dar de alta los aviones que llegan con toda su información asociada.

3.1.2.2 Entrada.

Tipo del avión, compañía, modelo, hora y lugar de salida, hora de llegada, numero de personas de la tripulación, capacidad máxima de queroseno, consumo de queroseno por minuto, duración del trayecto y prioridad del vuelo.

3.1.2.3 Proceso.

Almacenar toda la información y actualizar la base de datos.

3.1.2.4 Salida.

Actualización de la consola.

3.1.3 Alta de aviones que despegan.

3.1.3.1 Introducción.

Dar de alta los aviones que tienen que despegar, junto con toda su información.

3.1.3.2 Entrada.

Tipo del avión, compañía, modelo, hora salida, lugar y hora de llegada, numero de personas de la tripulación, capacidad máxima de queroseno, consumo de queroseno por minuto, duración del trayecto, prioridad de salida y lugar de estacionamiento.

3.1.3.3 Proceso.

Almacenar toda la información y actualizar la base de datos.

3.1.3.4 Salida

Actualización de la consola.

3.1.4 Alta de los vehículos terrestres.

3.1.4.1 Introducción.

Dar de alta los vehículos terrestres con su tipo y su situación.

3.1.4.2 Entrada.

Tipo del avión (autobús, portamaletas, ambulancia, furgoneta) y su identificación.

3.1.4.3 Proceso.

Almacenar la información y actualizar la base de datos.

3.1.4.4 Salida.

Actualización de la consola.

3.1.5 Asignar pista de aterrizaje, pista de acceso y puerta de embarque.

3.1.5.1 Introducción.

Asignar, para cada avión que quiere aterrizar, la pista de aterrizaje además de la pista de salida y la puerta de salida.

3.1.5.2 Entrada.

Aviso del avión que está dispuesto para aterrizar incluyendo tipo, compañía, modelo, número de pasajeros (si los hay), etc.

3.1.5.3 Proceso.

Buscar ruta de aterrizaje que esté libre (pista de aterrizaje, pista de entrada y puerta de salida) y asignársela.

Si no existe ninguna, verificar si alguna ruta va a quedar libres durante el aterrizaje del avión y asignársela.

Si no existe ninguna, ponerlo en lista de espera y tenerlo en el aire hasta que pueda aterrizar.

3.1.5.4 Salida.

Actualización de la consola (ocupación de pistas, puertas...) y actualización de la base de datos.

3.1.6 Asignar pista de despegue y pista de acceso.

3.1.6.1 Introducción.

Asignar, para cada avión que quiere despegar, la pista de despegue además de la pista de acceso al avión salida y la puerta de embarque.

3.1.6.2 Entrada.

La hora del reloj indica que el avión está dispuesto para despegar. Esto deberá de ser confirmado.

3.1.6.3 Proceso.

Buscar ruta de despegue que esté libre (pista de despegue, pista de acceso) y asignársela.

Si no existe ninguna, verificar si hay alguna ruta que vaya a quedar libres durante el despegue del avión y asignársela.

Si no existe ninguna, ponerlo en lista de espera y hacer que la hora de despegue se retrase.

3.1.6.4 Salida.

Actualización de la consola (liberación de pistas, puertas...) y actualización de la base de datos.

3.1.7 Control del trafico terrestre.

3.1.7.1 Introducción.

Controlar donde está cada vehículo y en que situación se encuentra en cada momento.

3.1.7.2 Entrada.

Leer de la base de datos la situación de cada vehículo.

3.1.7.3 Proceso.

A cada minuto verificar la situación de cada vehículo, comprobando si a cambiado, y si lo ha hecho, cambiar su estado.

Además, se cambiarán los estados de las pistas que estaban ocupando o que acaban de ocupar.

3.1.7.4 Salida.

Actualización de la consola y de la base de datos.

3.1.8 Control de combustible.

3.1.8.1 Introducción.

Controlar el combustible de cada avión.

3.1.8.2 Entrada.

Leer de la base de datos la situación de cada avión.

3.1.8.3 Proceso.

Actualizar a cada minuto el combustible de cada avión que está en marcha. Esto lo haremos comprobando la cantidad de combustible que gasta por minuto, según el tipo de avión que sea.

En el caso de que un avión se quede con menos de 100 litros de queroseno, pasará a tener la máxima prioridad de aterrizaje.

3.1.8.4 Salida.

Actualización de la consola y de la base de datos de aviones, donde actualizaremos la cantidad de combustible y si es necesario la prioridad.

3.1.9 Control de prioridades.

3.1.9.1 Introducción.

Controlar y mantener actualizadas la prioridad de cada avión.

3.1.9.2 Entrada.

Leer de la base de datos la situación de cada avión.

3.1.9.3 Proceso.

Las prioridades son asignadas por el tipo de avión según el siguiente orden de mayor a menor: pasajeros, comerciales, aerotaxi, avionetas.

Además un avión podrá pasar a tener la máxima prioridad si se queda con menos de 100 litros de ueroseno.

3.1.9.4 Salida.

Actualización de la consola y de la base de datos de los aviones.

3.1.10 Control de listas de espera.

3.1.10.1 Introducción.

Mantener información sobre los aviones que están en espera de aterrizar y de los que están en espera de despegue.

3.1.10.2 Entrada.

Control de prioridades que está en cualquiera de las dos colas.

3.1.10.3 Proceso.

Los aviones que no puedan despegar o aterrizar por estar todas las rutas ocupadas, se incorporarán a sus respectivas listas de espera. Estas se actualizarán teniendo en cuenta las prioridades de cada avión.

3.1.10.4 Salida.

Actualizar listas de espera y pasar próximo aterrizaje o despeje a los controles de asignación de pistas.

3.1.11 Control del reloj

3.1.11.1 Introducción.

Aumentar el contador del reloj en intervalos de un minuto y actualizar la situación de todo el aeropuerto.

3.1.11.2 Entrada.

Información de todo el aeropuerto.

3.1.11.3 Proceso.

Actualizar toda la información del aeropuerto que pueda cambiar con el tiempo.

3.1.11.4 Salida.

Actualización de la consola.

3.1.12 Minimización de victimas.

3.1.12.1 Introducción.

Tener en cuenta la cantidad de pasajeros y tripulación para que en caso de accidente haya el menor número posible de victimas.

3.1.12.2 Entrada.

Base de datos de aviones.

3.1.12.3 Proceso.

Entre dos aviones con la misma prioridad para aterrizar se dará preferencia al que tenga mayor cantidad de pasajeros + tripulación.

3.1.12.4 Salida.

Actualización de la consola y de la base de datos aviones.

3.1.13 Minimización de esperas.

3.1.13.1 Introducción.

Debemos de controlar el tiempo de espera de los pasajeros de cada vuelo para intentarlo minimizar al máximo.

3.1.13.2 Entrada.

Base de datos de aviones.

3.1.13.3 Proceso.

Entre dos aviones con la misma prioridad para despegar, se dará preferencia al vuelo que lleve un mayor retraso. Mientras que con la misma prioridad para aterrizar, se dará prioridad al avión que lleve un mayor retraso en tanto por cien, o sea:
(Retraso * 100)/tiempo del vuelo

3.1.13.4 Salida

Actualización de la consola y de la base de datos.

3.2 Requerimientos de interfaces externos

3.2.1 Interfaces de usuario

La aplicación se visualizará por medio de una pantalla tipo consola donde el usuario tendrá a su disposición menús para obtener información de todo el sistema.
Además, al ser una simulación tendrá una opción donde poder cambiar la hora del reloj, de manera que si adelantamos el reloj 1 hora nos diga cual es el estado en que ha quedado el aeropuerto.

3.2.2 Interfaces hardware

Por determinar.

3.2.3 Interfaces software

La aplicación funcionará bajo un entorno Windows 95/98.

3.2.4 Interfaces de comunicaciones

Por determinar.

3.3 Requerimientos de eficiencia

Al ser una versión de simulación en tiempo no real no se establecerá ningún requerimiento de eficiencia y como he señalado antes, la posibilidad de sacar conclusiones al adelantar el reloj.

3.4 Restricciones de diseño

3.4.1 Estándares cumplidos

A determinar.

3.4.2 Limitaciones hardware

A determinar.

3.5 Atributos

3.5.1 Seguridad

A determinar.

3.5.2 Mantenimiento

A determinar.

3.6 Otros requerimientos

3.6.1 Bases de datos

En la base de datos s e mantendrá información de todo el aeropuerto: red de pistas, terminales, puertas, aviones, vehículos...

3.6.2 Operaciones

Al iniciar la aplicación por primera vez habrá que dar de alta el aeropuerto y todos sus componentes, para después poder trabajar con estos datos.

3.6.3 Requerimientos de adaptación a situaciones

A determinar.