Resum

Aquest Treball de Fi de Grau (TFG) ha consistit en l'elaboració d'una plataforma capaç de simular el tràfic aeri al voltant d'un aeroport.

La plataforma permet mostrar d'una forma senzilla i clara com s'estructura tot l'espai d'un aeroport aconseguir un bon control de tot el trànsit aeri. Per tant, d'una banda, mostra en quines zones es divideix l'aeroport i el seu espai aeri. Per una altra banda, aquesta plataforma serveix per mostrar quins són els diferents estats pels quals passa un avió quan fa un trajecte d'un aeroport a un altre.

Abans de començar amb la programació d'aquesta plataforma, s'ha realitzat un estudi de la gestió del tràfic aeri. Per tant, s'ha buscat informació de com s'organitza l'espai d'un aeroport, tant el terrestre com l'aeri. També, s'ha buscat informació de quines són les fases en que es composa un vol d'un avió comercial de passatgers, des del lloc d'aparcament d'un aeroport fins al lloc d'estacionament de l'aeroport de destí.

Un cop s'ha disposat de la informació necessària per entendre bé tot el funcionament del tràfic d'avions, s'ha extret les parts necessàries per complir amb els objectius i s'ha esquematitzat. D'aquesta manera s'ha pogut simplificar la informació, per tal de transformar-la en un programa. Per realitzar aquest programa s'ha utilitzat el llenguatge de programació Python, ja que permet realitzar una programació orientada a objectes, que és just el que s'ha necessitat.

Un cop finalitzat el treball es disposa d'una plataforma que compleix amb els objectius de mostrar d'una forma clara i senzilla la distribució de l'espai d'un aeroport, tant el terrestre com l'aeri, i les diferents fases d'un vol.

Sumari

ETSEIB

1. Glossari

ATS: Air Traffic Service

OACI: Organització d'Aviació Civil Internacional

FIR: Flight Information Region

UIR: Upper Information Region

CTA: Controlled Traffic Area (Àrea de Control)

TMA: Terminal Manoeuvring Area (Àrea de Control Terminal)

AWY: Airway (Aerovies)

CTR: Controlled Traffic Region (Àrea de Control)

ATZ: Aerodrome Traffic Zone (Zona de tràfic de l'aeròdrom)

IFR: Instrumental Flight Rules

VFR: Visual Flight Rules

2. Prefaci

2.1. Origen del projecte

L'origen d'aquest treball final de grau es troba a la borsa de treballs proporcionada per la universitat.

2.2. Motivació

La motivació per realitzar aquest treball recau en la curiositat de poder entendre com es gestiona una cosa tant complexa com el tràfic aeri. A més, poder materialitzar aquests coneixements realitzant un programa informàtic, el qual permet una certa llibertat de creació.

2.3. Requeriments previs

El requeriment previ més important a tenir en compte per poder realitzar aquest treball és tenir coneixements sobre el llenguatge de programació Python.

3. Introducció

3.1. Objectius del projecte

L'objectiu d'aquest treball final de grau és el de realitzar una simulació del tràfic aeri d'un aeroport. Per tal de poder mostrar, d'una manera senzilla i entenedora, a quina de les zones i en quin estat estan cadascun dels avions que estan dins el volum sota responsabilitat del Centre de Control.

3.2. Abast del projecte

Aquest treball es basa únicament en la implementació (en llenguatge Python) d'un programa capaç de simular el tràfic aeri d'un aeroport. En el qual s'esquematitzen les zones més rellevants dels aeroport i el seu espai aeri, únicament pel que fa al tràfic d'avions. Aquesta simulació es planteja amb una estructura de classes, les quals inclouen una sèrie de llistes i cues per les quals es realitza el pas dels avions entre elles. El programa no mostra una imatge exacta d'un aeroport, sinó una esquematització senzilla, però alhora prou detallada, per poder entendre correctament els diferents estats d'un avió passant per les diferents zones existents en el recorregut d'un aeroport a un altre.

3.3. Metodologia emprada

Per tal d'aconseguir que aquesta simulació sigui senzilla però el més detallada possible s'ha seguit un procediment, sense el qual no s'haurien assolit els objectius:

1) Realitzar un estudi de les zones dels aeroports i les divisions de l'espai aeri que l'envolta.

2) Entendre el procediments i la normativa que els avions segueixen per passar d'una zona a una altra durant els seus trajectes.

3) De tota la informació aconseguida en els punt 1 i 2, extreure'n la part essencial per fer una correcte esquematització.

4) Traslladar aquesta esquematització al llenguatge Python.

5) Programar per fases. Primer una programa simple i anar millorant-lo a mesura que tot va funcionant com s'espera.

4. Descripció dels elements involucrats en el trànsit aeri

Com s'ha explicat a la introducció no es pot començar programant directament, abans de tot cal entendre el funcionament d'un aeroport real, per això es farà una breu descripció de les diferents zones i estats per les que passa un avió en el seu recorregut. Per així poder esquematitzar-ho i traslladar-ho al llenguatge Python amb certa facilitat, però alhora prou detalladament per apropar-nos el màxim possible a la realitat.

Per tant, no es farà una descripció al mil·límetre de cada zona, sinó únicament una explicació dels trets més característics i útils per complir el propòsit de la simulació. Alhora que es descartaran aquelles característiques que, tot hi ser importants a tenir en compte pel bon funcionament d'un aeroport real, només ens allunyarien de l'objectiu que es vol assolir, afegint, a més, un augment innecessari de complexitat en el programa.

4.1. Zones de l'aeroport o aeròdrom

Abans de començar s'inclouran les definicions d'aeroport i d'aeròdrom:

- Un aeroport és un aeròdrom proveït d'instal·lacions i serveis permanents que assisteixen amb regularitat el trànsit aeri.[5]

- Un aeròdrom és una zona de terreny o d'aigua utilitzada per l'envol i l'aterratge o l'amarratge d'aeronaus.[5]

Vistes les definicions, es pot aclarir que la simulació plantejada en el treball és la d'un aeròdrom, independentment de si és un aeroport o no. I per això, també cal dir que durant el treball s'utilitzen els termes aeroport i aeròdrom indistintament, ja que pel que fa la simulació no hi ha cap diferencia entre ells.

Els aeroport o aeròdroms es divideixen en tres zones: la plataforma, les pistes de rodatge i la pista d'enlairament i/o aterratge.

La plataforma és aquella "àrea definida, en un aeròdrom terrestre, destinada a donar cabuda a les aeronaus pels fins d'embarcament o de desembarcament de passatgers, correu o carga, proveïment de combustible, estacionament o manteniment".[1]

Des de la plataforma mitjançant les pistes de rodatge els avions poden circular fins arribar a la pista d'enlairament i/o aterratge, des de la qual podran realitzar l'enlairament per

començar la seva ruta cap a un altre aeroport. La pista es defineix com “àrea rectangular definida en un aeròdrom terrestre preparada per l’aterratge i l’enlairament de les aeronaus”.[1]

Les pistes de rodatge són ”vies definides en un aeròdrom terrestre, establertes pel rodatge d’aeronaus i destinades a proporcionar enllaç entre una i una altra part de l’aeròdrom”. Existeixen les pistes de rodatge que es troben a la plataforma, les quals donen accés als llocs d’estacionament i proporcionen una via de rodatge a través de la plataforma; i les que comuniquen la plataforma amb la pista d’enlairament i/o aterratge, les quals proporcionen una bona via de sortida de la pista aconseguint que aquesta estigui el menor temps possible ocupada.[1]

Aquesta circulació de la plataforma a la pista per enlairar-se; de la pista a la plataforma per estacionar; o a través de la plataforma, s’anomena rodatge. El qual és un “moviment autopropulsat d’una aeronau sobre la superfície d’un aeròdrom, excloent-hi l’aterratge i l’enlairament.”[1]

Quan un avió està a la plataforma preparat pel rodatge, haurà de demanar autorització per poder fer-ho i esperar-se estacionat abans no li donin. A més, durant el rodatge l’avió es pot trobar amb altres punts d’espera, en els que s’haurà d’aturar i mantenir-se a l’espera, a menys que la torre de control de l’aeròdrom li autoritzi una altra cosa. Es poden trobar punts d’espera a les entrades de la pista que les aeronaus no podran sobrepassar fins que la pista quedi lliure i la torre de control els hi autoritzi el pas. També es poden trobar punts d’espera intermedis, els quals serveixen per assegurar una distancia de seguretat optima amb la resta d’aeronaus que estiguin circulant en aquell moment. Bàsicament els punts d’espera s’utilitzen per regular el tràfic dels avions per l’aeròdrom i mantenir una correcte distancia de seguretat.[1]

Evidentment cada aeroport té unes característiques pròpies i aquest elements s’organitzen de manera diferent per cada un. Per exemple, l’aeroport de El Prat – Barcelona, té tres pistes, dues paral·leles i una de creuada, en canvi l’aeroport de Girona té només una pista. Per tant, a l’aeroport de El Prat s’hauran de tenir en compte més variables pel transit d’aeronaus que no pas a l’aeroport de Girona. Ara bé, la funció de cada zona dels dos aeroport és la mateixa i els estats que cada avió pot tenir en elles també. Per tant, de cara al programa tindrem més en compte les característiques d’un aeroport com el de Girona ja que és molt més senzill que no pas un aeroport com el de El Prat.

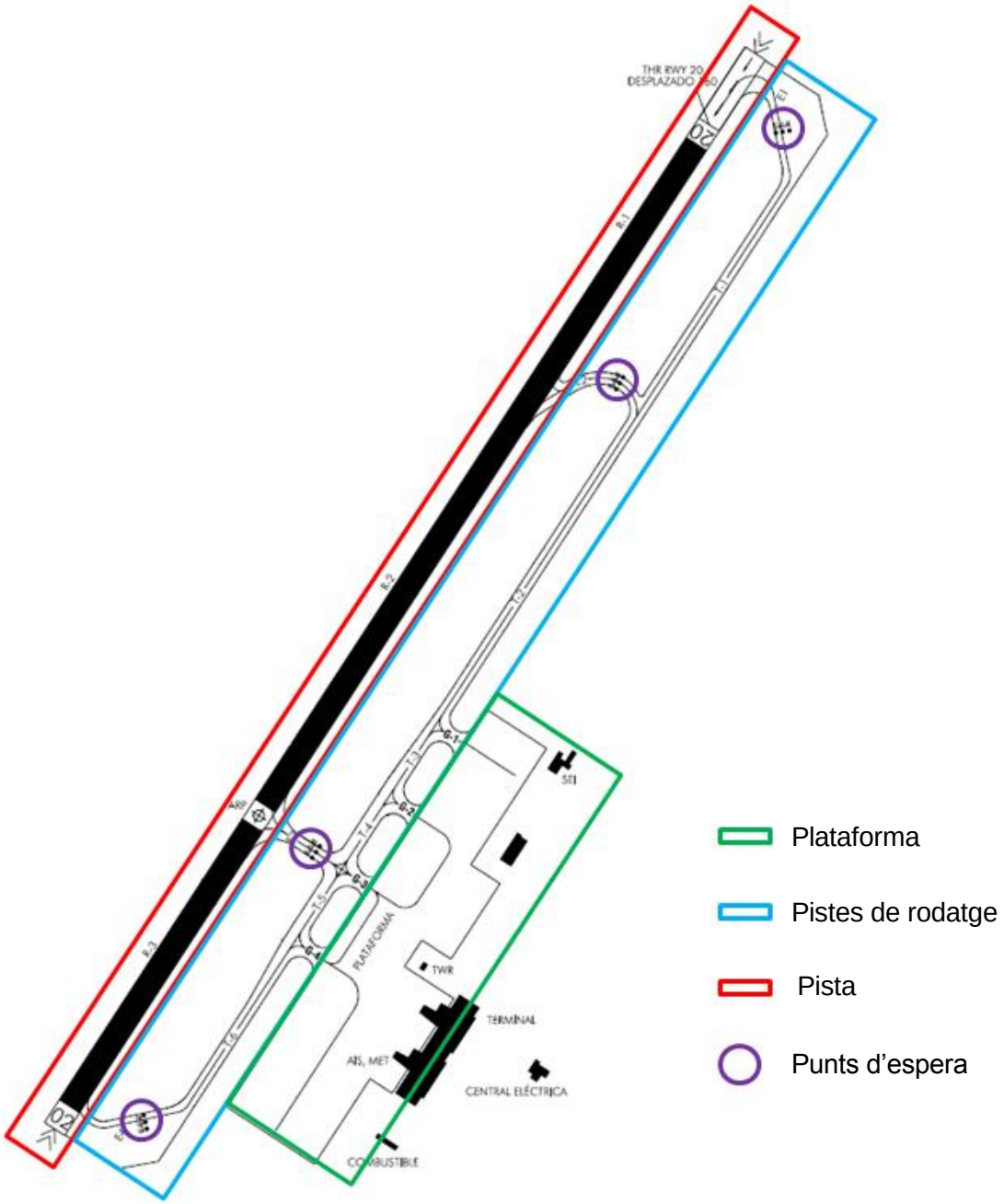

Figura 4-1. Plànol general de l'aeròdrom de Girona. [2]

Per tant de forma esquemàtica en un aeròdrom hi ha:

1) La plataforma, en la que un avió pot estar:

    a. Estacionat.

    b. Preparant-se per realitzar un vol (ja sigui desembarcant passatgers, embarcant-ne, proveint-se de combustible, etc.).

c. Esperant autorització per començar el rodatge.

d. En rodatge (ja sigui per estacionar o per dirigir-se a la pista)

2) La zona de rodatge, la qual inclou els punts d'espera, en la que un avió pot estar:

a. En rodatge

b. Esperant autorització (per seguir el rodatge, per entrar a pista, per entrar a la plataforma a estacionar)

3) La pista, en la que un avió pot estar:

a. Enlairant-se

b. Aterrant

4.2. Zones aèries

Per l'espai aeri hi circulen una quantitat enorme d'avions, per tant hi calen unes divisions i classificacions per poder mantenir-lo controlat i, així, aconseguir un transit aeri ordenat i segur.

Per una banda, l'espai aeri ATS (Air Traffic Service) en el que es facilita el servei de transit aeri es classifica segons si és un espai controlat o no controlat. L'espai aeri controlat es classifica en les classes A, B, C, D y E; i l'espai aeri no controlat en les classes G y F. Dintre de cada espai es poden realitzar uns tipus de vol o uns altres, però aquesta classificació no afectarà al tipus de simulació que es farà, per tant no cal aprofundir-ne més.[3]

Per altre banda, l'espai aeri es divideix en regions per facilitar els servies de control de les aeronaus. Aquesta serà la divisió que es tindrà en compte per la simulació, ja que els serveis de control són els que tenen controlades les aeronaus en tot moment, les que controlen el pas dels avions d'una regió a una altra i les que els permetran aterrar o els faran esperar en alguna zona abans de poder fer-ho. Tot i que cada regió té el seu servei de control propi, els quals es comuniquen entre ells per poder gestionar bé el tràfic aeri, el que es tindrà en compte sobretot és la divisió aèria que es fa i els estats dels avions en cada regió, tractant tots els serveis com un de sol.

L'Organització d'Aviació Civil Internacional (OACI) divideix el món en 9 regions d'informació de vol o FIRs (Flight Information Region). Aquestes 9 FIRs es subdivideixen en FIRs més petites, les quals es divideixen verticalment en dues subregions: les FIR i les UIR (Upper

Information Region). Espanya es composa de 3 FIRs, les de Barcelona, Madrid i Canàries.[3]

Dins les FIRs hi trobem les àrees de control (CTA) les quals s'expandeixen des d'una altura determinada cap amunt. Dins les CTA hi circulen els vols amb condicions IFR (Instrumental Flight Rules) i per sota els vols VFR (Visual Flight Rules). Tot i que a partir d'aquest punt no es faran més diferencies en els tipus de vols, ni es tindran en compte a la simulació, ja que a grans trets els dos tipus de vols passen pels mateixos estats.[1],[3]

Hi ha diversos tipus d'àrees de control, com per exemple les TMA (Àrees de control terminal o en anglès Terminal Manoeuvring Area) y les aerovies (AWY, abreviat de l'anglès airways). Les TMA són àrees controlades que s'estableixen sobre un o varis aeroports i s'encarreguen de gestionar les aproximacions i sortides dels aeroports. De la TMA hi conflueixen les aerovies, que són àrees de control disposades en forma de corredor, les quals s'encarreguen de canalitzar el tràfic aeri entre un aeroport i un altre. Pot ser que alguna àrea de control no estigui integrada per un sistema d'aerovies, llavors podran establir un sistema de rutes per facilitar el control del trànsit aeri.[1],[3]

Per unir els aeroports amb les TMA hi ha les zones de control o CTR (Controlled Traffic Region). És un espai aeri controlat associat a un aeroport, el qual s'encarrega de protegir les trajectòries d'entrada i sortida de vols IFR i de gestionar les aeronaus en els circuits d'espera de les seves proximitats. [1], [3]

Per últim es troben les ATZ (Aerodrome Traffic Zone), les quals són un espai aeri controlat gestionat per la TWR (torre de control de l'aeròdrom) utilitzat per controlar el trànsit al voltant de l'aeroport i els vols VFR. Quan ja existeix una CTR al voltant de l'aeroport, l'ATZ s'inclou en la CTR.

De cara a la simulació no es mostraran totes les zones explicades, sinó només les essencials que ajudaran a simular el transit al voltant de l'aeroport. És a dir, es tindran en comptes les CTR, TMA i AWY. Les quals es relacionen entre elles de la manera com es mostra a la imatge següent.

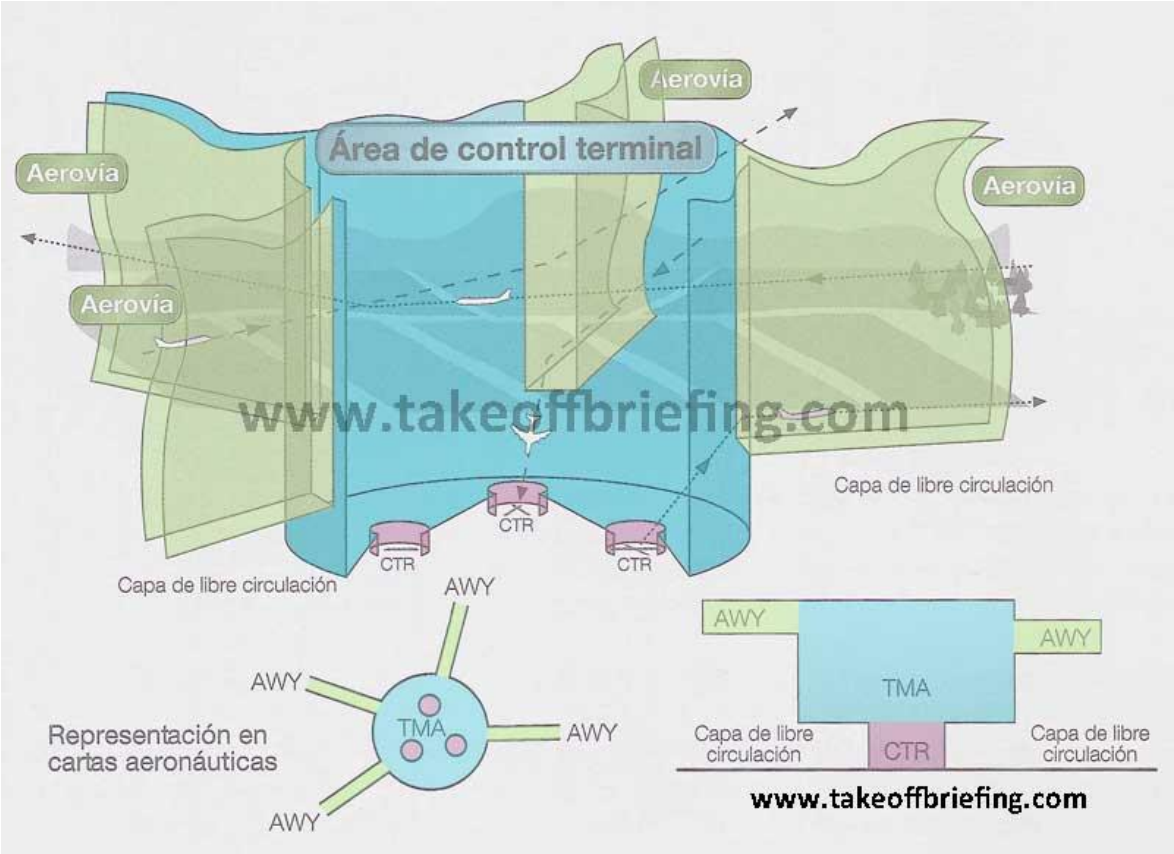

Figura 4-2. Representació de la CTR, TMA i les AWY. [4]

En resum, un avió que es dirigeix des d'un altre aeroport ho fa a través de les aerovies, fins que arribar a la TMA. Dins la TMA s'esperarà en un del circuits d'espera fins que li donin permís per realitzar l'aproximació final i aterrar. Un cop tingui l'autorització entrarà a la CTR, realitzarà l'aproximació final i entrarà en el circuit de transit per encarar-se correctament a la pista i fer l'aterratge. Per altra banda, un avió que se li doni permís per enlairar-se des de l'aeroport, entrarà a la CTR, ascendirà fins la TMA on es dirigirà cap a la ruta adient pel seu trajecte i finalment entrarà a la aerovia corresponent.

Per tant de forma esquemàtica les zones aèries es divideixen de la següent forma:

1) Aerovies (AWY), des d'on els avions podran:
    a. Dirigir-se cap a l'aeroport.
    b. Dirigint-se a un altre aeroport.
2) TMA, on els avions podran estar:
    a. Esperant autorització per aterrar.

b. Aproximant-se a l'aeroport.

c. Dirigint-se a l'aerovia indicada.

3) CTR, on els avions podran estar:

a. Aproximant-se a l'aeroport.

b. En el circuit de trànsit per aterrar.

c. Allunyant-se de l'aeroport.

4.3. Esquematització de les zones i estats

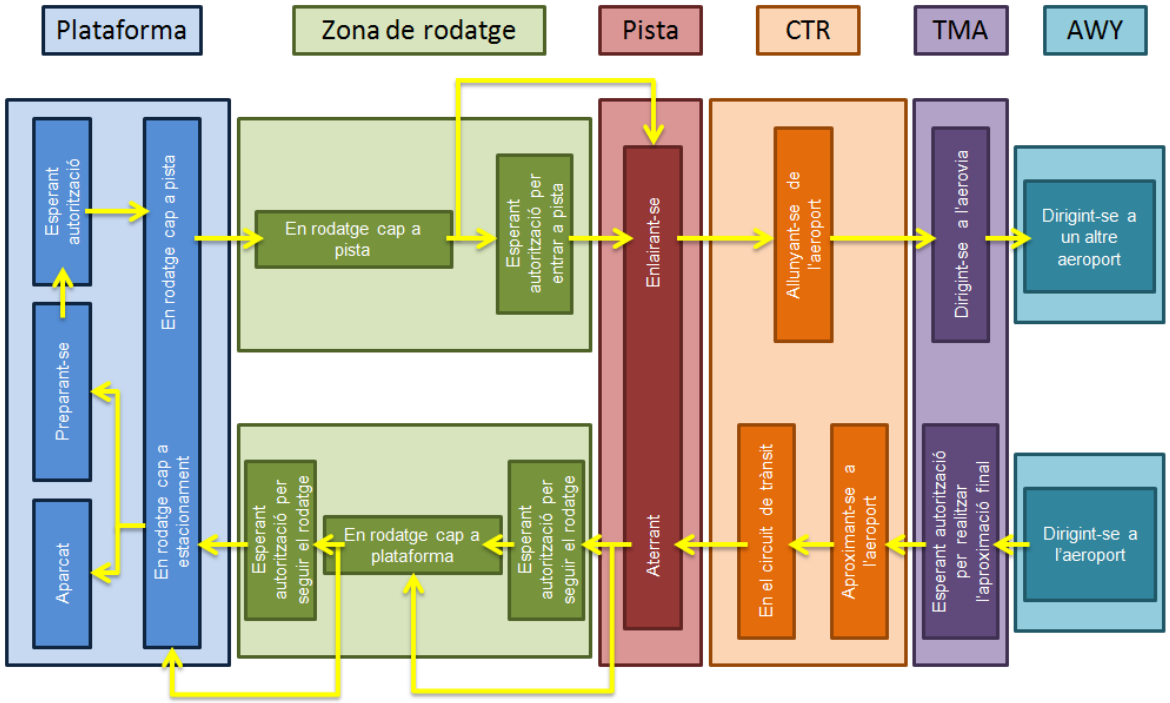

Figura 4-3. Esquema global de les zones i estats dels avions.

A la figura anterior s'esquematitza el que s'ha explicat fins ara i el que posteriorment es traslladarà al Python. Les diferents zones per les que transita un avió, els diferents estats en els que pot estar i el recorregut que fa l'avió des que s'aproxima per una aerovia fins que arriba a plataforma i torna a enlairar-se per dirigir-se a un altre aeroport.

S'ha representant un aeroport el més senzill possible. Amb una aerovia d'entrada i una de sortida, la TMA, el CTR, una sola pista d'aterratge i/o enlairament, la zona de rodatge amb

un sol camí per anar de la plataforma a la pista i un per anar de la pista a la plataforma, i la plataforma en la que només es permet que un avió estigui en rodatge. Tot això s'indica de la següent manera:

Els requadres més grans representen les diferents zones de l'aeroport. Les aerovies es representa amb dos requadres per mostrar que son dues aerovies diferents. I la zona de rodatge també es representa amb dos requadres, però per indicar que, tot hi ser la mateixa zona, es tractarà com dos blocs independents.

Els requadres petits representen els diferents estats possibles dels avions dins de cada zona. Tant a la plataforma com a la pista hi ha un requadre amb dos estats diferents. D'aquesta manera s'indica que només es permet que un avió estigui en aquella part de l'aeroport, és a dir, si un avió està en un dels estats no podrà haver-hi un altre en l'altre estat. Cal aclarir també que els estats de la plataforma "Aparcats", "Preparant-se" i "Esperant autorització", que estan en tres caselles diferents, no representen tres parts diferents de la plataforma, sinó totes tres representen l'estacionament, per tant, quan un avió passa de "Preparant-se" a "Esperant autorització" realment no s'ha mogut de lloc físic, només a canviat d'estat.

Per últim, les fletxes indiquen el camí que segueixen els avions, no només entre zones, sinó també entre els diferents estats. És pràcticament un camí recte, encara que cal destacar dues coses. Per una banda, a la zona de rodatge hi ha uns estats que l'avió pot evitar mitjançant una segona fletxa, aquest estats representen diferents punts d'espera i en els que els avions no caldrà que s'hi aturin si se'ls dona autorització prèvia. Per altra banda, entrant a la plataforma, des de l'estat de rodatge, es pot accedir a dos estats diferents. La casella "Aparcats" és per aquells avions que un cop arribin a la plataforma ja han acabat els seus trajectes i no tornaran a enlairar-se, en canvi, la casella "Preparant-se" és per aquells avions que un cop a la plataforma han de marxar cap a un altre aeroport.

4.4. Preferències del tràfic aeri

Un cop s'ha plantejat tot l'escenari per on els avions circularan, cal detallar com ho faran. En un aeroport real, són els controladors aeris el que donen autorització als avions per seguir amb el seu recorregut en les diferents fases d'aquest. Ho fan valorant en cada moment la situació dels avions respecte tot el que els envolta: la resta d'avions, altres vehicles que hi ha per l'aeroport, les condicions meteorològiques, etc. Ara bé, la simulació plantejada, com ja s'ha dit a la introducció, no és una simulació real de tot el que passa a l'aeroport, sinó que en aquesta simulació només es tindran en compte els avions que hi circularan i no la resta de vehicles que hi pugui haver en un aeroport o les diferents condicions meteorològiques que podrien afectar en el tràfic aeri.

De la mateixa manera, la simulació no contarà amb uns controladors que decideixin en el moment quins moviments poden fer els avions, sinó que contarà amb una sèrie de condicions fàcilment programables, basades en els objectius dels controladors i els criteris que utilitzen per aconseguir-los, amb les que es deixarà canviar o no els avions d'estat i de zona. En els casos reals els controladors tenen moltes normes i criteris a seguir per autoritzar els moviments dels avions, però només s'han tingut en compte aquelles que ajudin a fer circular els avions per l'esquema representat a l'apartat anterior. L'objectiu principal dels controladors és:

- Aconseguir una correcta circulació dels avions, assegurant un alt nivell de seguretat, adoptant tots els tipus de mesures per reduir el numero d'accidents i incidents. [3]

I els criteris i normes que s'han tingut en compte són els següents:

1) Una aeronau que aterra o es troba a les ultimes fases de l'aproximació per aterrar, tindrà normalment prioritat sobre una aeronau que estigui a punt de sortir. [1]

2) S'autoritza la sortida de les aeronaus en l'ordre en el que estiguin llestes per enlairar-se, però es pot seguir un ordre diferent per facilitar el major número de sortides amb el menor temps. [1]

3) Les aeronaus no podran esperar a una distancia de la pista inferior a la dels punts d'espera de la zona de rodatge. [1]

4) Es deixarà suficient separació entre les aeronaus en vol, que es trobin en el circuit de trànsit, per poder mantenir-se les separacions mínimes entre les aeronaus que arriben i surten. [1]

5) No es permetrà que inicií l'enlairament cap aeronau: [1]

    a. Fins que la aeronau que la precedeixi en l'ordre de sortida hagi creuat el final de la pista o hagi iniciat un viratge.

    b. Fins que totes les aeronaus que acabin d'aterrar hagin deixat la pista lliure.

6) No es permetrà creuar el començament de la pista, en la seva aproximació final, a cap aeronau que vagi a aterrar: [1]

    a. Fins que l'aeronau que la precedeixi en l'enlairament hagin creuat el final de la pista en us o hagin iniciat un viratge.

    b. Fins que totes les aeronaus que acabin d'aterrar hagin deixat la pista lliure.

7) L'autorització d'aterrar a una aeronau es pot concedir si es té un grau raonable de seguretat de que hi haurà la separació correcte entre aeronaus quan l'aeronau creui el llinda de la pista. [1]

A més, altres consideracions que s'han tingut en compte són:

1) Permetre que la pista s'alliberi amb facilitat

2) No fer esperar els que volen aterrar o enlairar-se durant molta estona en els punts d'espera.

3) Com s'especifica a les dades de l'aeròdrom de Girona (ja que és el que prèviament s'ha pres com exemple): a la plataforma només es permetrà el rodatge d'una única aeronau en cada moment.[2]

Totes aquestes normes, criteris o consideracions que s'utilitzen per fer transitar els avions pels aeroports i els seus voltants, són les que s'intentaran plasmar a la simulació. En els següents capítols s'explicarà com s'ha traslladat aquesta informació al Python.

4.5. Avions

Arribats a aquest apartat, s'han explicat les diferents zones d'un aeròdrom, els diferents estats d'un avió en el seu recorregut i els criteris que s'utilitzaran per la simulació del trànsit d'avions. Ara, cal explicar com es representaran o com es simplificaran els avions a la simulació.

Tot i que a la introducció s'ha explicat que la simulació és del trànsit aeri, però en concret per saber els estats del avions, fins ara s'han utilitzat tant el terme "avió" com el terme "aeronau" per fer les explicacions. Això és degut a que els aeròdroms i el control de trànsit aeri no esta pensat únicament pels avions, sinó que estan pensats per aeronaus de tot tipus que puguin estar circulant alhora. Anem a veure concretament les definicions d'aeronau i avió:

- Una aeronau és tota màquina que es pot sustentar a l'atmosfera per reaccions de l'aire que no siguin reaccions del mateix contra la superfície de la terra. [1]

- Un avió és un aerodina propulsada per motor, que deu la seva sustentació a l'aire principalment a reaccions aerodinàmiques exercides sobre superfícies que romanen fixes en determinades condicions de vol (les ales). [1]

Per entendre-ho bé, cal definir aerodina:

- Un aerodina és tota aeronau que principalment es sustenta a l'aire gràcies de forces aerodinàmiques (per exemple: avions, helicòpters).

Amb aquestes definicions, es pot dir que es correcte l'ús dels dos termes per fer les explicacions que s'han fet fins ara, però com la simulació és únicament per avions, a partir d'aquest moment es parlarà només dels avions i no de totes les aeronaus.

D'avions n'hi ha de molts tipus, models, tampanys i cadascun pot tenir funcions diferents, però per aquesta simulació es tractaran tots de la mateix manera, sense importar les diferencies que hi ha entre uns i altres.

El que ens interessarà del avions serà només allò que ens ajudi a fer la simulació desitjada. Com ja s'ha explicat anteriorment, no es pretén fer una simulació que copií exactament tot el que passa en un aeroport, sinó el que es vol aconseguir és una simulació que permeti diferenciar els estats de cada avió en cada zona de l'aeroport. Per tant les característiques pròpies de cada avio si que ajudarien a aconseguir una simulació més real, però, en el cas en el que ens trobem, només comportarien una complexitat extra en el programa. Per tant, dels avions només ens interessarà:

- El seu codi d'identificació per poder diferenciar-los correctament.
- La zona a la que es troben en cada moment.
- L'estat en que es troben en cada moment.
- El seu sentit de vol, és a dir, si és dirigeixen cap a la plataforma o cap a les aerovies.
- Si es tornaran a enlairar o no (els els avions que estan aterrant).
- L'aparcament que ocupen mentre estan a la plataforma.

5. Programa

En el capítol anterior s'ha explicat l'esquematització que s'ha fet d'un aeroport, la simplificació feta del avions, els diferents estats possibles d'aquests en cada una de les zones i algunes normes i criteris amb les que els controladors aeris fan transitar els avions per aquestes zones. En aquest capítol, s'explicarà per una banda com s'ha traslladat tota aquesta informació en un programa informàtic que permeti fer una simulació del trànsit aeri d'un aeroport.

Com s'ha comentat a la introducció, aquest programa s'ha escrit amb el llenguatge de programació Python, el qual permet fer una programació orientada a objectes que és el que s'ha fet servir. D'una manera resumida la informació explicada en el capítol anterior s'ha traslladat al programa de la següent manera:

1) Primer, s'han dissenyat un conjunt de classes per representar cada una de les zones de l'aeroport, tant les terrestres com les aèries, que s'han explicat al capítol anterior. Aquestes classes estan completament definides, per tant l'usuari no podrà decidir quin tipus d'aeroport voldrà utilitzar per les simulacions.

2) Després, s'ha preparat una classe per representar els avions que circularan per l'aeroport. L'usuari podrà decidir quants avions participaran a la simulació i en podrà determinar algunes de els característiques d'aquests.

3) I per últim, el programa consta d'un cos general, el qual és l'encarregat de la circulació dels avions per les diferents zones, però, a més, de permetre la interacció de l'usuari amb les diferents opcions del programa. Aquestes opcions van dirigides únicament a obtenir i visualitzar de la millor manera possible tots els detalls de la simulació.

Abans d'entrar en els detalls del programa cal destacar i comentar un dels aspectes més importants de la simulació, per tal de poder entendre correctament totes les explicacions posteriors que es faran sobre el programa. Aquest programa no realitza simulacions on es visualitza com els avions es van movent contínuament imitant el moviment que farien en un aeroport. Sinó que s'ha plantejat la simulació de la següent manera:

Inicialment l'usuari introduirà els avions que vulgui a la plataforma i a l'aerovia d'arribada. Un cop introduïts, els avions no aniran canviant de zona o estat automàticament cada cert temps, sinó que serà l'usuari qui decidirà, mitjançant una comanda del programa, quan actualitzar la simulació. Cada cop que l'usuari esculli la comanda "Actualitzar" els avions canviaran d'estat o zona següents sempre que les condicions els hi permeten fer-ho. Per

tant, aquesta simulació no és a temps continu, sinó que l'usuari té una imatge congelada del trànsit de l'aeroport i decideix quan actualitzar-la. Entre actualització i actualització l'usuari tindrà disponibles altres comandes per poder obtenir informació complementaria.

5.1. Classes de les diferents zones de l'aeroport

Per representar l'aeroport i el seu espai aeri s'ha utilitzat una classe per cada una de les zones. Com s'ha dit, aquestes classes estan completament definides, és a dir que el programa tindrà el mateix aeroport per totes les simulacions que es vulguin fer. Per definir aquestes classes s'han utilitzat una sèrie d'atributs, amb els que es defineix com és internament aquest aeroport, i una sèrie de mètodes que s'encarreguen de facilitar una correcta circulació dels avions.

En aquest apartat es presentaran aquestes classes i es farà una breu explicació dels seus atributs i mètodes. Abans de començar a explicar una per una totes les classes, es pot dir que totes elles tenen una estructura molt semblant i algunes característiques en comú.

Com a atributs en comú tenen:

1) Una llista per cada un dels estats que els avions poden tenir dins de la zona a la qual representa cada classe. Per aquestes llistes és per on els avions transitaran quan es posi en marxa la simulació, per tant es podrà saber en quin estat està cada avió sabent en quina de les llistes es troba. Aquests estats possibles són els que s'han esquematitzat al capítol anterior i en concret a la *Figura 4-3*.

2) No s'han determinat unes dimensions físiques de l'aeroport, perquè no aporta cap utilitat a la simulació. Però, si que s'ha determinat la capacitat màxima d'avions que cada zona pot acceptar. En algunes classes s'ha definit un màxim global, en altres s'ha definit un màxim per cada estat i en altres no s'ha definit cap capacitat màxima. En aquests últims no significa que tinguin una capacitat il·limitada d'avions, sinó que tal com s'han plantejat les condicions pel transit d'avions o bé aquella llista mai tindrà un elevat nombre d'avions encara que no s'especifiqui una capacitat màxima, o bé si en algun moment en alguna llista hi ha un nombre més elevat del que s'esperaria en un aeroport real significarà que s'han fet circular un nombre massa elevat d'avions en comparació a les dimensions de l'aeroport. Quan s'expliquin les classes una per una es concretarà de quin tipus és cada una.

I com a mètodes en comú tenen:

1) Un mètode per treure l'avió de la primera posició de cada una de les llistes. Hi ha un mètode d'aquest tipus per cada una de les llistes.

2) Un mètode per afegir un avió a l'últim posició de la llista. Hi ha un mètode d'aquest tipus per cada una de les llistes.

3) En les classes que s'han definit capacitats màximes, hi ha un o varis mètodes encarregats de determinar la diferencia d'avions entre la capacitat màxima i el nombre d'avions que hi estan circulant. Aquests mètodes ajuden a determinar si un avió pot moure's a la zona o estat següent.

A part d'aquests atributs i mètodes algunes classes en tenen alguns de propis que ajuden al bon funcionament del programa.

5.1.1. Classe Plataforma

Com a atributs aquesta classe consta de 5 llistes diferents:

1) Llista d'entrada: pels avions que entren a la plataforma des de la zona de rodatge.

2) Llista de sortida: pels avions que surten de la plataforma per anar a la pista.

3) Llista dels aparcats: pels avions que un cop arriben a la plataforma no tornaran a enlairar-se.

4) Llista de preparació: pels avions que estan fen els preparatius necessaris per poder dur a terme un vol.

5) Llista d'autorització: pels avions que ja estan preparats i estan esperant una autorització per iniciar el seu recorregut.

Té una capacitat màxima de vuit avions i una capacitat de rodatge màxima d'un avió. És a dir, com a molt entre les cinc llistes podran haver-hi vuit avions i com a molt entre les llistes d'entrada i sortida en podrà haver només un.

A més, aquesta classe consta de dos atributs propis més:

1) Temps de preparació: és un atribut que determina cada quantes actualitzacions un avió acabarà de preparar-se, és a dir, quantes actualitzacions queden perquè un avio de la llista de preparació passi a la llista d'autorització.

2) Temps d'autorització: és un atribut que determina quantes actualitzacions fa que no surt un avió de la llista d'autorització. Només incrementa el seu valor mentre hi hagi algun avió dins de la llista d'autorització, sinó es manté a zero, i un cop un avió surt de la llista torna a valor zero. Aquest atribut ajudarà a que els avions preparats no triguin massa estona a posar-se en marxa.

3) Una llista d'aparcaments, inicialment serà una llista amb vuit zeros. Aquesta llista representa els vuit llocs que te la plataforma per estacionar avions. Quan un avió estaciona a un aparcament, el zero que representa aquell aparcament passa a ser 1. I quan l'avió surt de la plataforma torna a ser 0. Aquesta llista ajudarà a una bona representació dels avions a la interfície gràfica del programa, però això s'explicarà més endavant.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a la posició final de cada llista.

2) El grup de mètodes per treure l'avió de la primera posició de cada llista.

3) Un mètode per indicar el nombre d'avions que encara poden entrar a la plataforma.

4) Un mètode per indicar si hi ha algun avió en rodatge o no dins la plataforma.

5) Un mètode per actualitzar el temps de preparació. Es va reduint fins que arriba a zero, moment en el que el primer avió de la llista està preparat per marxar i canvia a la llista d'autorització. En aquest mateix moment el temps d'espera s'actualitza per indicar quantes actualitzacions queden perquè el pròxim avió estigui preparat per marxar. El valor de preparació de cada avió no és sempre el mateix, s'aconsegueix mitjançant un *random* que aquest valor vagi variant, d'aquesta manera es representa que els avions triguen temps diferents a preparar-se.

6) Un mètode per incrementar el temps d'autorització

7) Un mètode per posar a zero el temps d'autorització.

8) Un mètode per buscar un aparcament lliure. Retorna la posició del primer zero que es trobi a la llista d'aparcaments.

9) Un mètode per col·locar un 1 a la posició de la llista d'aparcaments que se li indiqui. Serà la posició que un avió ocuparà al entrar a la plataforma.

10) Un mètode per col·locar un 0 a la posició de la llista d'aparcaments que se li indiqui. Serà la posició que un avió deixarà lliure quan abandoni la plataforma.

5.1.2. Classe Pistes de Rodatge

Com a atributs aquesta classe consta de 5 llistes:

1) Llista d'entrada: pels avions que han aterrat i es dirigeixen a la plataforma.

2) Llista de sortida: pels avions que es dirigeixen a la pista per enlairar-se.

3) Llista d'espera 1: pels avions que s'estan dirigint a la pista per enlairar-se, però no se'ls hi ha autoritzat encara l'entrada a pista pel trànsit d'altres avions amb prioritat respecte d'ells.

4) Llista d'espera 2: pels avions que han aterrat, però no se'ls hi ha autoritzat el rodatge cap a la plataforma pel transit d'altres avions amb prioritat respecte d'ells.

5) Llista d'espera 3: pels avions que s'estan dirigint a la plataforma, però no se'ls hi ha autoritzat l'entrada per culpa del trànsit d'altres avions amb prioritat respecte d'ells.

Aquesta classe no es controla amb una capacitat global, sinó que es controla individualment cada una de les llistes. En l'aeroport que s'ha dissenyat, només s'autoritza un avió en cada una d'aquestes llistes, per tant com a màxim hi podrà haver cinc avions, en el moment de màxima ocupació.

A més, aquesta classe consta d'un atribut propi més:

1) Temps d'espera 1: és un atribut que determina el nombre d'actualitzacions que un avió s'espera en el punt d'espera 1. Aquest atribut ajudarà a que els avions no s'esperin massa estona a prop de la pista sense poder enlairar-se. Es voldrà donar prioritat als avions que aterren, però no es vol que un avió preparat per marxar s'esperi molta estona.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a la posició final de cada llista.

2) El grup de mètodes per treure l'avió de la primera posició de cada llista.

3) El grup de mètodes per indicar si cada una de les llistes està ocupada o no amb algun avió.

4) Un mètode per incrementar el temps d'espera 1.

5) Un mètode per posar a zero el temps d'espera 1.

5.1.3. Classe Pista

Com a atributs aquesta classe consta de 2 llistes:

1) Llista d'aterratge: pels avions que estan aterrant.

2) Llista d'enlairament: pels avions que s'estan enlairant per marxar a un altre aeroport.

Entre aquestes dues llistes només hi podrà haver un sol avió com a molt en cada moment.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a cada llista.

2) El grup de mètodes per treure l'avió de cada llista.

3) Un mètode per indicar si hi ha un avió a la pista o no.

5.1.4. Classe CTR

Com a atributs aqueta classe consta de 3 llistes:

1) Llista d'aproximació: pels avions que se'ls a permès realitzar l'aproximació final a l'aeroport.

2) Llista del circuit de trànsit: pels avions que estan realitzant les maniobres per encarar-se correctament a la pista per poder aterrar.

3) Llista de sortida: pels avions que s'han enlairat i estan s'estan agafant altura per allunyar-se de l'aeroport.

Aquesta classe no consta d'una capacitat màxima global, però si d'unes capacitats per cada llista. En concret, és permet un només un avió a la llista d'aproximació i un avió a la llista del circuit de trànsit. En canvi, la llista de sortida no té cap limitació fixada, però no perquè es permetí una acumulació de molts avions en aquesta zona, sinó perquè tal com s'ha fet el programa, en aquesta llista com a molt hi haurà un avió, sense necessitat de controlar-ho.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a la posició final de cada llista.

2) El grup de mètodes per treure l'avió de la primera posició de cada llista.

3) El grup de mètodes per indicar si les llistes d'aproximació i del circuit de transit estan ocupades o no amb algun avió.

5.1.5. Classe TMA

Com a atributs aquesta classe consta de 3 llistes:

1) Llista d'arribada: pels avions que arriben des d'una de les aerovies i volen aterrar a l'aeroport.

2) Llista d'espera: pels avions que estan esperant una autorització per poder aproximar-se a l'aeroport i aterrar.

3) Llista de sortida: Pels avions que venen de la CTR i es dirigeixen a una aerovia per marxar cap a un altre aeroport.

Aquesta classe no consta d'una capacitat màxima global, ni de capacitats màximes per cap de les llistes. Pel que fa a les llistes d'arribada i sortida és perquè el programa està fet de tal manera que sense necessitat de controlar-ho aquestes llistes com a molt tindran un únic avió en cada moment. I pel que fa a la llisa d'espera no te capacitat màxima per evitar que algun avió que arribi de les aerovies no pugui entrar a la TMA. Depenent del tipus de simulació que es vulgui fer, si es fan arribar més o menys avions a l'aeroport, aquesta llista acumularà més o menys avions. En un aeroport real en condicions normals mai es farà esperar a molts avions als circuits d'espera, però això és pel fet que no es faran circular més avions dels que un aeroport pot absorbir. Per tant, si en alguna simulació aquesta llista acumula massa avions, no serà perquè el programa no funcioni correctament, sinó perquè s'ha volgut simular un trànsit d'avions que no s'aproxima a la realitat d'un aeroport d'unes característiques com aquest.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a la posició final de cada llista.

2) El grup de mètodes per treure l'avió de la primera posició de cada llista.

5.1.6. Classe AWY

Com a atributs aquesta classe consta de 2 llistes:

1) Llista d'arribada: per tots aquells avions que estan venint cap a l'aeroport.

2) Llista de sortida: per tots aquells avions que marxen de l'aeroport.

Aquesta classe no té cap capacitat màxima. Així es permet fer arribar el numero d'avions que es vulgui i poden marxar tots els avions que ho hagin de fer. Un cop hagin entrar a la llista de sortida no en sortiran fins que es tanqui la simulació, ja que aquest programa no

representa l'arribada a la destinació final dels avions que marxen, simplement es mantenen a la llista indicant que estan de camí a un altre aeroport, sense importar quan hi arribaran.

A més, aquesta classe incorpora un altre atribut:

1) Temps d'arribada: per representar les actualitzacions que queden perquè el primer avió de la llista d'arribada arribi a la TMA.

I com a mètodes aquesta classe consta dels següents:

1) El grup de mètodes per afegir un avió a la posició final de cada llista.

2) El grup de mètodes per treure l'avió de la primera posició de cada llista.

3) Un mètode per actualitzar el temps d'arribada. Es va reduint fins arribar a zero, moment en el que el primer avió de la llista entra a la TMA. Un cop arriba a zero, s'actualitza per indicar quan queda per l'arribada del següent avió. El valor d'arribada dels avions no és sempre el mateix, sinó que s'obté mitjançant un *random*, d'aqueta manera s'aconsegueix representar que els avions que arriben a un aeroport no ho fan a lapsus de temps constant.

5.2. Classe Avió

Per representar els avions en el programa s'ha fet mitjançant una altra classe. Aquesta no esta completament definida com ho estaven les classes que formaven l'aeroport, sinó que l'usuari podrà introduir els avions que desitgi per cada simulació que es vulgui fer. Com podrà fer-ho s'explicarà en els següents capítols, en aquest es farà una breu explicació dels seus atributs i mètodes, de la mateixa manera com s'ha fet amb les classes que formen part de l'aeroport.

La classe avio consta dels següents atributs que el defineixen:

1) El codi: és el codi d'identificació que el distingirà de la resta d'avions. L'usuari podrà decidir quin codi tenen tots els avions que es vulguin fer circular per l'aeroport.

2) La zona actual: indica en quina de les zones descrites en els capítols anteriors es troba l'avio en cada moment.

3) L'estat actual: indica en quin estat es troba l'avió dins de la zona en la que es trobi. Els estats possibles són els descrits en els capítols anteriors.

4) La direcció: informa si l'avió es dirigeix cap a la plataforma de l'aeroport o si es

dirigeix cap a un altre aeroport.

5) Trajectes: indica quants trajectes els hi queden per fer a cada avió. Pot prendre els valors 0, 1 o 2. Tindrà un 0 quan aquest avió estigui a la llista dels aparcats de la plataforma, voldrà dir que ja no ha de tornar a enlairar-se i per tant que no li queden més trajectes per realitzar. Tindrà el valor 1 en els casos que l'avió estigui dirigint-se a la plataforma i no hagi de tornar a enlairar-se o pels avions que estiguin marxant de l'aeroport, indicant així que serà el seu últim trajecte de la simulació. I, per últim, els avions que tinguin el valor 2 seran aquells que estiguin dirigint-se a la plataforma i hagin de tornar a enlairar-se, indicant que els hi queden dos trajectes, el que estan realitzant en aquell moment, que acabarà un cop arribin a la plataforma, i el trajecte de la plataforma al seu pròxim destí.

6) Moviment: Pot tenir els valors 0 o 1. És un atribut de control que serveix per assegurar que en cada actualització els avions només podran fer un moviment, és a dir a cada actualització els avions només podran moure's a la llista següent i no saltar-se de cop dos o tres llistes.

7) Posició d'aparcament: Tindrà un valor del 1 al 8 per indicar a quin lloc de la plataforma s'estacionarà. Mentre a un avió no se li hagi assignat cap aparcament o abandoni el que està ocupant tindrà el 0 com a posició d'aparcament. Tot i que els avions passaran d'una llista a una altre sense tenir en compte la posició de l'aparcament, aquest atribut és necessari per poder representar bé el transit dels avions per la plataforma en al interfície gràfica.

8) Uns atributs imatge: aquests atributs tindran la imatge d'un avió, les quals seran necessàries per representar el tràfic aeri a la interfície gràfica.

I com a mètodes té els 4 següents:

1) Un mètode per canviar el valor de l'atribut moviment de 1 a 0 un cop l'avió s'ha desplaçat d'una llista a una altra.

2) Un mètode per canviar l'atribut moviment de 0 a 1 un cop s'ha completat l'actualització, per preparar els avions per la següent.

3) Un mètode per canviar el valor de la posició d'aparcament de 0 a al valor que se li indiqui entre 1 i 8. Aquest valor serà el que li assignarà el mètode de la plataforma que s'encarrega de buscar llocs lliures a l'aparcament.

4) Un mètode per canviar el valor de la posició d'aparcament del valor que tingui en aquell moment a 0, quan deixi el seu lloc d'aparcament.

5.3. Cos general del programa

Un cop introduïdes les classes principals del programa, cal explicar com funciona el programa, és a dir, com es fan interactuar totes aquestes classes entre elles i com l'usuari pot interactuar amb el programa.

El cos general divideix en 2 parts més:

1) La part del codi que permet interactuar l'usuari amb el programa. La qual va mostrant per pantalla (pel terminal) la informació que l'usuari necessita per poder utilitzar el programa correctament i, a més, rep les ordres que l'usuari envia.

2) Una altra classe, anomenada classe "Aeroport". La qual es l'encarregada de dur a terme totes les ordres que l'usuari envia al programa. Per tant, també és l'encarregada de fer interactuar la classe avio amb la resta de classes que representen les diferents zones de l'aeroport.

A aquestes dues parts imprescindibles del programa se li poden afegir dues més. Les quals estan integrades en les altres dues, però per una bona explicació cal explicar-les per separat:

1) La interfície gràfica del programa. Tot el programa es controla mitjançant el terminal de l'ordinador, però el programa incorpora una interfície gràfica utilitzada únicament per visualitzar d'una manera molt més clara i gràfica el moviment dels avions.

2) Codi d'errors. És una part del codi que la seva única funció és evitar possibles errors en la introducció de les ordres de l'usuari i, per tant, evitar que el programa falli i es quedi penjat sense poder solucionar-ho.

5.3.1. Interacció usuari – programa

Com ja s'ha dit, la interacció de l'usuari amb el programa es fa a través del terminal de l'ordinador. Allà aniran apareixent una sèrie de missatges i indicacions per tal de poder dur a terme la simulació correctament. Aquesta part del codi es pot dividir en tres parts diferents.

En el moment en que s'executa el programa apareix un missatge de benvinguda a l'aeroport i uns missatges de les característiques bàsiques de l'aeroport. Aquest missatge explica les capacitats màximes de cada una de les zones terrestres de l'aeroport. Aquesta informació és necessària per entendre quin tipus d'aeroport tenim i, posteriorment, entendre millor el que es veurà a la simulació. Aquests missatges són la primera part del codi

d'interacció amb l'usuari.

La segona es basa amb la inicialització de la simulació. Per poder dur a terme la simulació caldrà introduir els avions amb els que es vulgui fer-la. Per tant, aquesta segona part del codi, dona instruccions a l'usuari perquè introdueixi el nombre d'avions que vol que hi hagi a la plataforma inicialment i els seu codis. I dona instruccions perquè indiqui quants avions vol que arribin des d'una aerovia, el seu codi i si aquests avions tornaran a enlairar-se o no. En aquest punt cal anar en compte, ja que l'aeroport només té vuit places disponibles per aparcar avions, per tant, si s'indica que vuit o més avions dels que aterrin no es tornaran a enlairar, es bloquejaria la simulació. Cal destacar en aquest punt, que en un aeroport real aquesta situació no es donaria mai, ja que tots els vols estan molt controlats i per tant tots els avions que arriben a un aeroport ho fan sabent que podran aterrar. I, en el cas que hi hagués alguna situació d'emergència en el que no es permetés aterrar, s'enviarien els avions cap a un aeroport propè.

Un cop s'han introduït els avions que participaran a la simulació s'arriba a la tercera part de la interacció de l'usuari amb el programa. Aquesta part consisteix en un bucle de comandes. L'usuari té 6 comandes disponibles que podrà anar seleccionant al llarg de la simulació, fins que seleccioni la sisena comanda, la qual serveix per sortir de la simulació i tancar el programa. Al seleccionar les cinc primeres comandes, el que s'està fent internament és seleccionar un o varis mètodes de la classe "Aeroport" que s'explicarà a continuació. La sisena simplement finalitza el bucle de comandes i per tant, tanca el programa. Les 6 comandes possibles són les següents:

1) Introduir avions. Aquesta comanda serveix per poder introduir nous avions un cop iniciada la simulació. Només es permetrà afegir avions a les aerovies d'arribada i s'afegiran al final de la llista. És a dir, que es poden afegir nous avions, però afectant el menys possible a la simulació inicial.

2) Actualitzar. Aquesta és la comanda principal, la que s'encarrega de cridar els mètodes de la classe "Aeroport" que tenen com a funció actualitzar les llistes dels avions i mostrar-les tant per la terminal com per la interfície gràfica.

3) Llistes d'avions. Aquesta comanda únicament mostra les llistes dels avions tal i com estan en el moment en que es consulta, no realitza cap canvi en elles.

4) Forats lliures. Aquesta comanda s'encarrega de mostrar quants avions poden entrar a cada zona de l'aeroport (únicament les terrestres) per arribar al màxim de la seva capacitat.

5) Consultar avió. De forma general el programa mostra el moviment de tots els avions

de manera conjunta, però no aprofundeix en la informació de cada avió. Però si en algun moment es vol saber més informació sobre un avió concret, es pot seleccionar aquesta comanda. Permet entrar el codi de l'avió que es vulgui i retorna la informació més útil per entendre la seva situació actual a la simulació.

6) Tancar simulació. Com ja s'ha explicat abans, aquesta comanda finalitza el bucle de comandes i, per tant, finalitza la simulació i tanca el programa.

5.3.2. Classe Aeroport

Aquesta és la classe principal del programa, ja que és la unió entre el codi que s'acaba de descriure i el conjunt de totes les altres classes que representen les zones de l'aeroport i els avions.

Com a atributs aquesta classe té els següents:

1) Un atribut per cada una de les classes que representen les zones de l'aeroport. D'aquesta manera la classe "Aeroport" incorpora com a seves les altres classes i per tant, podrà accedir i utilitzar fàcilment a als seus atributs i els seus mètodes.

2) Una llista anomenada "llista_codis", la qual inicialment esta buida. És una llista complementaria per emmagatzemar tots els codis que s'assignin als avions que s'introdueixen. És un atribut útil per dues funcions: primer, que a l'hora d'introduir els avions, l'usuari no repeteixi el mateix codi dos cops; i la segona, per quan es vulgui consultar un avio, es pugui comprovar fàcilment si existeix l'avió que es vol consultar.

3) Una llista de totes les llistes de les classes. És un atribut que s'utilitza únicament quan l'usuari consulta un avió. Un cop es comprova que el codi de l'avió existeix es busca entre totes les llistes per trobar l'avió desitjat, i així poder accedir als atributs de l'avió.

4) Per últim, una imatge. Aqueta imatge serà el fons de la pantalla de la interfície gràfica.

I com a mètodes aqueta classe consta dels següents:

1) Un mètode per crear un avió, que anirà dirigit a la plataforma, a partir de la classe "Avio". Els avions creats a partir d'aquest mètode van dirigits exclusivament a la plataforma, ja que es creen amb l'atribut de zona actual igual a "Plataforma", i amb un sol trajecte, ja que l'únic que faran aquest avions serà marxar de l'aeroport.

2) Un mètode per crear un avió, que anirà dirigit a l'aerovia d'arribada, a partir de la classe "Avio". Els avions creats a partir d'aquest mètode van dirigits exclusivament a l'aerovia d'arribada, ja que es creen amb l'atribut de zona actual igual a "Aerovia", i segons el que se li indiqui respecte si tornarà a enlairar-se o no, assignarà un 1 o un 2 a l'atribut "trajectes".

3) Un mètode que a partir d'un codi buscar l'avió entre totes les llistes i mostra per pantalla la informació bàsica de l'avió. Mostra per pantalla exactament la següent informació:

    a. El codi de l'avió.

    b. La zona actual en la que es troba.

    c. L'estat actual de l'avió dins de la zona.

    d. L'aparcament que té assignat. Si l'avió consultat és un que està marxant de l'aeroport i ja ha deixat el seu lloc de l'aparcament o si l'avió consultat encara no té una posició assignada, el apareixerà un missatge indicant que no té cap aparcament assignat.

    e. La direcció de l'avió.

    f. Pels avions que estan a la plataforma, apareixerà un missatge indicant si tornaran a enlairar-se o no.

4) Un mètode que serveix per recorre totes les llistes que conten avions i crear unes altres llistes amb els codis d'aquests avions. D'aquesta manera, quan pel terminal es mostren les llistes amb els avions, es mostren únicament els codis dels avions i no tot l'objecte avió amb el que seria molt més complicat identificar quins avions hi ha a cada llista.

5) Un mètode que s'encarrega de moure els avions d'unes llistes a unes altres, és a dir, d'actualitzar-les. Com ho fa s'explicarà a part, en el *capítol 5.3.3. Mètode "actualitzar"*.

6) Un mètode per crear una interfície gràfica per poder mostrar els moviments que fan els avions de la manera més visual i entenedora possible. S'expliquen més detalls en el *capítol 5.3.4. Interfície gràfica.*

7) Un mètode per actualitzar la interfície gràfica amb els moviments dels avions. S'expliquen més detalls en el *capítol 5.3.4. Interfície gràfica.*

5.3.3. Mètode “actualitzar”

El mètode actualitzar és el mètode principal del programa, ja que és l’encarregat de fer els moviments dels avions d’una llista a una altra, és a dir, és el mètode que fa la funció dels controladors aeris en un aeroport real. Per tant, cal recordar quin era l’objectiu principal dels controladors:

- Aconseguir una correcta circulació dels avions, assegurant un alt nivell de seguretat, adoptant tots els tipus de mesures per reduir el numero d’accidents i incidents.

Cada cop que es cridi el mètode actualitzar es comprovarà llista per llista si hi ha algun avió dins, si és així es comprovaran una sèrie de condicions per determinar si el primer avió de la llista pot moure’s a la següent o no. Per fer les comprovacions i els moviments d’una llista a una altra, s’ajudarà dels atributs i mètodes de les altres classes.

L’ordre en el qual es comproven les llistes és el camí que està marcat a la *figura 4-3*, començant per l’aerovia d’arribada fins a la plataforma i de la plataforma a l’aerovia de sortida. S’ha agafat aquest ordre i no el contrari per tal de donar prioritat als moviments d’aterratge, ja que generalment els avions disposats per aterrar tenen prioritat sobre els avions que han d’enlairar-se. Així doncs, s’explicaran pas a pas quines condicions s’han escollit per cada un dels moviments possibles:

1) De l’aerovia d’arribada a la llista d’arribada de la TMA:

   Només es comprova si el temps d’arribada és igual a 0. Si és així, el primer avió de la llista d’arribades es mou a la llista d’arribada de la TMA. Si no és així, no hi ha cap moviment.

2) De la llista d’arribada de la TMA a la llista d’espera de la TMA:

   Si l’avió de la llista no ha realitzat cap moviment en aquesta actualització es realitza el moviment d’una llista a una altra. Sinó, no es realitza cap moviment, per evitar que el mateix avió faci dos moviments en una sola actualització.

3) De la llista d’espera de la TMA a la llista d’aproximació de la CTR:

   Es fan varies comprovacions. Primer, si el primer avió de la llista encara no s’ha mogut en aquesta actualització. Segon, si la llista d’aproximació de la CTR té algun lloc lliure. Si les dos es compleixen, aleshores es comproven dos conjunts de condicions, si algun dels dos es compleix es realitza el moviment. Els dos conjunts de condicions són els següents:

a. Si hi ha un lloc lliure a la llista de circuit de transit, si hi ha un lloc lliure a la pista, si hi ha un lloc lliure a la llista del punt d'espera 2 i si el temps d'espera 1 és més petit que 2.

b. Si no hi ha un lloc lliure a la llista del circuit de transit, però hi ha un lloc lliure a la pista, hi ha un lloc lliure a la llista del punt d'espera 2, hi ha un lloc lliure a la llista d'entrada de la zona de rodatge, hi ha un lloc lliure a la llista del punt d'espera 3 i el temps d'espera 1 és més petit que 2.

La idea del primer grup de condicions és que si l'avió té camí lliure fins el punt d'espera més proper se li permet aterrar. I, la idea del segon grup de condicions és que si no l'avió del punt d'espera de la TMA té un avió que està aterrant abans que ell, se li permetí aterrar, si s'assegura que el primer avió podrà aturar-se en un punt d'espera i el segon avió en l'altre punt d'espera. S'han escollit aquests dos grups de condicions per realitzar el moviment, ja que així es compleixen els criteris i normes, explicats al *capítol 4.4*, que els controladors aeris tenen en compte a l'hora de donar autoritzacions, en concret es compleixen els criteris 1, 4, 6 i 7.

4) De la llista d'aproximació de la CTR a la llista del circuit de transit de la CTR:

L'única condició que es comprova és que l'avió no hagi realitzat cap moviment en aqueta actualització.

5) De la llista del circuit de transit de la CTR a la llista d'aterratge de la pista:

L'única condició que es comprova és que l'avió no hagi realitzat cap moviment en aquesta actualització. És a dir, en el moment que s'autoritza a un avió de la llista d'espera de la TMA començar el moviment, ja no tindrà cap més restricció fins que hagi aterrat.

6) De la llista d'aterratge de la pista a la llista del punt d'espera 2 o a la llista d'entrada de la pista de rodatge:

Primer es comprova si l'avio de la llista d'aterratges no ha fet cap moviment. Si és així es comprova si la llista d'entrada de la pista de rodatge està lliure i si la llista del punt d'espera 3 està lliure. Si és així es mou l'avió de la pista a la llista d'entrada, però si no és així, es mou de la pista al punt d'espera 2, però el que no farà mai és quedar-se a la pista. Impedint que l'avió es quedi aturat a la pista es compleix la norma numero 3 de les que s'han explicat al *capítol 4.4*.

7) De la llista del punt d'espera 2 de la pista de rodatge a la llista d'entrada de la pista

de rodatge:

Si l'avió encara no ha realitzat cap moviment, es comprova si la llista d'entrada de la pista de rodatge esta lliure. Si ho està es comprova si la llista del punt d'espera 3 està lliure. Si ho està es realitza el moviment de l'avio. Però si no, es comprova si la zona de rodatge de la plataforma esta lliure, si la plataforma té un lloc lliure per estacionar i si el temps d'autorització és inferior a 2. Si es compleixen aquestes condicions, el moviment de l'avió també es realitza. Però si no, l'avió es manté en el punt d'espera.

8) De la llista d'entrada de la pista de rodatge a la llista del punt d'espera 3 o a la llista d'entrada de la plataforma:

   Si l'avió encara no ha realitzat cap moviment, es comprova si la zona de rodatge de la plataforma esta lliure, si la plataforma té algun lloc lliure i si el temps d'autorització és inferior a 2. Si es compleixen l'avió es mou a la llista d'entrada de la plataforma, si no es mou al punt d'espera 3.

9) De la llista del punt d'espera 3 a la llista d'entrada de la plataforma:

   Si l'avió encara no ha realitzat cap moviment, es comprova si la zona de rodatge de la plataforma esta lliure, si la plataforma té algun lloc lliure i si el temps d'autorització és inferior a 2. Si es complexen l'avió es mou a la llista d'entrada de la plataforma.

10) De la llista d'entrada de la plataforma a la llista dels aparcats o a la llista de preparació:

    Si l'avió encara no ha realitzat cap moviment, es comprova si l'avió ha de tornar a enlairar-se o no. Segons el que hagi de fer, anirà a una llista o a l'altra.

11) De la llista de preparació a la llista d'autorització:

    Si l'avió encara no ha realitzat cap moviment i el temps de preparació és igual a 0, l'avió canvia de llista.

12) De la llista d'autorització a la llista de sortida de la plataforma:

    Si l'avió encara no ha realitzat cap moviment, es comprova si la zona de rodatge de la plataforma està lliure i si abans de passar per el punt 10 també ho estava. Amb aquesta segona condició s'aconsegueix que els avions tinguin el mateix comportament a la zona de rodatge de la plataforma. Sempre que estigui ocupada, ja sigui d'entrada o sortida, a la següent actualització quedarà buida. Sense aquesta

segona condició, això no es compliria sempre.

Un cop comprovades aquestes dues condicions, es comproven tres grups de condicions més i si algun dels grups es compleix, es realitza el moviment:

a. Si la pista de rodatge de sortida i el punt d'espera 1 estan lliures.

b. Si la pista de rodatge està lliure, el punt d'espera 1 no ho està i si la pista ho està i ho ha estat abans de començar aquesta actualització. Amb aquesta segona condició s'aconsegueix el mateix resultat amb la pista, que el que tenim a la zona de rodatge de la plataforma.

c. Si independentment de com estigui el punt d'espera, la pista està ocupada, però la pista està lliure i ho ha estat abans de començar l'actualització i les llistes del circuit de transit i d'aproximació de la CTR estan lliures i la llista d'espera de la TMA està buida. Amb aquestes condicions s'aconsegueix que si no han d'aterrar avions en les pròximes actualitzacions, els avions de la plataforma puguin marxar de forma més ràpida.

13) De la llista de sortida de la plataforma a la llista de sortida de la pista de rodatge:

Únicament es comprova si l'avió no ha realitzat cap moviment en aquesta actualització.

14) De la llista de sortida de la pista de rodatge a la llista del punt d'espera 1 o a la llista d'enlairaments de la pista:

Si l'avió encara no ha realitzat cap moviment, es comprova si la pista esta lliure, si ho ha estat i si no hi ha cap avió al circuit de transit que hagi d'aterrar a la pròxima actualització. Si es compleixen, l'avió es mou a la llista d'enlairaments de la pista. Si no es compleixen, l'avió és mou a la llista del punt d'espera 1. Així, es compleix les normes 3 i 5 de les que s'han explicat al *capítol 4.4*.

15) De la llista del punt d'espera 1 a la llista d'enlairaments de la pista:

Si l'avió encara no ha realitzat cap moviment, es comprova si la pista esta lliure, si ho ha estat i si no hi ha cap avió al circuit de transit que hagi d'aterrar a la pròxima actualització. Si es compleixen, l'avió es mou a la llista d'enlairaments de la pista. Si no es compleixen, l'avió no es mou de la llista del punt espera 1. Així, es compleix les normes 3 i 5 de les que s'han explicat al *capítol 4.4*.

16) De la llista d'enlairaments de la pista a la llista de sortida de la CTR.

Únicament es comprova si l'avió no ha realitzat cap moviment en aquesta actualització.

17) De la llista de sortida de la CTR a la llista de sortida de la TMA:

Únicament es comprova si l'avió no ha realitzat cap moviment en aquesta actualització.

18) De la llista de sortida de la TMA a la llista de l'aerovia de sortida:

Únicament es comprova si l'avió no ha realitzat cap moviment en aquesta actualització.

5.3.4. Interfície gràfica

Amb tot el que s'ha explicat fins ara, el programa ja compleix els objectius del projecte. És a dir, es un programa capaç de simular el transit aeri d'un aeroport, amb la qual es mostra, d'una manera senzilla i entenedora, a quina de les zones i en quin estat estan cadascun dels avions que circulen dins el volum sota responsabilitat del Centre de Control. Ara bé, la manera com és mostra el moviment entre llistes no és la millor manera que es pot aconseguir. Amb el que s'ha explicat fins ara tota la interacció de l'usuari amb el programa es fa del la terminal de l'ordinador. El que es pretén amb aquesta interfície gràfica és poder mostrar el trànsit dels avions d'una manera molt més visual i clara. El programa seguirà funcionant a través del terminal, però a més, hi haurà una pantalla extra que ajudarà amb el seguiment dels avions.

Per aconseguir aqueta interfície gràfica hi intervenen diferents elements. D'una banda una imatge que serà el fons de la pantalla, en el qual es puguin diferenciar les diferents zones de l'aeroport, d'una manera semblant a l'esquema de la *figura 4-3,* la pantalla que s'aconsegueix és la de la *figura 5-1*. Cal destacar que en aquest fons no es representen les aerovies, d'aquesta manera s'evita la sensació de que tots els avions de les aerovies estan directament al costat de la TMA esperant per entrar, sinó que van arribant gradualment. D'una altra banda, els avions de la *figura 5*-2 que aniran circulant per aquest fons a mesura que es vagi actualitzant la simulació. I per últim, el codi necessari per aconseguir que la interfície gràfica funcioni correctament.

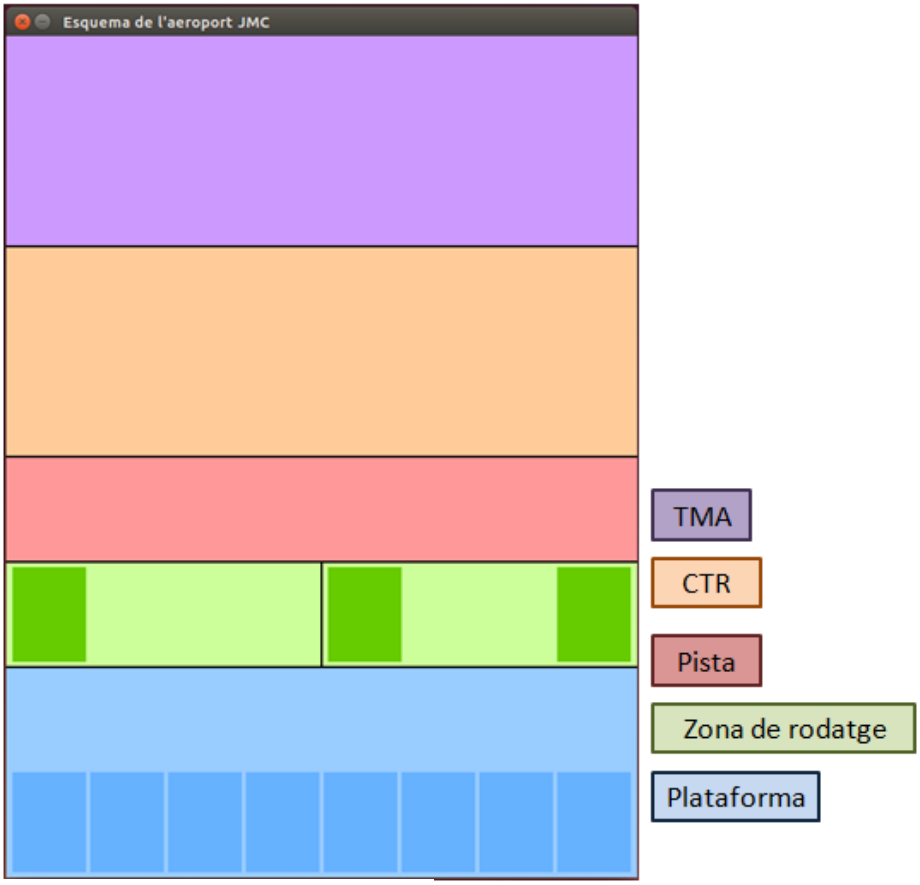

Figura 5-1. Interfície gràfica.

A la figura anterior es pot veure la interfície gràfica que es tindrà quan s'executi el programa. Es diferencien clarament les 5 zones i a més, es poden diferenciar els punts d'espera i els aparcaments amb un color més fosc, dins de les seves zones. No és necessària una interfície més complexa que aquesta per complir l'objectiu que es vol aconseguir. La imatge del fons s'ha aconseguit mitjançant comandes del Python, de manera que es té controlat perfectament les dimensions de cada una de les zones, cosa necessària per poder col·locar els avions a la posició correcta de la pantalla. A més, a l'hora de crear el fons s'ha tingut en compte la dimensió de les imatges dels avions, per tal que s'ajustin bé a cada zona i en els requadres dels punts d'espera i dels aparcaments.

A la figura següent es poden veure les imatges que s'utilitzen per representar els diferents avions. Aquestes imatges són de color negre perquè contrastin amb el fons de pantalla, però amb una franja de colors, per tal de poder diferenciar d'una manera més fàcil el seu moviment respecte els altres avions, a més, d'aquesta manera es fa notar d'alguna manera que els avions reals no són tots iguals.

Figura 5-2. Avions per la interfície gràfica.

Per aconseguir el bon funcionament de la interfície gràfica es poden diferenciar tres parts del codi que estan pensades exclusivament per això:

1) Per una banda, hi ha els mètodes 6 i 7 de la classe “Aeroport”. El primer crea una pantalla amb les dimensions de la imatge de fons. El segon mètode enganxa la imatge de fons i després comprova totes les llistes d’avions per enganxar la imatge de cada avió en el lloc que li correspongui. Cal destacar dos punts d’aquest segon mètode i, per tant, del funcionament de la interfície gràfica:

    a. El primer és que a la zona d’espera de la TMA es podran mostrar 4 avions com a molt, ja que per l’espai disponible no permet mostrar-ne més. Per tant, si es fa una simulació massa carregada d’avions, potser a la llista d’espera hi ha més avions dels que es poden mostrar a la interfície.

    b. El segon punt té a veure amb la zona d’aparcament. Amb el que s’ha explicat fins ara, les llistes que formen l’aparcament no tenen en compte quina posició d’estacionament té cada avió. En canvi, aquest segon mètode si que té en compte quin lloc d’estacionament li correspon a cada avió per enganxar-lo en el requadre de l’aparcament corresponent.

2) Per una altra banda, el codi que crida aquests dos mètodes en els moments correctes per aconseguir que els programa mostri la interfície gràfica i l’actualitzi en el moment adequat. Just després d’haver introduït els avions inicials es criden els dos mètodes per tenir una imatge inicial de la distribució dels avions. Després, només caldrà utilitzar el segon mètode cada cop que es cridi la comanda “Actualitzar”, per tant, primer s’actualitzen les llistes amb el mètode “actualitzar” i després s’actualitza la pantalla amb la nova distribució dels avions.

3) Per últim, cal destacar la part del codi que s’encarrega de seleccionar el lloc corresponent de l’aparcament per cada avió. Aquest codi s’incorpora en el codi del mètode “actualitzar”. Un cop s’han actualitzat totes les llistes s’implementa un codi per tal d’assignar un lloc d’aparcament als avions que han aterrat el més ràpidament possible i no esperar a que l’avió arribi a la plataforma. Així,s’aconsegueix el mateix comportament que en un aeroport real, en els que es comunica el lloc d’estacionament dels avions el més ràpidament possible. Com és lògic, es

s'assignarà abans un lloc d'estacionament a un avió que estigui al punt d'espera 3 (el que està just abans d'entrar a la plataforma), que a un avió que acabi d'entrar a la zona de rodatge. Per tant, es comprovarà si tenen lloc assignat i, en cas que no en tinguin, s'intentarà assignar-ne un als avions de les següents llistes i en el següent ordre: llista d'entrada de la plataforma, llista del punt d'espera 3, llista d'entrada de la pista de rodatge i llista del punt d'espera 2.

5.3.5. Codi d'errors

Amb tot el que s'ha explicat fins aquest punt, ja es té un programa que compleix l'objectiu i, a més, té una petita interfície gràfica per mostrar la simulació d'una manera molt visual. Però s'ha de tenir en compte, que al ser un programa que es controla pel terminal hi poden haver errors a l'hora d'introduir les ordres del programa. Per tant, s'ha escrit un cert codi de protecció per evitar que el programa es pengi quan hi ha alguna entrada des del terminal que el programa no podria processar.

Aquest codi de protecció s'ha introduït en els llocs del programa en que l'usuari ha de respondre a una petició del programa. D'aquesta manera el programa podrà evitar els següents errors d'entrada de l'usuari:

1) Quan s'ha d'introduir el nombre d'avions que es volen per la simulació, el programa només accepta un valor en dígits. Si no és així, el programa ho detecta i envia un missatge explicant que s'han d'entrar dígits. Fins que no és així el programa no continua endavant. Però mai es quedarà penjat.

2) Quan s'ha d'entrar el codi de l'avió, el programa no acceptarà com a codi que l'usuari hagi polsat per error la tecla d'intro i no acceptarà que el codi siguin espais en blanc. Fins que no s'introdueixi un codi el programa demanarà que es faci correctament.

3) Si per error s'introdueix el mateix codi dos cops, el programa ho detectarà i donarà un avís perquè s'introdueixi un codi diferent.

4) En el moment que s'ha de decidir si els avions que aterren o no, el programa només acceptarà com a valides les entrades: "1", "Si", "2" o "No". Mentre no sigui així, demanarà que es faci correctament.

5) En el moment d'entrar una de les comandes possibles del programa, es podrà fer de dues maneres, o entrant el numero de la comanda o escrivint el nom de la comanda. Si per error s'escull una comanda que no existeix, el programa ho detectarà i enviarà un missatge per tal que s'introdueixi correctament.

6) Per últim, quan es selecciona la comanda “Consultar avio”, i s’introdueix el codi incorrectament, apareix el següent missatge: “No hi ha cap avió amb aquest codi”. I en el cas que es vulgui provar el programa sense introduir cap avió el missatge que apareix és el següent: “No s’ha introduït cap avió”. Però en cap dels dos casos, el programa deixaria de funcionar.

Ja s’ha comentat en el *capítol 5.3.1.* que el programa no controlarà quants avions dels que aterraran no tornaran a enlairar-se. És a dir, si l’usuari decideix que vuit o més avions no tornaran a enlairar-se ho pot fer, però ha de tenir clar que estarà bloquejant la simulació i els avions que vinguin després no podran acabar el seu trajecte.

6. Exemple de simulació

Un cop s'han explicat tots els detalls importants per entendre com s'ha estructurat el programa i com funciona, es realitza un petit exemple. S'ha decidit fer un exemple de com un avió aterra i arriba al seu lloc d'estacionament a la plataforma començant des de l'aerovia d'arribada. L'exemple també serveix per mostrar com funcionen les dues comandes més importants del programa: la comanda "Actualitzar" i la comanda "Consultar avió". Són les dues més importants ja que són les que garanteixen l'objectiu del programa i del treball. "Actualitzar" garanteix un correcte moviment dels avions entre les llistes i "Consultar avió" permet consultar de cada avió la zona de l'aeroport on està i en quin estat està, justament la informació que a l'objectiu del treball s'explica que es vol aconseguir mostrar de forma clara en el programa.

Per poder començar l'exemple cal executar el programa des del terminal de l'ordinador. A la carpeta on es guarden tots els arxius del programa hi ha exactament 5 arxius ".py" i una carpeta d'imatges, en la que es troben la imatge del fons de la interfície i les imatges dels avions. En aquests 5 arxius és on s'ha escrit tot el codi del programa. Es troben els següents:

1) Classes.py: és l'arxiu on s'han programat les diferents classes que representen l'aeroport i la classe avió.

2) Aeroport.py: és l'arxiu on s'ha programat la classe aeroport.

3) Actualitzar.py: és l'arxiu on s'ha programat el mètode actualitzar de la classe aeroport.

4) Print_Llistes.py: és l'arxiu on s'ha programat el mètode encarregat de crear les llistes amb els codis dels avions.

5) Simulacio.py: és l'arxiu on s'ha programat el codi d'interacció amb l'usuari. És aquest l'arxiu que cal executar per començar la simulació.

Per realitzar l'exemple s'introduiran 3 avions, anomenats a1, a2 i a3. Un a la plataforma i dos a l'aerovia d'arribada, dels quals el primer es tornarà a enlairar i el segon no. D'aquesta manera es tindran els tres tipus d'avions possibles. Ara bé, només es consultarà la informació del segon avió, el a2, ja que és el primer avió que farà el recorregut d'aterratge.

Pas1. S'executa el programa i apareix la informació bàsica de l'aeroport.

```
moratojoan@moratojoan-Latitude-3540:~/Escritorio/PROGRAMA$ python Simulacio.py


          BENVINGUTS A L'AEROPORT JMC


 CARACTERISTIQUES BASIQUES DE L'AEROPORT JMC:

 A) La Plataforma te una capacitat d'estacionament per a 8 avions, i nomes hi
    podra haver 1 avio fent maniobres de rodatge.

 B) La Zona de Rodatge de sortida te una capacitat maxima de 2 avions, repartits
    entre la pista de rodatge de sortida i la zona d'espera a pista.

 C) La Zona de Rodatge d'entrada te una capacitat maxima de 3 avions, repartits
    entre les dues zones d'espera i la pista de rodatge

 D) Hi ha 1 Pista per aterrar i/o enlairar-se.
```

Figura 6-1. Característiques bàsiques de l'aeroport JMC.

Pas 2. S'introdueixen els 3 avions.

```
INTRODUIR ELS AVIONS PER LA SIMULACIO:

1) PLATAFORMA:
Quants avions es volen introduir a la Plataforma? 1

Avio numero 1 de la Plataforma:
Codi d'idinteficacio de l'avio: a1

2) AEROVIA D'ARRIBADA: (cal indicar si tornaran a enlairar-se o no)
Quants avions es volen fer aterrar a l'aeroport? 2

Avio d'arribada numero 1 :
Codi d'idetificacio de l'avio: a2
L'avio es torna a enlairar? (1-Si; 2-NO): 1

Avio d'arribada numero 2 :
Codi d'idetificacio de l'avio: a3
L'avio es torna a enlairar? (1-Si; 2-NO): 2
```

Figura 6-2. Introducció dels avions a la simulació.

Pas 3. Apareix la interfície gràfica inicial i es consulta l'avió a2.

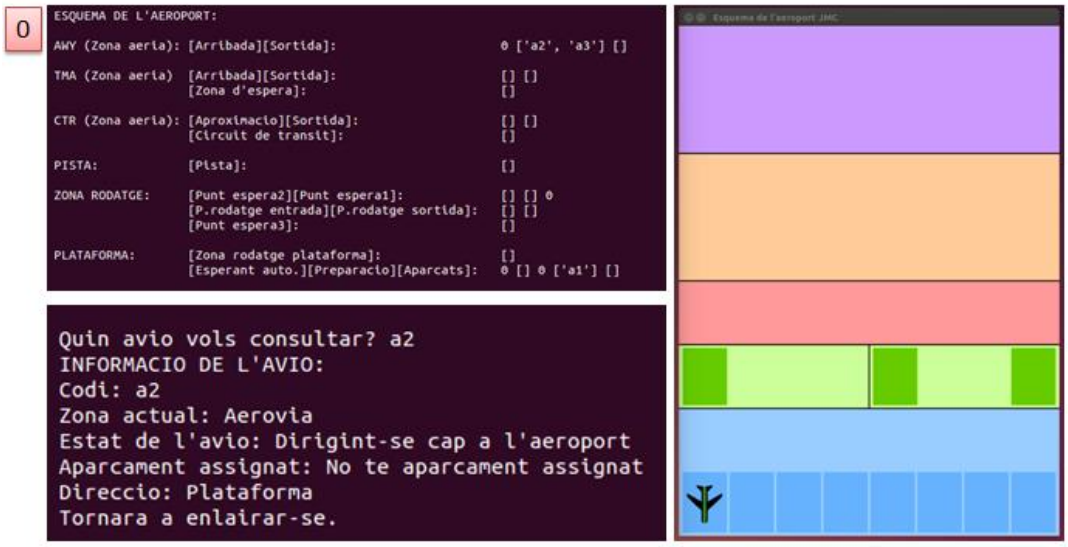

Figura 6-3. Interfície gràfica, llistes d'avions i informació de l'avió 2 inicials.

Comentaris inicials:

- El requadre de la dreta és la interfície gràfica, la qual anirà mostrant el moviment dels avions.

- El requadre de dalt a l'esquerra conté les llistes dels avions. Mostra la mateixa informació que la interfície gràfica, però ampliada amb els codis i amb els temps d'arribada, d'espera 1, d'autorització i de preparació, que són els números que hi ha al costat de les llistes corresponents. Com es pot observar els temps d'arribada i preparació estan a 0, per tant a la pròxima actualització els avions respectius es mouran de llista.

- El requadre de baix a l'esquerra conté la informació que es mostra quan es selecciona la comanda "Consultar avió".

Pas 4. Es van actualitzant les llistes i es va consultant l'avió a2 fins que aterra.

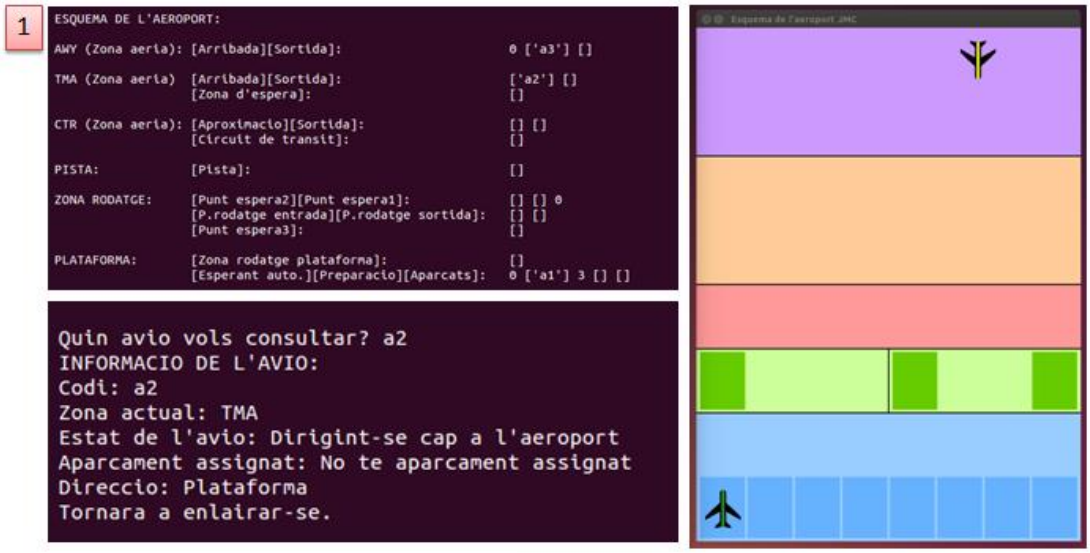

Figura 6-4. Actualització 1.

Comentaris de l'actualització 1:

- Després de la primera actualització es comprova que el temps d'arribada torna a ser zero, per tant a la següent actualització el següent avió també entrarà a la TMA.
- En canvi el temps de preparació a canviat a 3, per tant, el pròxim avió que arribi a la plataforma, trigarà tres actualitzacions a preparar-se i passar a la llista dels que esperen autorització.

Figura 6-5. Actualització 2.

Comentaris de l'actualització 2:

- Efectivament, el segon avió de l'aerovia a entrat a la TMA.

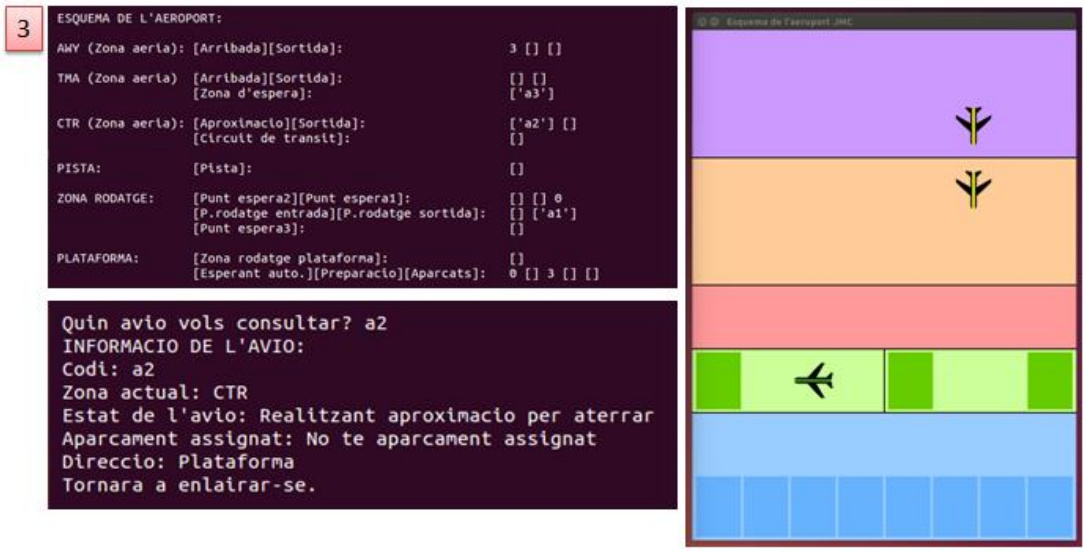

Figura 6-6. Actualització 3.

Figura 6-7. Actualització 4.

Comentaris de l'actualització 4:

- L'avió que està a la zona de rodatge, en comptes d'entrar a la pista per enlairar-se, ha entrat en el punt d'espera 1, ja que es dona prioritat als avions que estan aterrant.

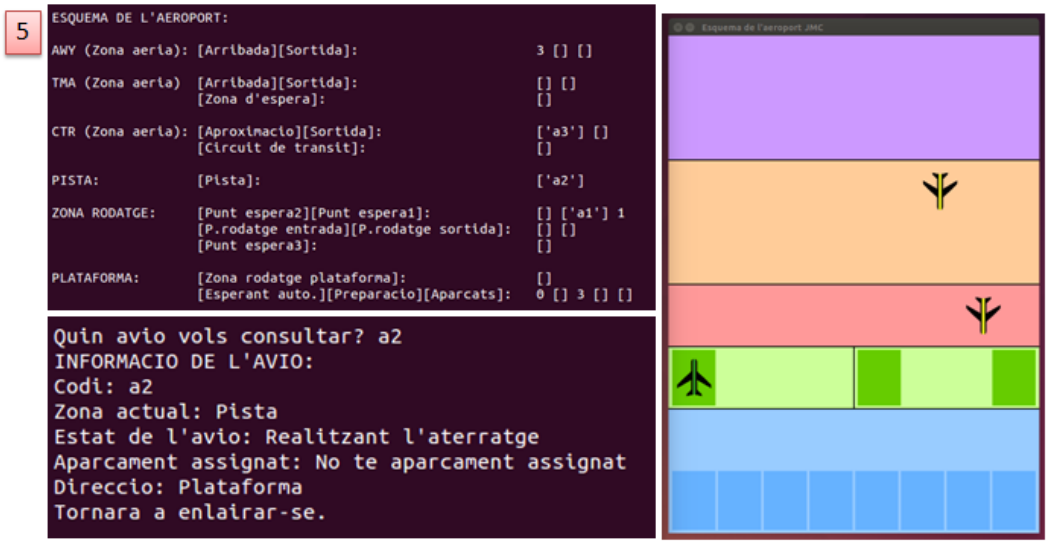

Figura 6-8. Actualització 5.

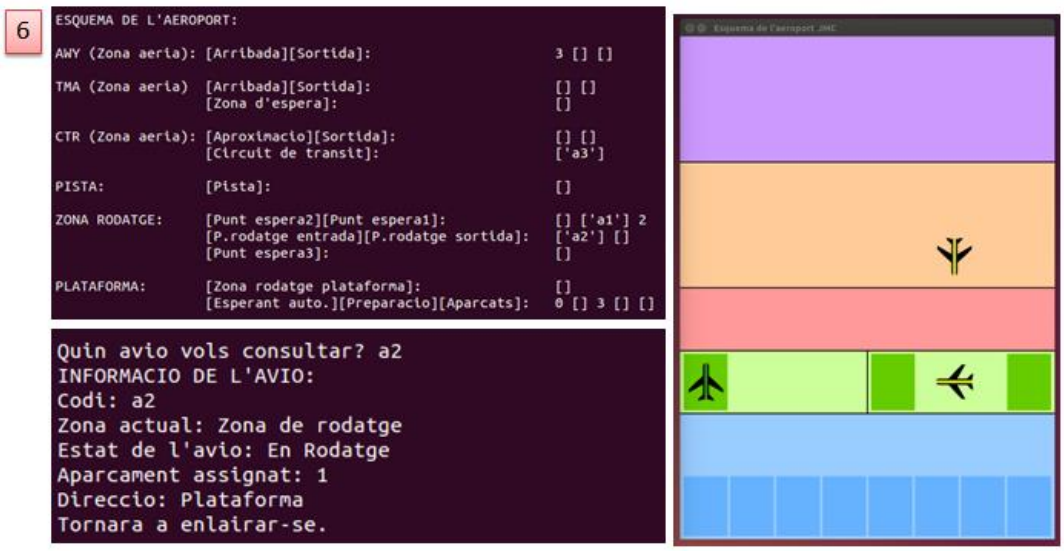

Figura 6-9. Actualització 6.

Comentaris de l'actualització 6:

- Un cop l'avió a2 surt de la pista i entra a la zona de rodatge se li assigna una lloc d'aparcament de la plataforma.

Figura 6-10. Actualització 7.

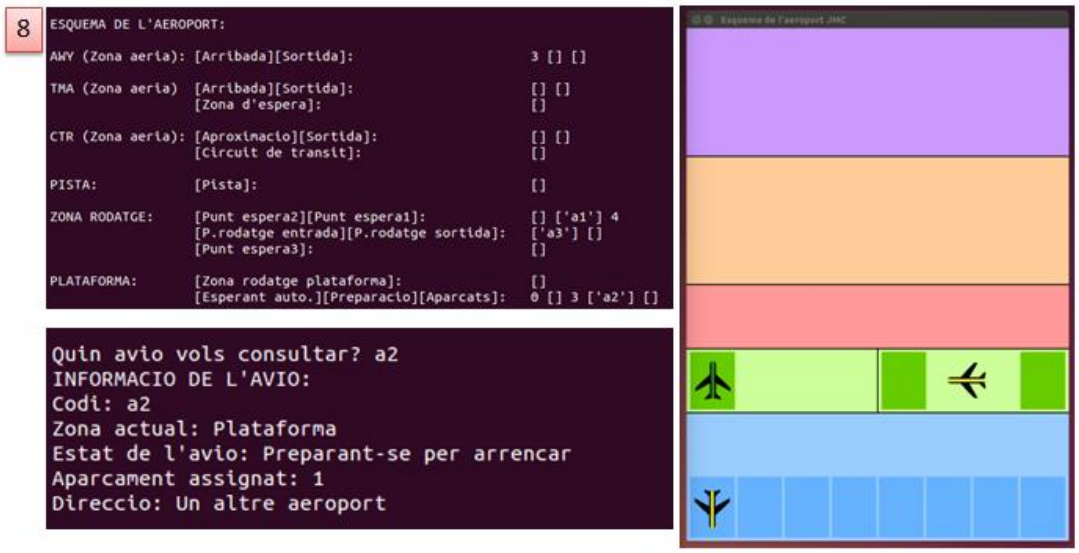

Figura 6-11. Actualització 8.

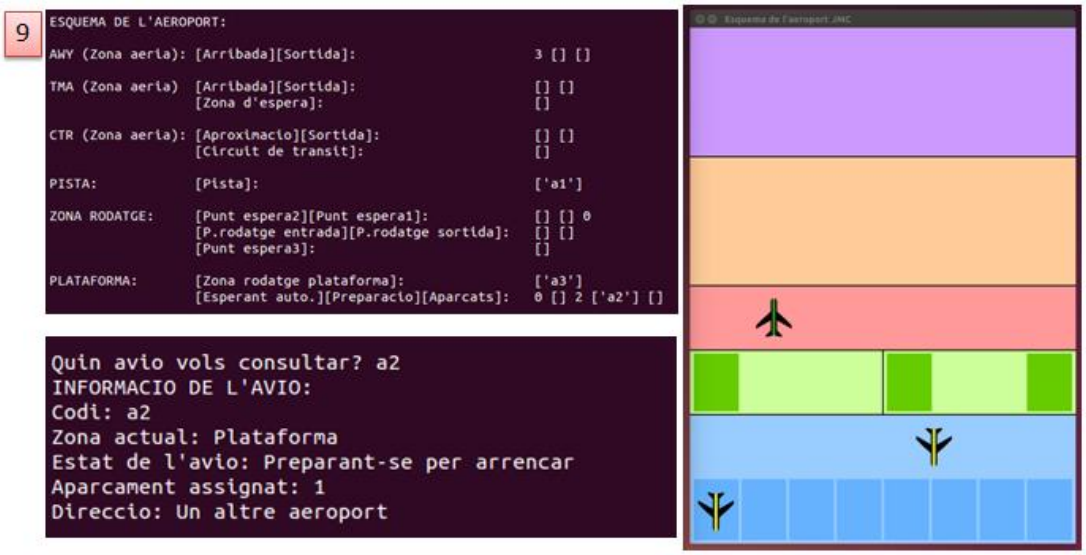

Figura 6-12. Actualització 9.

Comentaris de l'actualització 9:

- A l'actualització 8, l'avió a2 ja havia arribat al seu lloc d'estacionament de la plataforma. En aquesta nova actualització es pot comprovar com el temps de preparació a disminuït de 3 a 2. Fins que no arribi a 0 l'avió a2 no estarà llest per canviar de llista.

7. Estudi econòmic

Com qualsevol projecte aquest també té els seus costos econòmics. Els principals costos que s'han tingut en compte en aquest projecte són els següents:

1) Aquest treball ha estat realitzat abans de finalitzar el grau, així que es fixa un salari de 8€/h, corresponent al d'un becari. El treball final de grau correspon a 12 crèdits i cada un és equivalent a 30 hores de feina, és a dir són un total de 360 hores de feina. Per tant, es calcula un salari total de 360h x 8€/h = 2880€.

2) S'han dedicat unes 20 hores de supervisió per part del tutor, amb un cost de 30€/h. Per tant, 20h x 30€/h = 600€.

3) Per la realització del treball s'ha utilitzat un ordinador amb un preu aproximat a 1000€. Amb un cicle de vida d'uns 5 anys. Per tant, el cost associat de l'ordinador per realitzar el treball és de:

$$preu\_hardware = \frac{hores\_treballades}{5anys \cdot \frac{8760h}{1any}} \cdot preu\_ordinador = 8.2€ \qquad \text{(Equació 7-1)}$$

4) Per la realització del programa de simulació s'ha utilitzat el llenguatge de programació Python, el qual és d'ús gratuït. Per tant, el cost d'ús ha sigut 0€.

5) El cost d'altres despeses com l'electricitat consumida per aconseguir una bona il·luminació per treballar, el transport, el cost de la impressió de la memòria del treball, s'estima d'uns 250€.

Concepte	Cost	Hores	Quantitat associada	Cost total
Salari estudiant	8 €/h	360 h	-	2880 €
Supervisió	30€/h	20h	-	600 €
Hardware	1000€	-	3/365	8 €
Software	0€	-	-	0 €
Altres costos	250€	-	-	250 €
TOTAL				**3738 €**

Figura 7-1. Costos del treball final de grau.

8. Impacte mediambiental

L'impacte mediambiental generat en la realització d'aquest treball es pot considerar molt baix. Tot i això s'han de tenir en compte alguns aspectes, els principals són els següents:

1) L'electricitat consumida. Tot el treball ha estat realitzat mitjançant un ordinador, ja sigui per buscar informació, realitzar el programa o escriure la memòria. A més, cal afegir l'energia consumida per aconseguir una bona il·luminació per treballar en els moments en els que una il·luminació natural no era possible.

2) El material d'oficina utilitzat. El paper utilitzat, ja sigui per fer esborranys o notes durant la realització del treball o per la impressió final; la tinta consumida per la impressió; els bolígrafs utilitzats; etc.

Conclusions

Realitzar aquest treball ha sigut una experiència molt bona, ja que poder analitzar i entendre com es gestiona el tràfic aeri dels aeroports per acabar realitzant un programa informàtic que ho simuli, ha sigut gratificant.

A més, en aquest treball s'ha pogut utilitzar l'experiència aconseguida durant tot el grau. Encara que no s'hagin aplicat coneixements concrets de moltes de les matèries del grau, si s'ha pogut aplicar la metodologia apresa en la realització de tots els treballs que s'han hagut de fer durant el grau.

S'han complert els objectius, s'ha elaborat un programa que pot simular el tràfic aeri d'un aeroport. És capaç de simular diferents situacions de tràfic aeri, ja siguin amb pocs avions, com el cas de l'exemple, o amb una quantitat d'avions més elevada.

A part de simular-ho correctament, amb aquesta plataforma els potencials usuaris podran entendre perfectament en quines zones es divideix tot el volum sota responsabilitat del Centre de Control. I podran entendre perfectament per quins estats passa un avió al llarg d'un dels seus trajectes, sempre hi quan no hi hagin imprevistos que facin prendre accions fora del normal, les quals no es tenen en compte a la simulació.

Per últim, cal afegir que en tots els programes informàtics sempre es poden anar arreglant coses per optimitzar-los, ampliar-los i millorar-los. Aquest no és una excepció, de idees per millorar-lo en poden aparèixer moltes, però l'important del programa i del treball és que compleix la funció pel qual ha estat dissenyat.

Agraïments

Un cop finalitzat el treball, vull aprofitar per agrari al meu tutor, el Lluís Pérez Vidal, per la proposta del treball, el qual he trobat molt interessant. Agrair-lo també per tot el temps dedicat a la supervisió, pel suport, l'ajuda i els consells al llarg de tot el treball.

Aprofito per agrair també als familiars i amics que m'han donat els seus punts de vista sobre el programa i m'han fet costat durant aquests últims mesos de l'elaboració d'aquest Treball Final de Grau.

Bibliografia

Referències bibliogràfiques

[1] Joaquín C. Adsuar, *DERECHO AÉREO*, Paraninfo, S.A., 2008.

[2] [http://www.enaire.es/csee/Satellite/navegacion-aerea/es/Page/1078418725163/?other=1083158950596&other2=1083857758201#ancla317, 27 de març de 2016]

[3] [http://www.enaire.es/csee/Satellite/navegacion-aerea/es/Page/1078418725153/?other=1078418770003#ancla2, 28 de març de 2016]

[4] [http://www.takeoffbriefing.com/como-se-divide-y-organiza-el-espacio-aereo-firctrtmaatzcta-video/, 28 de març de 2016]

[5] [http://www.diccionari.cat/, 10 d'abril de 2016]

Bibliografia complementària

[http://www.faa.gov/regulations_policies/handbooks_manuals/, 1 de febrer de 2016]

[http://www.faa.gov/regulations_policies/handbooks_manuals/aviation/pilot_handbook/media/PHAK%20-%20Chapter%2014.pdf, 1 de febrer de 2016]

[http://surcandoloscielos.es/blog/el-espacio-aereo-primera-parte/, 28 de març de 2016]

[http://surcandoloscielos.es/blog/el-espacio-aereo-segunda-parte/, 28 de març de 2016]

[https://docs.python.org/2/library/, 2 d'abril de 2016]

[http://www.pygame.org/docs/, 2 d'abril de 2016]

[http://blog.rvburke.com/2006/11/22/programacion-orientada-a-objetos-en-python/comment-page-1/, 2 d'abril de 2016]

Annex

Arxiu Simulacio.py

```
import Aeroport

#INICIALITZA AEROPORT
a=Aeroport.Aeroport()
print ""
print ""
print ""
print "          ","BENVINGUTS A L'AEROPORT JMC"
print ""
print ""

#INFORMACIO DE LES CAPACITATS MAXIMES DE CADA ZONA DE
L'AEROPORT
print " CARACTERISTIQUES BASIQUES DE L'AEROPORT JMC:"
print ""
print " A) La Plataforma te una capacitat d'estacionament per a 8 avions, i nomes hi"
print "    podra haver 1 avio fent maniobres de rodatge."
print ""
print " B) La Zona de Rodatge de sortida te una capacitat maxima de 2 avions, repartits"
print "    entre la pista de rodatge de sortida i la zona d'espera a pista."
print ""
print " C) La Zona de Rodatge d'entrada te una capacitat maxima de 3 avions, repartits"
print "    entre les dues zones d'espera i la pista de rodatge"
print ""
print " D) Hi ha 1 Pista per aterrar i/o enlairar-se."
print ""
print ""
print ""

#INTRODUIR AVIONS INICIALS
print " INTRODUIR ELS AVIONS PER LA SIMULACIO:"
print ""
print " 1) PLATAFORMA:"
x=raw_input(" Quants avions es volen introduir a la Plataforma? ")
while True:
   while True:
      if x.isdigit():
         x=int(float(x))
         break
```

```
        else:
            x=raw_input (" Cal escriure digits: ")
    if x>a.Plataforma.capacitat:
        print " No hi ha prou espai a la Plataforma per tants avions."
        print " Com a molt hi caben", a.Plataforma.capacitat, "avions"
        x=raw_input(" Escull un nombre d'avions que hi capiguen: ")
    else:
        break
for i in range(x):
    print ""
    print " Avio numero", i+1,"de la Plataforma:"
    avio=a.crear_avio1()
    a.Plataforma.afegir_pre(avio)
print ""
print " 2) AEROVIA D'ARRIBADA: (cal indicar si tornaran a enlairar-se o no)"
y=raw_input(" Quants avions es volen fer aterrar a l'aeroport? ")
while True:
    if y.isdigit():
        y=int(float(y))
        break
    else:
        y=raw_input (" Cal escriure digits: ")

for i in range(y):
    print ""
    print " Avio d'arribada numero", i+1,":"
    avio=a.crear_avio2()
    a.AWY.afegir_arribada(avio)

print ""
print " 3) INICI DE LA SIMULACIO:"
print ""
##### PROGRAMA PER LA INTERFICIE GRAFICA
a.crear_interficie()
a.enganxar_avions()
#LLISTA AVIONS A LA TERMINAL
a.llista_avions_codis()

#BUCLE DE COMANDES
llistacomandes=['1', 'Introduir avions', '2', 'Actualitzar', '3', "Llistes d'avions", '4', 'Forats
lliures', '5', 'Consultar avio', '6', 'Tancar simulador']
ll=[" 1- Introduir avions (d'arribada)", ' 2- Actualitzar', " 3- Llistes d'avions",' 4- Forats
lliures', ' 5- Consultar avio', ' 6- Tancar simulador']
while True:
    #INTRODUCCIO DE LA COMANDA
    print ""
```

```
    print ' LLISTA DE COMANDES:'
    for i in ll:
        print i
    x=raw_input(' Tria una comanda: ')
    while x not in llistacomandes:
        print ""
        print ' No existeix la comanda desitjada!'
        print ""
        print ' Llista de comandes:'
        for i in ll:
            print i
        x=raw_input(' Introdueix la comanda desitjada correctament: ')
    print ""

    #TOTS ELS IFS AMB LES COMANDES POSSIBLES
    #INTRODUIR NOUS AVIONS PERQUE ARRIBIN A L'AEROPORT
    if x=='Introduir avions' or x=='1':
        x = raw_input(" Quants avions nous es volen fer arribar a l'aeroport? ")
        while True:
            if x.isdigit():
                x=int(float(x))
                break
            else:
                x=raw_input (" Cal escriure digits: ")
        for i in range(x):
            print ""
            print " Avio numero", i+1,':'
            avio=a.crear_avio2()
            a.AWY.afegir_arribada(avio)
        a.llista_avions_codis()

    #ACTUALITZAR LES LLISTES AMB ELS AVIONS
    elif x=='Actualitzar' or x=='2':
        a.actualitzar()
        a.llista_avions_codis()
        a.enganxar_avions()

    #MOSTRAR QUINS AVIONS (ELS CODIS) ESTAN A CADA LLISTA
    elif x=="Llistes d'avions" or x== '3':
        a.llista_avions_codis()

    #DONAR INFORMACIO DE QUANTS AVIONS HI CABEN A CADA LLISTA (EN
QUALSEVOL MOMENT DE LA SIMULACIO)
    elif x=='Forats lliures' or x=='4':
        print " LLOCS LLIURES DE CADA ZONA DE L'AEROPORT"
        print " Plataforma:", a.Plataforma.forat_lliure()
```

```
        print " Rodatge a Plataforma:", a.Plataforma.forat_lliure_rodatge()
        print " Zona de rodatge de sortida:",a.PRodatge.forat_lliure_surt() +
a.PRodatge.forat_lliure_espera1()
        print " Zona de rodatge d'entrada:",a.PRodatge.forat_lliure_espera3() +
a.PRodatge.forat_lliure_entra()+a.PRodatge.forat_lliure_espera2()
        print " Pista:",a.Pista.forat_lliure()


    #CONSULTAR LA INFORMACIO D'UN AVIO A PARTIR DEL SEU CODI
D'IDENTIFICACIO
    elif x=='Consultar avio' or x=='5':
        if len(a.llista_codis)>0:
            c=raw_input(" Quin avio vols consultar? ")
            if c not in a.llista_codis:
                print ""
                print " No hi ha cap avio amb aquest codi!"
            else:
                a.buscar_avio(c)
        else:
            print " No s'ha introduit cap avio."

    #TANCAR EL SIMULADOR
    elif x=='Tancar simulador' or x=='6':
        break
```

Arxiu Aeroport.py

```
import Classes
import Actualitzar
import Print_Llistes
import pygame

class Aeroport():
   def __init__(self):
      self.Plataforma=Classes.Plataforma()
      self.PRodatge=Classes.PRodatge()
      self.Pista=Classes.Pista()
      self.CTR=Classes.CTR()
      self.TMA=Classes.TMA()
      self.AWY=Classes.AWY()
      self.llista_codis=[]
      self.llista_llistes=[self.Plataforma.ll_aparcats,
                           self.Plataforma.ll_auto,
                           self.Plataforma.ll_pre,
                           self.Plataforma.ll_entra,
                           self.Plataforma.ll_surt,
                           self.PRodatge.ll_espera2,
                           self.PRodatge.ll_entra,
                           self.PRodatge.ll_espera3,
                           self.PRodatge.ll_surt,
                           self.PRodatge.ll_espera1,
                           self.Pista.ll_aterra,
                           self.CTR.ll_ct,
                           self.CTR.ll_aprox,
                           self.CTR.ll_surt,
                           self.TMA.ll_espera,
                           self.TMA.ll_arribada,
                           self.TMA.ll_sortida,
                           self.AWY.ll_arribada,
                           self.AWY.ll_sortida]
      self.Imatge=pygame.image.load('Fotos/FONS.png')

   #INTRODUIR AVIONS A LA PLATAFORMA
   def crear_avio1(self):
      codi=raw_input(" Codi d'idinteficacio de l'avio: ")
      while len(codi)==0 or codi.isspace()==True:
         codi=raw_input( " Escriu un codi: ")
      while True:
         if codi not in self.llista_codis:
            self.llista_codis.append(codi)
```

```
            break
        else:
            print ""
            print " Ja existeix un avio amb aquest codi!"
            codi=raw_input(" Tria un altre codi de l'avio: ")
    zona_actual="Plataforma"
    estat=""
    direccio=""
    trajectes=1
    avio=Classes.Avio(codi, zona_actual, estat, direccio, trajectes)
    pos=self.Plataforma.buscar_aparcament()
    avio.entra_aparcament(pos)
    self.Plataforma.ocupar_aparcament(pos)
    return avio

#INTRODUIR AVIONS A LES AEROVIES D'ARRIBADA
def crear_avio2(self):
    codi=raw_input(" Codi d'idetificacio de l'avio: ")
    while len(codi)==0 or codi.isspace()==True:
        codi=raw_input( " Escriu un codi: ")
    while True:
        if codi not in self.llista_codis:
            self.llista_codis.append(codi)
            break
        else:
            print ""
            print " Ja existeix un avio amb aquest codi!"
            codi=raw_input(" Tria un altre codi de l'avio: ")
    zona_actual="Aerovia"
    estat="Dirigint-se cap a l'aeroport"
    direccio="Plataforma"
    trajectes=raw_input(" L'avio es torna a enlairar? (1-Si; 2-NO): ")
    while True:
        if trajectes=='1' or trajectes=='Si':
            trajectes=2
            break
        elif trajectes=='2' or trajectes=='No':
            trajectes=1
            break
        else:
            trajectes=raw_input(" Escriu 1, Si, 2 o No: ")
    avio=Classes.Avio(codi, zona_actual, estat, direccio, trajectes)
    return avio

#A PARTIR D'UN CODI BUSCA L'AVIO ENTRE LES LLISTES I MOSTRE LA
SEVA INFORMACIO
```

```
    def buscar_avio(self, codi):
        control=0
        while True:
            for i in range(len(self.llista_llistes)):
                if control==1:
                    break
                else:
                    for t in range(len(self.llista_llistes[i])):
                        if control==1:
                            break
                        else:
                            if self.llista_llistes[i][t].codi==codi:
                                print " INFORMACIO DE L'AVIO:"
                                print " Codi:",self.llista_llistes[i][t].codi
                                print " Zona actual:",self.llista_llistes[i][t].zona_actual
                                print " Estat de l'avio:",self.llista_llistes[i][t].estat
                                p=self.llista_llistes[i][t].pos_aparcament
                                if p==0:
                                    print " Aparcament assignat: No te aparcament assignat"
                                else:
                                    print " Aparcament assignat:",p
                                d=self.llista_llistes[i][t].direccio
                                print " Direccio:",d
                                traj=self.llista_llistes[i][t].trajectes
                                if d=='Plataforma':
                                    if traj==1:
                                        print " No tornara a enlairar-se."
                                    elif traj==2:
                                        print " Tornara a enlairar-se."
                                control=1
                                break
            if control==0:
                print " No s'ha trobat l'avio"
                break
            else:
                break

    #DE LES LLISTES AMB AVIONS CREA I PRINTA LLISTES AMB ELS CODIS
DELS AVIONS
    def llista_avions_codis(self):
        Print_Llistes.llista_avions_codis(self)

    #BUCLE D'ACTUALITZACIO DE LES LLISTES
    def actualitzar(self):
        Actualitzar.actualitzar(self)
```

```
#CREAR UNA INTERFICIE GRAFICA
def crear_interficie(self):
    ##### Crear la pantalla de la interficie
    size=self.Imatge.get_size()
    self.Fons=pygame.display.set_mode(size)
    pygame.display.set_caption("Esquema de l'aeroport JMC")

#POSA EL FONS I ELS AVIONS A LA INTERFICIE
def enganxar_avions(self):
    ##### Enganxar el fons
    self.Fons.blit(self.Imatge,(0,0))
    ##### Enganxar els avions
    #Plataforma
    x=len(self.Plataforma.ll_auto)
    for i in range(x):
        pos=self.Plataforma.ll_auto[i].pos_aparcament
        pos=pos-1
        self.Fons.blit(self.Plataforma.ll_auto[i].avio63,(11+pos*60+pos*14,720))
    x=len(self.Plataforma.ll_pre)
    for i in range(x):
        pos=self.Plataforma.ll_pre[i].pos_aparcament
        pos=pos-1
        self.Fons.blit(self.Plataforma.ll_pre[i].avio61,(11+pos*60+pos*14,720))
    x=len(self.Plataforma.ll_aparcats)
    for i in range(x):
        pos=self.Plataforma.ll_aparcats[i].pos_aparcament
        pos=pos-1
        self.Fons.blit(self.Plataforma.ll_aparcats[i].avio61,(11+pos*60+pos*14,720))
    x=len(self.Plataforma.ll_surt)
    if x==1:
        self.Fons.blit(self.Plataforma.ll_surt[0].avio63,(190,620))
    x=len(self.Plataforma.ll_entra)
    if x==1:
        self.Fons.blit(self.Plataforma.ll_entra[0].avio61,(350,620))
    #PRodatge
    x=len(self.PRodatge.ll_espera1)
    if x==1:
        self.Fons.blit(self.PRodatge.ll_espera1[0].avio63,(11,520))
    x=len(self.PRodatge.ll_surt)
    if x==1:
        self.Fons.blit(self.PRodatge.ll_surt[0].avio62,(160,520))
    x=len(self.PRodatge.ll_espera3)
    if x==1:
        self.Fons.blit(self.PRodatge.ll_espera3[0].avio61,(311,520))
    x=len(self.PRodatge.ll_entra)
    if x==1:
```

```
    self.Fons.blit(self.PRodatge.ll_entra[0].avio62,(410,520))
x=len(self.PRodatge.ll_espera2)
if x==1:
    self.Fons.blit(self.PRodatge.ll_espera2[0].avio61,(529,520))
#Pista
x=len(self.Pista.ll_enlaire)
if x==1:
    self.Fons.blit(self.Pista.ll_enlaire[0].avio63,(100,420))
x=len(self.Pista.ll_aterra)
if x==1:
    self.Fons.blit(self.Pista.ll_aterra[0].avio61,(480,420))
#CTR
x=len(self.CTR.ll_surt)
if x==1:
    self.Fons.blit(self.CTR.ll_surt[0].avio63,(190,270))
x=len(self.CTR.ll_aprox)
if x==1:
    self.Fons.blit(self.CTR.ll_aprox[0].avio61,(410,220))
x=len(self.CTR.ll_ct)
if x==1:
    self.Fons.blit(self.CTR.ll_ct[0].avio61,(410,320))
#TMA
x=len(self.TMA.ll_sortida)
if x==1:
    self.Fons.blit(self.TMA.ll_sortida[0].avio63,(190,70))
x=len(self.TMA.ll_arribada)
if x==1:
    self.Fons.blit(self.TMA.ll_arribada[0].avio61,(410,20))
x=len(self.TMA.ll_espera)
if x==1:
    self.Fons.blit(self.TMA.ll_espera[0].avio61,(410,120))
elif x>1 and x<5:
    for i in range(x):#NOMES MOSTRA 4 AVIONS COM A MOLT
        self.Fons.blit(self.TMA.ll_espera[i].avio61,(311+i*11+i*60,120))
elif x>4:
    for i in range(4):#NOMES MOSTRA 4 AVIONS COM A MOLT
        self.Fons.blit(self.TMA.ll_espera[i].avio61,(311+i*11+i*60,120))

##### Actualitzar la pantalla de la interficie
pygame.display.flip()
```

Arxiu Actualitzar.py

```
def actualitzar(self):
   avions_moguts=[] #avio.mov_fet() despres de treurel, avions_moguts.append(avio)
despres d'afegirlo

   #De AWY.arribades a TMA.arribades
   if len(self.AWY.ll_arribada)>0:
      if self.AWY.temps_arribada==0:
         avio=self.AWY.treure_arribada()
         avio.mov_fet()
         self.TMA.afegir_arribada(avio)
         avions_moguts.append(avio)
      self.AWY.temps_arribada_actualitzar()

   #De TMA.arribades a TMA.espera
   if len(self.TMA.ll_arribada)>0:
      if self.TMA.ll_arribada[0].mov==1:
         avio=self.TMA.treure_arribada()
         avio.mov_fet()
         self.TMA.afegir_espera(avio)
         avions_moguts.append(avio)

   #De TMA.espar a CTR.aprox
   if len(self.TMA.ll_espera)>0:
      if self.TMA.ll_espera[0].mov==1:
         if self.CTR.forat_lliure_aprox()>0:
            if self.CTR.forat_lliure_ct()>0 and self.Pista.forat_lliure()>0 and
self.PRodatge.forat_lliure_espera2()>0 and self.PRodatge.forat_lliure_entra()>0 and
self.PRodatge.temps_espera1<2:
               avio=self.TMA.treure_espera()
               avio.mov_fet()
               self.CTR.afegir_aprox(avio)
               avions_moguts.append(avio)
            elif self.CTR.forat_lliure_ct()==0 and self.Pista.forat_lliure()>0 and
self.PRodatge.forat_lliure_espera2()>0 and self.PRodatge.forat_lliure_entra()>0 and
self.PRodatge.forat_lliure_espera3()>0 and self.PRodatge.temps_espera1<2:
               avio=self.TMA.treure_espera()
               avio.mov_fet()
               self.CTR.afegir_aprox(avio)
               avions_moguts.append(avio)

   #De CTR.aprox a CTR.ct
   if len(self.CTR.ll_aprox)>0:
      if self.CTR.ll_aprox[0].mov==1:
```

```
            avio=self.CTR.treure_aprox()
            avio.mov_fet()
            self.CTR.afegir_ct(avio)
            avions_moguts.append(avio)

    #De CTR.ct a Pista.aterra
    if len(self.CTR.ll_ct)>0:
        if self.CTR.ll_ct[0].mov==1:
            avio=self.CTR.treure_ct()
            avio.mov_fet()
            self.Pista.afegir_aterra(avio)
            avions_moguts.append(avio)

    #De Pista.aterra a PRodatge.entra o PRodatge.espera2
    Control_Pista=0
    if len(self.Pista.ll_aterra)>0:
        Control_Pista=1
        if self.Pista.ll_aterra[0].mov==1:
            if self.PRodatge.forat_lliure_entra()>0 and self.PRodatge.forat_lliure_espera3()>0:
                avio=self.Pista.treure_aterra()
                avio.mov_fet()
                self.PRodatge.afegir_entra(avio)
                avions_moguts.append(avio)
            else:
                avio=self.Pista.treure_aterra()
                avio.mov_fet()
                self.PRodatge.afegir_espera2(avio)
                avions_moguts.append(avio)

    #De PRodatge.espera2 a PRodatge.entra
    if len(self.PRodatge.ll_espera2)>0:
        if self.PRodatge.ll_espera2[0].mov==1:
            if self.PRodatge.forat_lliure_entra()>0:
                if self.PRodatge.forat_lliure_espera3()>0:
                    avio=self.PRodatge.treure_espera2()
                    avio.mov_fet()
                    self.PRodatge.afegir_entra(avio)
                    avions_moguts.append(avio)
                elif self.PRodatge.forat_lliure_espera3()==0 and
self.Plataforma.forat_lliure_rodatge()>0 and self.Plataforma.forat_lliure()>0 and
self.Plataforma.temps_auto<2:
                    avio=self.PRodatge.treure_espera2()
                    avio.mov_fet()
                    self.PRodatge.afegir_entra(avio)
                    avions_moguts.append(avio)
```

```
    #De PRodatge.entra a PRodatge.espera3 o Plataforma.entra
    if len(self.PRodatge.ll_entra)>0:
        if self.PRodatge.ll_entra[0].mov==1:
            if self.Plataforma.forat_lliure_rodatge()>0 and self.Plataforma.forat_lliure()>0 and
self.Plataforma.temps_auto<2:
                avio=self.PRodatge.treure_entra()
                avio.mov_fet()
                self.Plataforma.afegir_entra(avio)
                avions_moguts.append(avio)
            else:
                avio=self.PRodatge.treure_entra()
                avio.mov_fet()
                self.PRodatge.afegir_espera3(avio)
                avions_moguts.append(avio)

    #De PRodatge.espera3 a Plataforma.entra
    if len(self.PRodatge.ll_espera3)>0:
        if self.PRodatge.ll_espera3[0].mov==1:
            if self.Plataforma.forat_lliure_rodatge()>0 and self.Plataforma.forat_lliure()>0 and
self.Plataforma.temps_auto<2:
                avio=self.PRodatge.treure_espera3()
                avio.mov_fet()
                self.Plataforma.afegir_entra(avio)
                avions_moguts.append(avio)

    #De Plataforma.entra a Plataforma.Aparcats o Plataforma.pre
    Control_Plata=0
    if len(self.Plataforma.ll_entra)>0:
        Control_Plata=1
        if self.Plataforma.ll_entra[0].mov==1:
            if self.Plataforma.ll_entra[0].trajectes==2:
                avio=self.Plataforma.treure_entra()
                avio.mov_fet()
                self.Plataforma.afegir_pre(avio)
                avions_moguts.append(avio)
            else:
                avio=self.Plataforma.treure_entra()
                avio.mov_fet()
                self.Plataforma.afegir_aparcats(avio)
                avions_moguts.append(avio)

    #De Plataforma.pre a Plataforma.auto
    if len(self.Plataforma.ll_pre)>0:
        if self.Plataforma.ll_pre[0].mov==1:
            if self.Plataforma.temps_pre==0:
                avio=self.Plataforma.treure_pre()
```

```
                avio.mov_fet()
                self.Plataforma.afegir_auto(avio)
                avions_moguts.append(avio)
            self.Plataforma.temps_pre_actualitzar()

    #De Plataforma.auto a Plataforma.surt
    if len(self.Plataforma.ll_auto)>0:
        if self.Plataforma.ll_auto[0].mov==1:
            self.Plataforma.temps_auto_incrementa()
            if self.Plataforma.forat_lliure_rodatge()>0 and Control_Plata==0:
                if self.PRodatge.forat_lliure_surt()>0 and
self.PRodatge.forat_lliure_espera1()>0:
                    avio=self.Plataforma.treure_auto()
                    avio.mov_fet()
                    self.Plataforma.afegir_surt(avio)
                    avions_moguts.append(avio)
                    self.Plataforma.temps_auto_zero()
                elif self.PRodatge.forat_lliure_surt()>0 and
self.PRodatge.forat_lliure_espera1()==0 and self.Pista.forat_lliure()>0 and
Control_Pista==0:
                    avio=self.Plataforma.treure_auto()
                    avio.mov_fet()
                    self.Plataforma.afegir_surt(avio)
                    avions_moguts.append(avio)
                    self.Plataforma.temps_auto_zero()
                elif self.PRodatge.forat_lliure_surt()==0 and self.Pista.forat_lliure()>0 and
Control_Pista==0 and self.CTR.forat_lliure_ct()>0 and self.CTR.forat_lliure_aprox()>0
and len(self.TMA.ll_espera)==0:
                    avio=self.Plataforma.treure_auto()
                    avio.mov_fet()
                    self.Plataforma.afegir_surt(avio)
                    avions_moguts.append(avio)
                    self.Plataforma.temps_auto_zero()

    #De Plataforma.surt a PRodatge.surt
    if len(self.Plataforma.ll_surt)>0:
        if self.Plataforma.ll_surt[0].mov==1:
            avio=self.Plataforma.treure_surt()
            avio.mov_fet()
            self.PRodatge.afegir_surt(avio)
            avions_moguts.append(avio)

    #De PRodatge.surt a PRodatge.espera1 o Pista.enlaire
    if len(self.PRodatge.ll_surt)>0:
        if self.PRodatge.ll_surt[0].mov==1:
            if self.Pista.forat_lliure()>0 and Control_Pista==0 and self.CTR.forat_lliure_ct()>0:
```

```
            avio=self.PRodatge.treure_surt()
            avio.mov_fet()
            self.Pista.afegir_enlaire(avio)
            avions_moguts.append(avio)
        else:
            avio=self.PRodatge.treure_surt()
            avio.mov_fet()
            self.PRodatge.afegir_espera1(avio)
            avions_moguts.append(avio)

#De PRodatge.espera1 a Pista.enlaire
if len(self.PRodatge.ll_espera1)>0:
    if self.PRodatge.ll_espera1[0].mov==1:
        if self.Pista.forat_lliure()>0 and Control_Pista==0 and self.CTR.forat_lliure_ct()>0:
            avio=self.PRodatge.treure_espera1()
            avio.mov_fet()
            self.Pista.afegir_enlaire(avio)
            avions_moguts.append(avio)
            self.PRodatge.temps_espera1_zero()
        else:
            self.PRodatge.temps_espera1_incrementa()

#De Pista.enlaire a CTR.surt
if len(self.Pista.ll_enlaire)>0:
    if self.Pista.ll_enlaire[0].mov==1:
        avio=self.Pista.treure_enlaire()
        avio.mov_fet()
        self.CTR.afegir_surt(avio)
        avions_moguts.append(avio)

#De CTR.surt a TMA.sortida
if len(self.CTR.ll_surt)>0:
    if self.CTR.ll_surt[0].mov==1:
        avio=self.CTR.treure_surt()
        avio.mov_fet()
        self.TMA.afegir_sortida(avio)
        avions_moguts.append(avio)

#De TMA.sortida a AWY.sortida
if len(self.TMA.ll_sortida)>0:
    if self.TMA.ll_sortida[0].mov==1:
        avio=self.TMA.treure_sortida()
        avio.mov_fet()
        self.AWY.afegir_sortida(avio)
        avions_moguts.append(avio)
```

```
#Assignar aparcaments:
if len(self.Plataforma.ll_entra)>0:
    if self.Plataforma.ll_entra[0].pos_aparcament==0:
        pos=self.Plataforma.buscar_aparcament()
        if pos!=0:
            self.Plataforma.ll_entra[0].entra_aparcament(pos)
            self.Plataforma.ocupar_aparcament(pos)

if len(self.PRodatge.ll_espera3)>0:
    if self.PRodatge.ll_espera3[0].pos_aparcament==0:
        pos=self.Plataforma.buscar_aparcament()
        if pos!=0:
            self.PRodatge.ll_espera3[0].entra_aparcament(pos)
            self.Plataforma.ocupar_aparcament(pos)

if len(self.PRodatge.ll_entra)>0:
    if self.PRodatge.ll_entra[0].pos_aparcament==0:
        pos=self.Plataforma.buscar_aparcament()
        if pos!=0:
            self.PRodatge.ll_entra[0].entra_aparcament(pos)
            self.Plataforma.ocupar_aparcament(pos)
if len(self.PRodatge.ll_espera2)>0:
    if self.PRodatge.ll_espera2[0].pos_aparcament==0:
        pos=self.Plataforma.buscar_aparcament()
        if pos!=0:
            self.PRodatge.ll_espera2[0].entra_aparcament(pos)
            self.Plataforma.ocupar_aparcament(pos)


#ACTUALITZAR MOV
for i in range(len(avions_moguts)):
    avions_moguts[i].nou_mov()
```

Arxiu Print_Llistes.py

```
def llista_avions_codis(self):
    print " ESQUEMA DE L'AEROPORT:"
    print ""

    #PLATAFORMA
    ll_aparcats_codis=[]
    ll_auto_codis=[]
    ll_pre_codis=[]
    ll_rodatge_plata_codis=[] #mateixa llista de codis pels avions en rodatge a
plataforma(entrant i surtint)
    x=len(self.Plataforma.ll_aparcats)
    for i in range(x):
        ll_aparcats_codis.append(self.Plataforma.ll_aparcats[i].codi)
    x=len(self.Plataforma.ll_auto)
    for i in range(x):
        ll_auto_codis.append(self.Plataforma.ll_auto[i].codi)
    x=len(self.Plataforma.ll_pre)
    for i in range(x):
        ll_pre_codis.append(self.Plataforma.ll_pre[i].codi)
    x=len(self.Plataforma.ll_entra)
    for i in range(x):
        ll_rodatge_plata_codis.append(self.Plataforma.ll_entra[i].codi)
    x=len(self.Plataforma.ll_surt)
    for i in range(x):
        ll_rodatge_plata_codis.append(self.Plataforma.ll_surt[i].codi)

    #ZONA DE RODATGE ARRIBADES
    ll_entra_codis=[]
    ll_espera2_codis=[]
    ll_espera3_codis=[]
    x=len(self.PRodatge.ll_espera2)
    for i in range(x):
        ll_espera2_codis.append(self.PRodatge.ll_espera2[i].codi)
    x=len(self.PRodatge.ll_entra)
    for i in range(x):
        ll_entra_codis.append(self.PRodatge.ll_entra[i].codi)
    x=len(self.PRodatge.ll_espera3)
    for i in range(x):
        ll_espera3_codis.append(self.PRodatge.ll_espera3[i].codi)

    #ZONA DE RODATGE SORTIDES
    ll_surt_codis=[]
    ll_espera1_codis=[]
```

```
x=len(self.PRodatge.ll_surt)
for i in range(x):
   ll_surt_codis.append(self.PRodatge.ll_surt[i].codi)
x=len(self.PRodatge.ll_espera1)
for i in range(x):
   ll_espera1_codis.append(self.PRodatge.ll_espera1[i].codi)

#PISTA
ll_p_codis=[]
x=len(self.Pista.ll_aterra)
for i in range(x):
   ll_p_codis.append(self.Pista.ll_aterra[i].codi)
x=len(self.Pista.ll_enlaire)
for i in range(x):
   ll_p_codis.append(self.Pista.ll_enlaire[i].codi)

#CTR
ll_ct_codis=[]
ll_aprox_codis=[]
ll_surt_ctr_codis=[]
x=len(self.CTR.ll_ct)
for i in range(x):
   ll_ct_codis.append(self.CTR.ll_ct[i].codi)
x=len(self.CTR.ll_aprox)
for i in range(x):
   ll_aprox_codis.append(self.CTR.ll_aprox[i].codi)
x=len(self.CTR.ll_surt)
for i in range(x):
   ll_surt_ctr_codis.append(self.CTR.ll_surt[i].codi)

#TMA
ll_espera_codis=[]
ll_arribada_codis=[]
ll_sortida_codis=[]
x=len(self.TMA.ll_espera)
for i in range(x):
   ll_espera_codis.append(self.TMA.ll_espera[i].codi)
x=len(self.TMA.ll_arribada)
for i in range(x):
   ll_arribada_codis.append(self.TMA.ll_arribada[i].codi)
x=len(self.TMA.ll_sortida)
for i in range(x):
   ll_sortida_codis.append(self.TMA.ll_sortida[i].codi)

#AWY
ll_awy_a_codis=[]
```

```
    ll_awy_s_codis=[]
    x=len(self.AWY.ll_arribada)
    for i in range(x):
        ll_awy_a_codis.append(self.AWY.ll_arribada[i].codi)
    x=len(self.AWY.ll_sortida)
    for i in range(x):
        ll_awy_s_codis.append(self.AWY.ll_sortida[i].codi)

    #PRINT
    print " AWY (Zona aeria): [Arribada][Sortida:                  ",
self.AWY.temps_arribada,ll_awy_a_codis, ll_awy_s_codis
    print ""
    print " TMA (Zona aeria)  [Arribada][Sortida:                  ", ll_arribada_codis,
ll_sortida_codis
    print "                   [Zona d'espera]:                    ", ll_espera_codis
    print ""
    print " CTR (Zona aeria): [Aproximacio][Sortida:               ", ll_aprox_codis,
ll_surt_ctr_codis
    print "                   [Circuit de transit]:               ", ll_ct_codis
    print ""
    print " PISTA:            [Pista]:                            ", ll_p_codis
    print ""
    print " ZONA RODATGE:     [Punt espera2][Punt espera1]:          ", ll_espera2_codis,
ll_espera1_codis,self.PRodatge.temps_espera1
    print "                   [P.rodatge entrada][P.rodatge sortida]:  ", ll_entra_codis, ll_surt_codis
    print "                   [Punt espera3]:                     ", ll_espera3_codis
    print ""
    print " PLATAFORMA:       [Zona rodatge plataforma]:             ",
ll_rodatge_plata_codis
    print "                   [Esperant auto.][Preparacio][Aparcats]:  ",
self.Plataforma.temps_auto, ll_auto_codis, self.Plataforma.temps_pre, ll_pre_codis,
ll_aparcats_codis
```

Arxiu Classes.py

```
import random
import pygame

#CLASSE AVIO, ES CREARAN ELS AVIONS QUE CIRCULARAN PER
L'AEROPORT
class Avio:
    def __init__(self, codi, zona_actual, estat, direccio, trajectes):
        self.codi=codi
        self.zona_actual=zona_actual
        self.estat=estat
        self.direccio=direccio #aparcament o zona2
        self.trajectes=trajectes #0: Ha arribat a la destinacio final; 1: Esta realitzant el seu
ultim trajecte; 2: Els avions que estan aterrant i s'han de tornar a enlairar.
        self.mov=1
        self.pos_aparcament=0
        x=random.randint(0,3)

avions1=['Fotos/avio01.png','Fotos/avio11.png','Fotos/avio21.png','Fotos/avio31.png','Fotos
/avio41.png']

avions2=['Fotos/avio02.png','Fotos/avio12.png','Fotos/avio22.png','Fotos/avio32.png','Fotos
/avio42.png']

avions3=['Fotos/avio03.png','Fotos/avio13.png','Fotos/avio23.png','Fotos/avio33.png','Fotos
/avio43.png']
        self.avio61=pygame.image.load(avions1[x])
        self.avio62=pygame.image.load(avions2[x])
        self.avio63=pygame.image.load(avions3[x])
    def mov_fet(self):
        self.mov=0
    def nou_mov(self):
        self.mov=1
    def entra_aparcament(self,pos):
        self.pos_aparcament=pos
    def surt_aparcament(self):
        self.pos_aparcament=0

#CLASSES QUE FORMARAN PART DE L'AEROPORT
class Plataforma:
    def __init__(self):
        self.capacitat=8
        self.capacitat_rodatge=1
        self.aparcaments=[0]*8
```

```
        self.ll_aparcats=[]
        self.ll_auto=[]
        self.ll_pre=[]
        self.ll_entra=[]
        self.ll_surt=[]
        self.temps_pre=0
        self.temps_auto=0
    def forat_lliure(self):
        return (self.capacitat - len(self.ll_aparcats) - len(self.ll_pre) - len(self.ll_auto) -
len(self.ll_entra) - len(self.ll_surt))
    def forat_lliure_rodatge(self):
        return (self.capacitat_rodatge - len(self.ll_entra) - len(self.ll_surt))
    def afegir_aparcats(self,avio):
        avio.estat="Aparcat"
        avio.direccio="-"
        avio.trajectes=0
        self.ll_aparcats.append(avio)
    def afegir_auto(self,avio):
        avio.estat="Esperant autoritzacio per rodatge"
        self.ll_auto.append(avio)
    def afegir_pre(self,avio):
        avio.estat="Preparant-se per arrencar"
        avio.direccio="Un altre aeroport"
        self.ll_pre.append(avio)
    def afegir_entra(self,avio):
        avio.zona_actual='Plataforma'
        avio.estat="En Rodatge"
        self.ll_entra.append(avio)
    def afegir_surt(self,avio):
        avio.estat="En Rodatge"
        self.ll_surt.append(avio)
    def treure_auto(self):
        avio=self.ll_auto[0]
        self.ll_auto=self.ll_auto[1:]
        pos=avio.pos_aparcament
        avio.surt_aparcament()
        Plataforma.buidar_aparcament(self,pos)
        return avio
    def treure_pre(self):
        avio=self.ll_pre[0]
        self.ll_pre=self.ll_pre[1:]
        return avio
    def treure_entra(self):
        avio=self.ll_entra[0]
        self.ll_entra=self.ll_entra[1:]
        return avio
```

```
    def treure_surt(self):
        avio=self.ll_surt[0]
        self.ll_surt=self.ll_surt[1:]
        return avio
    def temps_pre_actualitzar(self):
        if self.temps_pre==0:
            self.temps_pre=random.randint(0,3)
        else:
            self.temps_pre=self.temps_pre-1
    def temps_auto_incrementa(self):
        self.temps_auto=self.temps_auto+1
    def temps_auto_zero(self):
        self.temps_auto=0
    def buscar_aparcament(self):
        i=1
        n=0
        while i==1 and n<8:
          if self.aparcaments[n]==0:
              i=0
          else:
              n=n+1
        if n==8:
            pos=0
        else:
            pos=n+1
        return pos
    def ocupar_aparcament(self, pos):
        pos=pos-1
        self.aparcaments[pos]=1
    def buidar_aparcament(self, pos):
        pos=pos-1
        self.aparcaments[pos]=0

class PRodatge:
    def __init__(self):
        self.capacitat_entra=1
        self.capacitat_surt=1
        self.capacitat_espera1=1
        self.capacitat_espera2=1
        self.capacitat_espera3=1
        self.ll_entra=[]
        self.ll_surt=[]
        self.ll_espera1=[]
        self.ll_espera2=[]
        self.ll_espera3=[]
        self.temps_espera1=0#nombre d'actualitzacions que un avio s'espera a la zona espera1
```

```
def forat_lliure_entra(self):
    return (self.capacitat_entra - len(self.ll_entra))
def forat_lliure_surt(self):
    return (self.capacitat_surt - len(self.ll_surt))
def forat_lliure_espera1(self):
    return (self.capacitat_espera1 - len(self.ll_espera1))
def forat_lliure_espera2(self):
    return (self.capacitat_espera2 - len(self.ll_espera2))
def forat_lliure_espera3(self):
    return (self.capacitat_espera3 - len(self.ll_espera3))
def afegir_entra(self,avio):
    avio.zona_actual='Zona de rodatge'
    avio.estat='En Rodatge'
    self.ll_entra.append(avio)
def afegir_surt(self,avio):
    avio.zona_actual='Zona de rodatge'
    avio.estat='En Rodatge'
    self.ll_surt.append(avio)
def afegir_espera1(self,avio):
    avio.estat="Esperant autoritzacio per entrar a pista"
    self.ll_espera1.append(avio)
def afegir_espera2(self,avio):
    avio.zona_actual='Zona de rodatge'
    avio.estat="Esperant autoritzacio per rodatge"
    self.ll_espera2.append(avio)
def afegir_espera3(self,avio):
    avio.estat="Esperant autoritzacio per entrar a plataforma"
    self.ll_espera3.append(avio)
def treure_entra(self):
    avio=self.ll_entra[0]
    self.ll_entra=self.ll_entra[1:]
    return avio
def treure_surt(self):
    avio=self.ll_surt[0]
    self.ll_surt=self.ll_surt[1:]
    return avio
def treure_espera1(self):
    avio=self.ll_espera1[0]
    self.ll_espera1=self.ll_espera1[1:]
    return avio
def treure_espera2(self):
    avio=self.ll_espera2[0]
    self.ll_espera2=self.ll_espera2[1:]
    return avio
def treure_espera3(self):
    avio=self.ll_espera3[0]
```

```
        self.ll_espera3=self.ll_espera3[1:]
        return avio
    def temps_espera1_incrementa(self):
        self.temps_espera1=self.temps_espera1+1
    def temps_espera1_zero(self):
        self.temps_espera1=0

class Pista:
    def __init__(self):
        self.capacitat=1
        self.ll_aterra=[]
        self.ll_enlaire=[]
    def forat_lliure(self):
        return (self.capacitat - len(self.ll_aterra) - len(self.ll_enlaire))
    def afegir_aterra(self,avio):
        avio.zona_actual='Pista'
        avio.estat="Realitzant l'aterratge"
        self.ll_aterra.append(avio)
    def afegir_enlaire(self,avio):
        avio.zona_actual='Pista'
        avio.estat="Realitzant l'enlairament"
        self.ll_enlaire.append(avio)
    def treure_aterra(self):
        avio=self.ll_aterra[0]
        self.ll_aterra=self.ll_aterra[1:]
        return avio
    def treure_enlaire(self):
        avio=self.ll_enlaire[0]
        self.ll_enlaire=self.ll_enlaire[1:]
        return avio

class CTR:
    def __init__(self):
        self.capacitat_aprox=1
        self.capacitat_ct=1
        self.ll_aprox=[]
        self.ll_ct=[]
        self.ll_surt=[]
    def forat_lliure_aprox(self):
        return (self.capacitat_aprox - len(self.ll_aprox))
    def forat_lliure_ct(self):
        return (self.capacitat_ct - len(self.ll_ct))
    def afegir_aprox(self,avio):
        avio.zona_actual='CTR'
        avio.estat="Realitzant aproximacio per aterrar"
        self.ll_aprox.append(avio)
```

```
    def afegir_ct(self,avio):
        avio.estat="Maniobrant en el circuit de transit"
        self.ll_ct.append(avio)
    def afegir_surt(self,avio):
        avio.zona_actual='CTR'
        avio.estat="Allunyant-se de l'aeroport"
        self.ll_surt.append(avio)
    def treure_aprox(self):
        avio=self.ll_aprox[0]
        self.ll_aprox=self.ll_aprox[1:]
        return avio
    def treure_ct(self):
        avio=self.ll_ct[0]
        self.ll_ct=self.ll_ct[1:]
        return avio
    def treure_surt(self):
        avio=self.ll_surt[0]
        self.ll_surt=self.ll_surt[1:]
        return avio

class TMA:
    def __init__(self):
        self.ll_espera=[]
        self.ll_arribada=[]
        self.ll_sortida=[]
    def afegir_espera(self,avio):
        avio.estat="Esperant autoritzacio per realitzar aproximacio"
        self.ll_espera.append(avio)
    def afegir_arribada(self,avio):
        avio.zona_actual='TMA'
        self.ll_arribada.append(avio)
    def afegir_sortida(self,avio):
        avio.zona_actual='TMA'
        avio.estat="Dirigint-se cap a una aerovia"
        self.ll_sortida.append(avio)
    def treure_espera(self):
        avio=self.ll_espera[0]
        self.ll_espera=self.ll_espera[1:]
        return avio
    def treure_arribada(self):
        avio=self.ll_arribada[0]
        self.ll_arribada=self.ll_arribada[1:]
        return avio
    def treure_sortida(self):
        avio=self.ll_sortida[0]
        self.ll_sortida=self.ll_sortida[1:]
```

```
        return avio

class AWY:
    def __init__(self):
        self.ll_arribada=[]
        self.ll_sortida=[]
        self.temps_arribada=0
    def afegir_arribada(self,avio):
        self.ll_arribada.append(avio)
    def afegir_sortida(self,avio):
        avio.zona_actual='AWY'
        avio.estat="En ruta per l'aerovia"
        self.ll_sortida.append(avio)
    def treure_arribada(self):
        avio=self.ll_arribada[0]
        self.ll_arribada=self.ll_arribada[1:]
        return avio
    def treure_sortida(self):
        avio=self.ll_sortida[0]
        self.ll_sortida=self.ll_sortida[1:]
        return avio
    def temps_arribada_actualitzar(self):
        if self.temps_arribada==0:
            self.temps_arribada=random.randint(0,3)
        else:
            self.temps_arribada=self.temps_arribada-1
```

Arxiu Fons.py

- Codi utilitzat per crear la imatge de fons de la interfície gràfica:

```
A=pygame.Surface((600,800))

A.fill((204,153,255),(0,0,600,199))#TMA
A.fill((255,204,153),(0,201,600,198))#CTR
A.fill((255,153,153),(0,401,600,98))#PISTA
A.fill((204,255,153),(0,501,299,98))#ZONA RODATGE 1
A.fill((102,204,0),(6,505,70,90))
A.fill((204,255,153),(301,501,299,98))#ZONA RODATGE 2
A.fill((102,204,0),(306,505,70,90))
A.fill((102,204,0),(524,505,70,90))
A.fill((153,204,255),(0,601,600,199))#RODATGE PLATAFORMA
A.fill((102,178,255),(6,700,70,95))#PLATAFORMA ESTACIONAMENTS
A.fill((102,178,255),(6+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4+70+4+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4+70+4+70+4+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4+70+4+70+4+70+4+70+4,700,70,95))
A.fill((102,178,255),(6+70+4+70+4+70+4+70+4+70+4+70+4+70+4,700,70,95))

pygame.image.save(A,'FONS.png')
```