Perry Rhodan: The Cosmic League
Memoria del Proyecto

Pedro Redondo Quesada
46471662-R
20/06/2014

Índice

Análisis de requisitos....3
Casos de Uso....14
Diseño de Interficie Gráfica....41
Especificación....54
Diseño....72
Implementación....97
Estudio de la Inteligencia Artificial....100
Progreso del Proyecto....116
Anexo....128

Análisis de requisitos

Análisis de requisitos....3
1. Resumen....4
2. Planeta....5
2.1. Recursos....5
3. Jugador....6
3.1. Vuelo....7
3.2. Acciones Planetarias....7
3.3. Realizar intervenciones....9
4. Tecnologías....10
4.1 Parámetros del jugador....10
4.2 Modificadores de Acciones....11
4.3 Acciones Especiales....11
5. Intervenciones....12
5.1. Aumentos....12
5.2. Alteración de las reglas....13

1. Resumen

- El objetivo del juego es el de ganar 70 Megagalax
  - El jugador gana megagalax transportando recursos desde un planeta a otro
    - El movimiento y numero de recursos que el jugador puede transportar está limitado
      - El jugador puede usar los megagalax obtenidos para comprar mejoras que le permitirán transportar recursos de manera mas eficiente
    - El jugador rival tiene el mismo objetivo
      - Las intervenciones del jugador rival condicionaran al otro jugador

2. Planeta

El lugar sobre el cual los jugadores interactuarán. Cada planeta esta colocado a una distancia fija del Sol que será la misma en todas las partidas:

Sol Coronis Naginther Vinau II Taghelia Talmazzar Dahkit

2.1. Recursos

Principal fuente de megagalax del juego.

- Cada planeta demanda un tipo especifico de recurso
- Cada planeta dispone de una serie de recursos para ser explotados (excluyen al anterior[1])

Los recursos estarán representados por cartas de recurso. Cada carta de recurso estará definida por:

- El tipo de recurso
- Un valor (2-3) que será el número de megagalax que obtendremos

Las cartas de recursos son reversibles. Los recursos del mismo tipo podrán ser apilados independientemente del valor que tengan.

1 Las cartas de recursos que coincidan con el tipo de recurso demandado por el planeta en el que se encuentran, serán giradas para mostrar el recurso alternativo

3. Jugador

Representación del usuario. Puede ser:

- Un jugador humano (Online)
- Una Inteligencia Artificial (Varios tipos)

Información relativa al jugador:

- Megagalax acumulados
- Posición respecto al sol
- Recursos que transporta
- Cartas disponibles (5 como máximo)
- Tecnologías adquiridas (condiciona los puntos anteriores)

Durante su turno, cada jugador tendrá derecho a realizar:

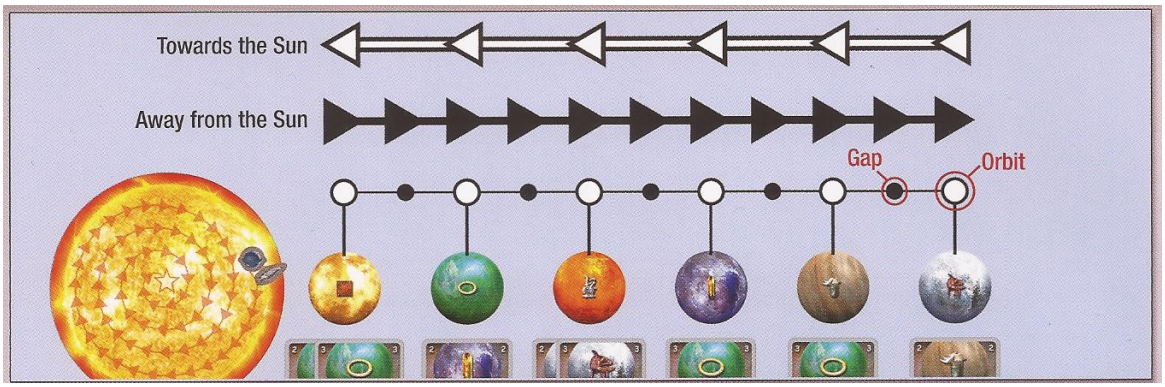
- 2 acciones planetarias
- 2 intervenciones
- 1 acción de vuelo

Al final de su turno, el jugador descartará las cartas que no necesite.

3.1. Vuelo

Al principio de cada turno el jugador lanzará su dado para determinar las unidades de energía (rango de movimiento) disponible. Consideraciones:

- En caso de sacar un 1, se volverá a lanzar el dado y se le sumará 1 al resultado de la tirada (efecto acumulativo)
- El movimiento en dirección opuesta al Sol es mas costoso (ver figura posterior)
- No es necesario utilizar todo el movimiento
- Esta acción solo se puede utilizar una vez por turno[2].
- Realizar una acción planetaria de cualquier tipo terminará el vuelo
- Tanto salir como entrar a un planeta desde su órbita cuesta 1 unidad de energía independientemente de la dirección del vuelo



No es necesario que el vuelo termine en un planeta, un vuelo puede terminar en cualquier parte del recorrido. Las unidades sobrantes de energía se perderán tras el vuelo

3.2. Acciones Planetarias

Durante su turno, el jugador tendrá derecho a 2 acciones planetarias[3]. A saber:

- Obtener recursos
- Cambiar recursos por megagalax
- Cambiar megagalax por tecnología

Para realizar este tipo de acciones es imprescindible que el jugador se encuentre directamente en el planeta en cuestión[4] sobre el que quiere realizar la acción. Estas acciones no tienen porque realizarse en un mismo planeta

2 Excepciones: Intervención Afterburner ignition
3 Excepciones: Intervención Agent Infiltration
4 Excepciones: Tecnología Orbital Station

3.2.1. Obtención de recursos

Se considera la acción de cargar un único contenedor con un tipo de recursos de el planeta en el que se encuentra. Consideraciones:

- Para cargar un contenedor, éste debe estar vacío
- El contenedor será cargado con todas las cartas del tipo de recursos escogido (pila de recursos) que haya en el planeta en cuestión
- Tras realizar la acción, el contenedor quedará sellado y no se podrá volver a utilizar bajo ningún concepto, hasta que no se descargue en el planeta de destino
- El jugador no sabrá en ningún momento lo que hay en el reverso de las cartas cargadas

Un jugador podrá disponer de tantos contenedores como cartas de tecnología Container tenga (1-3)

3.2.2. Cambio de recursos

Se considera la acción de vaciar de cambiar el contenido de un único contenedor por tantos megagalax como la suma del valor de los recursos que contenga. Tras esto, se girarán las cartas de recursos cambiadas, y éstos pasaran a formar parte del inventario del planeta de destino. Consideraciones:

- Al descargar recursos sobre un planeta que no disponga de ningún tipo de recurso se obtendrá 1 megagalax adicional por carta de recurso descargada
- Si al girar las cartas de recursos, apareciera una pareja de cartas del mismo tipo, ésta sera removida del juego (consideramos únicamente las cartas descargadas)
- En caso de producirse el suceso anterior con 2 o más parejas, el jugador podrá elegir cual de ellas será eliminada del juego

En caso de agotarse todos los recursos del tablero antes de cumplir la cuota de megagalax, el jugador que haya acumulado mas puntos será declarado vencedor

3.2.3. Compra de Tecnologías

Se considera la acción de añadir una nueva tecnología de la mano del jugador al haber de tecnologías de éste. Realizar esta acción consume tantos megagalax como tecnologías de las que dispone actualmente el jugador. Consideraciones

- Inicialmente, el jugador dispondrá de la tecnología Start Card. Esta tecnología también se tiene en cuenta para calcular el coste
- El coste de las tecnologías se actualiza al final de un turno, por lo que si el jugador adquiere una segunda tecnología durante el presente turno, el coste no habrá incrementado aún.
- El efecto de las tecnologías es permanente y no se puede anular

En caso de empate a megagalax tras agotarse los recursos, el jugador que disponga de un mayor número de tecnologías será declarado vencedor. De producirse de nuevo un empate el juego quedará en empate

3.3. Realizar intervenciones

Durante su turno, el jugador tendrá derecho a realizar 2 intervenciones que alterarán el desarrollo normal del juego. Consideraciones:

- Las intervenciones pueden jugarse en cualquier momento del turno y desde cualquier lugar
- Una intervención sólo podrá jugarse si se cumplen las condiciones de ejecución de ésta
- El oponente tiene la posibilidad de neutralizar la intervención descartándose esa misma carta[5] de su mano

El apartado Intervenciones (5) contiene información concreta sobre los tipos de intervenciones así como sus condiciones de ejecución

5 Excepciones: Intervenciones de tipo Passenger

4. Tecnologías

Bonificaciones que afectan a los parámetros del jugador, comportamiento de sus acciones o de las condiciones de activación de éstas. Se requiere una acción planetaria completa y un coste en megagalax (ver punto 3.2.3) para poder activarlas. Una vez activadas el efecto de las tecnologías es permanente.

Las tecnologías vienen representadas por las cartas de tecnología (distinguibles por el símbolo ∞).

4.1 Parámetros del jugador

Estas tecnologías definen (o modifican) los parámetros del jugador:

4.1.1 Start Card

Actúa a la vez como una tecnología Container y Replentishment. El jugador siempre dispondrá de esta tecnología al inicio del juego.

4.1.2 Container

Almacena recursos de un mismo tipo (planeta de destino) y planeta de origen. Se requiere una acción planetaria completa para cargar o descargar un contenedor. Cuantos más Containers tenga el jugador, más veces podrá realizar la acción Obtención de Recursos concurrentemente. Inicialmente estará vacío

4.1.3 Replentishment

Al final de su turno, el jugador podrá robar tantas cartas como tecnologías Replentishment tenga (incluyendo Start Card). En ningún caso el numero de cartas en su mano será mayor que 5.

4.2 Modificadores de Acciones

Estas tecnologías modifican las acciones del jugador o las condiciones de activación de éstas

4.2.1 Fusion Drive

Por cada tecnología de este tipo que tenga el jugador, se le sumará 1 unidad de energía a todas las acciones de vuelo que ejecute. Este efecto se aplicará a posteriori (después de solucionar las tiradas adicionales provocadas por los 1)

4.2.2 Orbital Station

Por cada tecnología de tipo que tenga el jugador, se le permitirá ejecutar una acción planetaria[6] de su turno desde la órbita del planeta, en lugar de desde el propio planeta.

4.3 Acciones Especiales

Las siguientes tecnologías dan la posibilidad de ejecutar acciones especiales durante el turno del jugador.

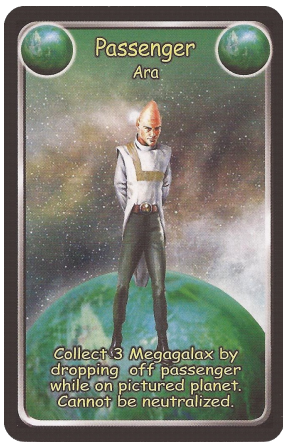
4.3.1 Hyperjump Inward/Outward

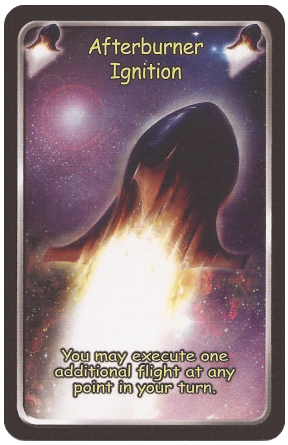
Una vez por turno, el jugador puede teletransportar su nave a la órbita del planeta más cercano/alejado al Sol. Esta acción no se considera como vuelo

6 Como Passenger no es una acción planetaria, sinó una intervención, no se podrá usar una Orbital Station para llevarlo a su planeta

5. Intervenciones

Acciones que alteran el funcionamiento normal del juego, cuyo objetivo es el de beneficiar al jugador y/o perjudicar al oponente. Un jugador puede realizar 2 intervenciones durante su turno (ver punto 3.3).





Las Intervenciones están representadas por las cartas de intervención (no tienen símbolo ∞). A diferencia de las Tecnologías, su efecto es instantáneo y son descartadas tras utilizarse. Una vez agotadas todas las cartas, se continuará el juego sin ellas

5.1. Aumentos

Intervenciones que proporcionan un beneficio directo al jugador ya sea aumentado el numero de acciones disponibles o proporcionándole megagalax

5.1.1. Passenger

Esta intervención proporciona 3 megagalax al ser utilizada en el lugar indicado. Existen 6 cartas diferentes de este tipo, una para cada planeta posible. A diferencia del resto de intervenciones, no puede ser neutralizada

Requerimientos: El jugador se encuentra sobre el planeta mostrado en el dibujo

5.1.2. Agent Infiltration

Permite ejecutar una nueva acción planetaria.

Requerimientos: Los mismos que los de las acción planetaria a emular

5.1.3. Afterburner Ignition

Permite ejecutar una nueva acción de vuelo. Esta intervención no proporciona más acciones planetarias o intervenciones

Requerimientos: Los mismos que los de vuelo

5.2. Alteración de las reglas

Estas intervenciones alteran el flujo normal del juego de manera similar a las Tecnologías. Aunque generalmente suelen buscar el beneficio del jugador, en algunos casos pueden perjudicar al oponente indirectamente.

5.2.1. Sun Tunneling

Teletransporta al jugador desde la órbita del planeta más cercano al Sol a la del más lejano utilizando 1 unidad de energía. No funciona en sentido contrario

Requerimientos:

- Durante la acción de vuelo: al menos 1 unidad de energía disponible
- La nave del jugador se encuentra en la órbita del planeta más cercano al Sol

5.2.2. Inner/Outer Loading

Permite al jugador cargar un contenedor con recursos de cualquiera de los dos planetas más cercanos/alejados al Sol. No consume una acción planetaria y puede realizarse desde cualquier lugar

Requerimientos: Los mismos que los de la acción planetaria Obtener Recursos

5.2.3. On-board Transmitter Activation

Intercambia el contenido de uno de tus contenedores con el de uno de los contenedores del oponente. Puede realizarse desde cualquier lugar

Requerimientos: Ambos contenedores deben tener al menos una carta de recurso

5.2.4. Transmitter Gate Activation

Intercambia tu posición y la posición de tu rival independientemente de dónde se encuentren.

Requerimientos: Ninguno

5.2.5. Hyperstorm Release

El jugador descarta su mano y mezcla su mazo con la pila de descarte y baraja, formando un nuevo mazo. Tras esto, el jugador roba 5 cartas del nuevo mazo y termina su turno

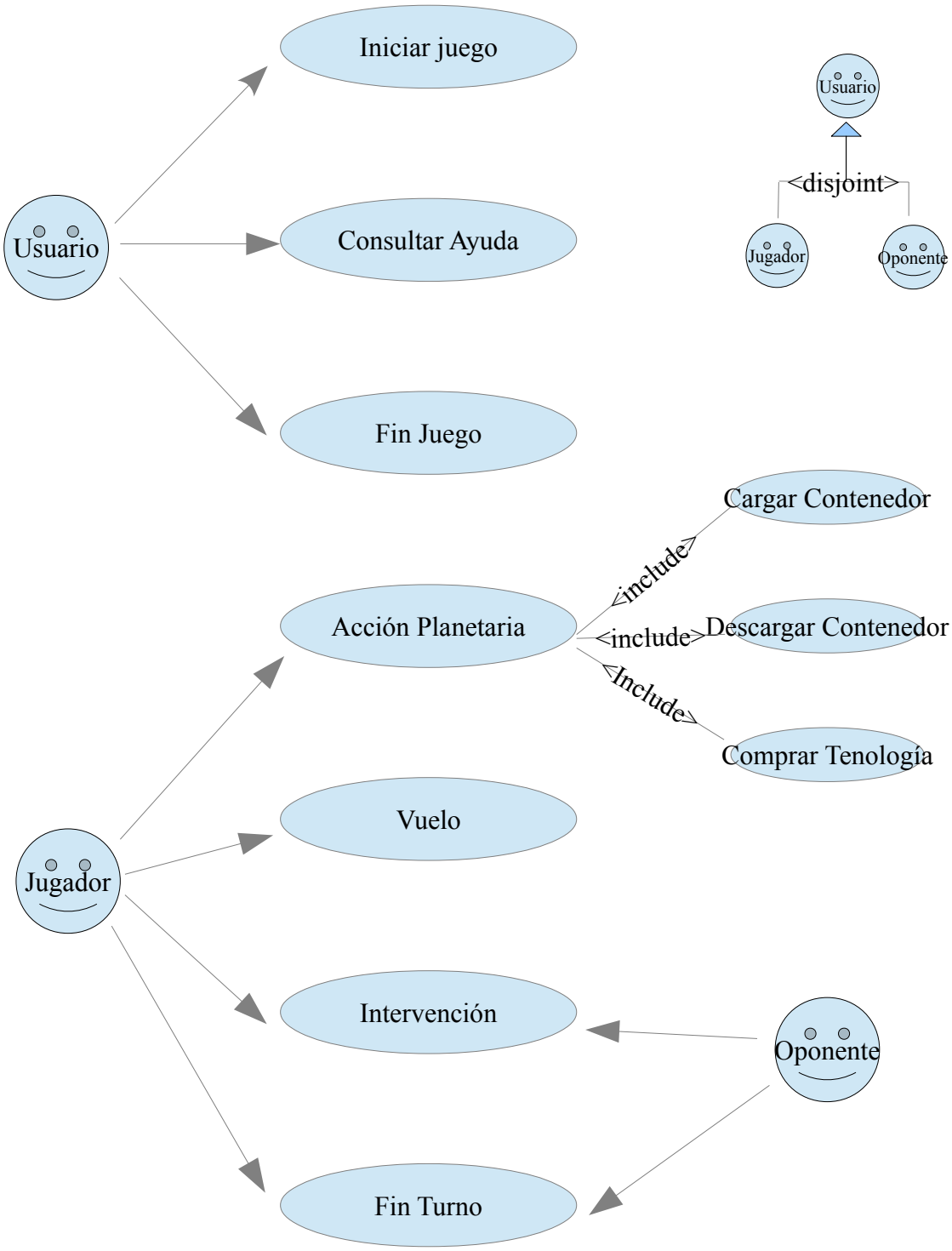
Requerimientos: Ninguno

Casos de Uso

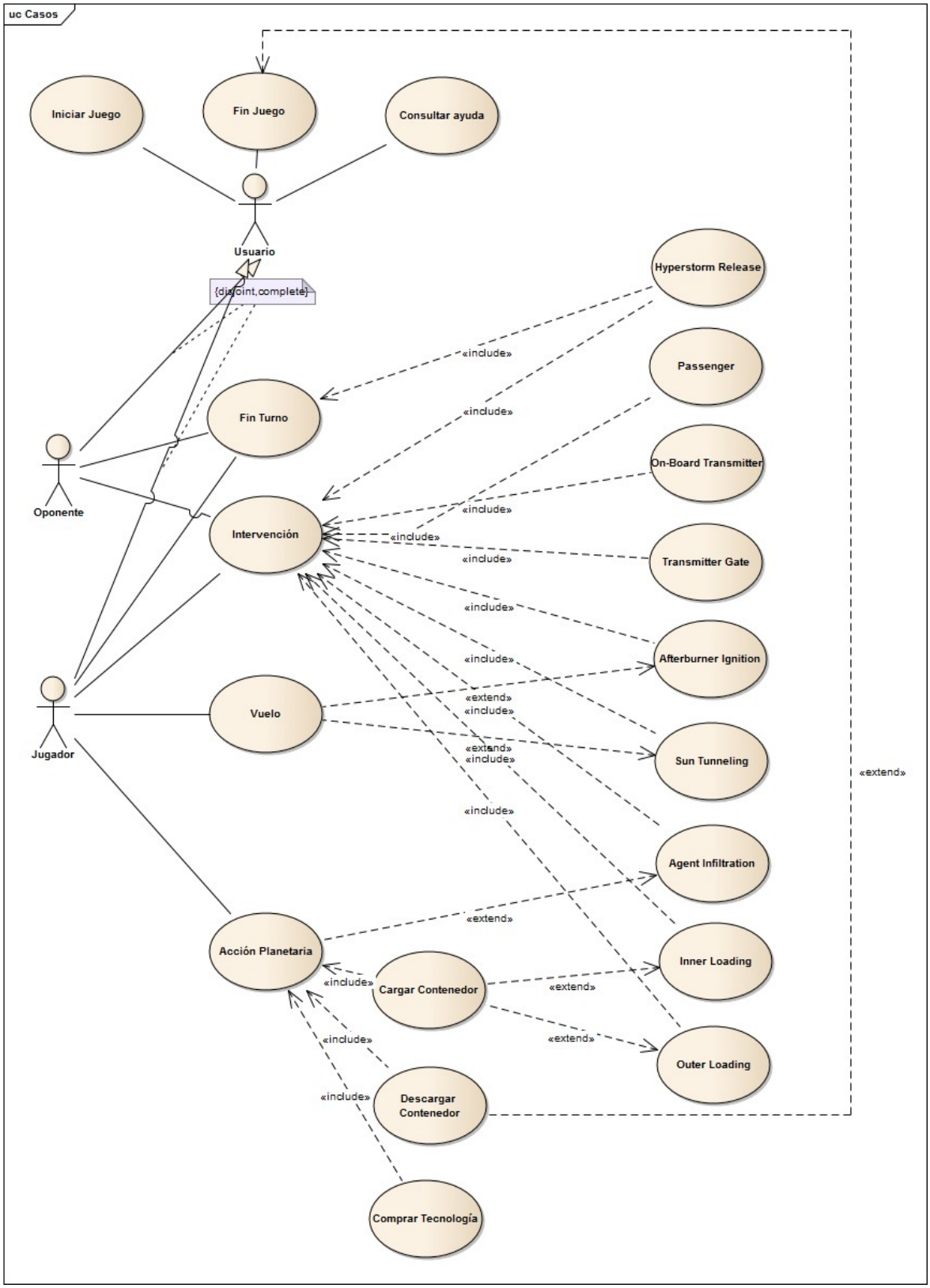
Casos de Uso....14
1. Diagrama de Casos de Uso....15
2. Iniciar Juego....17
3. Consultar Ayuda....18
4. Fin Juego....19
5. Acción Planetaria....20
5A. Cargar Contenedor....21
5B. Descargar Contenedor....22
5C. Comprar Tecnología....24
6. Vuelo....25
7. Intervención....27
7A. Passenger....29
7B. Agent Infiltration....30
7C. Afterburner Ignition....31
7D. Sun Tunneling....32
7E. Inner Loading....33
7F. Outer Loading....34
7G. On-board Transmitter....35
7H. Transmitter Gate....36
7I. Hyperstorm Release....37
8. Fin Turno....38
9. Hyperjump Inward....39
10. Hyperjump Outward....40

1. Diagrama de Casos de Uso

1.1. Simplificado:



1.2. Formal:


uc Casos
Iniciar Juego
Fin Juego
Consultar ayuda
Usuario
{disjoint,complete}
Hyperstorm Release
Passenger
On-Board Transmitter
Transmitter Gate
Afterburner Ignition
Sun Tunneling
Agent Infiltration
Inner Loading
Outer Loading
Fin Turno
Oponente
Intervención
Jugador
Vuelo
Acción Planetaria
Cargar Contenedor
Descargar Contenedor
Comprar Tecnología
«include»
«extend»

2. Iniciar Juego

Este caso de uso sería el primero que se ejecutaría en el sistema. El usuario elige los parámetros del juego que desea empezar y el sistema se encarga de realizar las inicializaciones pertinentes. Estos parámetros serán el tipo de oponente, la dificultad y el numero de puntos necesarios para ganar.

2.1. Flujo Básico

1. el usuario selecciona el tipo de juego "contra IA"
2. el usuario selecciona el tipo de IA disponible
3. el usuario introduce los parámetros del juego (puntos necesarios para ganar, tipo de IA, Avatar)

   {Inicialización}
4. el Usuario confirma su selección

2.2. Fujos Alternativos

2.2.1. Oponente Humano

1. El usuario selecciona, “contra jugador”
2. {Host} El usuario selecciona la opcion “host” y espera una conexión
3. {Client} Otro usuario selecciona la opcion “cliente” y introduce la dirección del host al que se quiere conectar
4. Se establece una conexión

2.2.2. Deshacer Inicio

En cualquier momento el Usuario puede cancelar la acción, lo que deshará todo el proceso

2.3. Precondiciones

- No hay ningún juego en curso

2.4. Postcondiciones

- El sistema inicializa el juego con los parámetros y oponente definidos por el usuario.
- Usuario pasa a ser Jugador

2.5. Puntos de Inclusión / Extensión

Ninguno

3. Consultar Ayuda

Consulta la ayuda sobre el juego. La ayuda contiene información relativa al universo de Perry Rhodan: The Cosmic League. Reglas generales del juego y explicaciones concretas de todas las cartas de éste. Este último apartado también podrá ser consultado directamente desde el juego.

3.1. Flujo Básico

1. El Usuario selecciona el apartado de ayuda
2. El Sistema muestra el índice de la ayuda
3. El Usuario navega por la ayuda
4. El Usuario consulta el apartado que le interesa
5. Repetir 3-4 hasta que el jugador cierra la ayuda

3.2. Flujos Alternativos

3.2.1. Ayuda In-game

{Durante el juego}

1. El usuario selecciona el elemento en cuestión
2. El usuario pulsa el apartado de ayuda
3. El sistema muestra al usuario la información de ayuda de ese elemento en concreto

3.3. Precondiciones

Ninguna

3.4. Postcondiciones

El usuario obtiene la información que busca. Este caso de uso no modifica el desarrollo del juego

3.5. Puntos de Inclusión

Ninguno

3.6 Puntos de Extensión

Ninguno

4. Fin Juego

Caso de uso que representa la acción de terminar juego. El juego terminará automáticamente cuando se hayan cumplido las condiciones de fin de juego.

4.1 Flujo Básico

1. El sistema determina que se han cumplido las condiciones de fin de juego (ver glosario) {Fin de juego}
2. El sistema determina el vencedor
3. El sistema anuncia el vencedor.
4. El juego termina

4.2 Flujos Alternativos

4.2.1 Empate

{Fin de juego}

1. El sistema no puede determinar ningún vencedor
2. El juego se declara en empate
3. El juego termina

4.3 Precondiciones

- Durante un juego
- Se alcanzan las condiciones de fin de juego

4.4 Postcondiciones

- El juego termina
- El jugador y oponente pasan a ser usuarios

4.5 Puntos de Inclusión

Ninguno

4.6 Puntos de Extensión

Ninguno

5. Acción Planetaria

Caso de uso que engloba a las acciones planetarias: Cargar un contenedor, Descargar un contenedor o Comprar una tecnología. Abstracto

5.1. Flujo Básico

1. El jugador indica que quiere hacer una acción planetarias
2. El jugador escoge la acción planetaria a realizar

   {Acción Planetaria}
3. El sistema descuenta de las acciones planetarias restantes del usuario

5.2. Flujos Alternativos

5.2.1. Deshacer Acción Planetaria

El jugador tiene la opción de cancelar la acción planetaria antes de concluirla, lo que deshará el todo el proceso devolviendo el juego al estado anterior

5.3. Precondiciones

- El jugador dispone al menos de una acción planetaria
- El jugador se encuentra en el planeta en cuestión (o en la órbita de este y dispone de la tecnología Orbital Station)

5.4. Postcondiciones

- El jugador dispone de una acción planetaria menos
- El juego ha sido modificado por la acción planetaria
- El jugador ha sido modificado por la acción planetaria

5.5. Puntos de Inclusión

- Cargar Contenedor (2)
- Descargar Contenedor (2)
- Comprar Tecnología (2)

5.6. Puntos de Extensión

- Agent Infiltration (3)

5A. Cargar Contenedor

Caso particular relativo a la acción planetaria de cargar un contenedor. El jugador tomará todos los recursos del tipo (planeta de destino) que él elija. Como toda acción planetaria, esto tendrá repercusión directa en el estado del juego y del jugador

5A.1. Flujo básico

{Acción planetaria}

1. El jugador selecciona una pila de recursos del planeta dónde se encuentran
2. El jugador toma los recursos seleccionados y los almacena en un contenedor libre
3. El jugador confirma la acción planetaria
4. El sistema actualiza el juego

5A.2. Flujos alternativos

Los mismos que los de Acción Planetaria

5A.3. Precondiciones

- Las mismas que las de Acción Planetaria
- El jugador dispone de un contenedor libre

5A.4. Postcondiciones

- Las mismas que las de Acción Planetaria
- La pila de recursos escogida ha sido almacenado en uno de los contenedores libres

5A.5. Puntos de Inclusión

Ninguno

5A.6. Puntos de Extensión

- Inner Loading (1)
- Outer Loading (1)

5B. Descargar Contenedor

Caso particular relativo a la acción de descargar un contenedor. Si uno de los contenedores del jugador contiene el tipo de recurso solicitado por el planeta, puede desprenderse del contenido de este obteniendo a cambio tantos megagalax como indiquen las cartas de recurso. Estos recursos pasaran a formar parte del planeta.

5B.1. Flujo básico

{Acción planetaria}

1. El jugador selecciona uno de sus contenedores que contenga el tipo solicitado de recursos
2. El jugador vacía su contenedor
3. El jugador confirma la acción planetaria y obtiene la cantidad de megagalax indicada
4. El sistema incluye en el inventario del planeta los recursos del reverso de las cartas obtenidas

{Recursos canjeados}

5B.2. Flujos alternativos

Los mismos de la acción planetaria además de los siguientes

5B.2.1. Recursos Repetidos

{Recursos canjeados}

1. Al girar las cartas, el sistema encuentra una pareja de recursos del mismo tipo
2. Si se encuentran 2 o más parejas de recursos del mismo tipo, el jugador elegirá cual de ellas se elimina.
3. El sistema elimina de la pila de recursos canjeada la pareja de recursos

5B.2.2 Bonificación de Megagalax

Si el planeta no cuenta con ningún tipo de recurso antes de descargar el contenedor, el jugador obtendrá 1 megagalax de bonificación por cada carta de recurso de la pila descargada

5B.2.3 Final de juego

Al realizar esta acción es posible que se cumplan las condiciones de fin de juego, ya sea porque se ha cubierto la quota de megagalax o porque se han eliminado todos los recursos. Es ese caso, el sistema ejecutará el caso de uso Fin juego

5B.3. Precondiciones

- Las mismas que las de Acción planetaria
- El jugador dispone de un contenedor que contiene una pila de recursos del tipo solicitado

5B.4. Postcondiciones

- Los mismos que los de Acción planetaria
- El contenedor escogido ha pasado a estar libre

5B.5. Puntos de Inclusión

Ninguno

5B.6. Puntos de Extensión

- Fin de juego

5C. Comprar Tecnología

Caso particular relativo a la compra de tecnología. El jugador activará una carta de tecnología de su mano pagando tantos megagalax como cartas de tecnología tuviera al inicio de su turno. Esta tecnología proporciona una bonificación permanente al jugador.

5C.1. Flujo básico

1. El jugador selecciona una carta de tecnología de su mano.
2. El jugador añade esta carta de tecnología a su inventario de tecnologías
3. El jugador confirma su acción
4. El sistema substrae los megagalax indicados

5C.2. Flujos alternativos

Los mismos que los de Acción Planetaria

5C.3. Precondiciones

- Las mismas que las de Acción Planetarias
- El jugador dispone almenos de los megagalax necesarios para esta transacción

5C.4. Postcondiciones

- Las mismas que las de Acción Planetaria
- Se ha substraído la carta utilizada de la mano del jugador
- Se han substraído los megagalax necesarios del haber del jugador

5C.5. Puntos de Inclusión

Ninguno

5C.6. Puntos de Extensión

Ninguno

6. Vuelo

Este caso de uso representa a la acción de vuelo. El Jugador lanzará un dado para determinar el numero de unidades de energía (rango) que tiene disponibles y moverá su nave comforme al resultado de este. El coste de de esta acción varía según la dirección del movimiento

6.1. Flujo básico

1. El jugador indica que quiere hacer un vuelo
2. El jugador lanza un dado y obtiene tantas unidades de energía como éste indique
   {Unidades de Energía disponibles}
3. El jugador suma 1 por cada tecnología de tipo Fusion Drive que tenga
4. El jugador mueve su nave a la posición deseada gastando unidades de energía en el proceso
   {Vuelo}
5. Repetir acción 4 hasta agotar todas las unidades de energía o hasta que el jugador así lo decida
6. El jugador confirma su acción
7. El sistema acualiza la posición del jugador y descuenta la acción de vuelo

6.2. Flujos alternativos

Para mantener la consistencia en el sistema, Vuelo se considera una acción atómica, por lo que no podrá deshacerse

6.2.1. Tirada de bonificación

1. El jugador ha obtenido 1 como resultado del dado
2. El jugador añade 1 al total de unidades de energía obtenidas en la siguiente tirada
3. Repetir 1-2 hasta obtener un resultado diferente de 1
4. Volver al paso 3 del flujo principal

6.3. Precondiciones

- El jugador dispone de la acción de vuelo
- El destino de la acción de vuelo se encuentra dentro del rango indicado por las unidades de energía

6.4. Postcondiciones

- El jugador no dispone de la acción de vuelo
- La posición de la nave del jugador ha pasado a ser la indicada

6.5. Puntos de Inclusión

Ninguno

6.6. Puntos de Extensión

- Sun Tunneling (4)
- Afterburner Ignition (7)

7. Intervención

Caso de uso que engloba a todas las intervenciones. Las intervenciones son acciones que alteran el flujo normal del juego, por lo que la mayoría de ellas serán extensiones de los casos de uso anteriormente descritos. Abstracta

7.1. Flujo básico

1. El jugador indica que quiere realizar una intervención
2. El jugador juega una carta de intervención de su mano

   {Intervención}
3. El jugador descarta la intervención
4. El sistema actualiza el estado y descuenta la acción de intervención

7.2. Flujos alternativos

7.2.1. Cancelar Intervención

{Intervención}

1. El oponente juega la misma carta de intervención de su mano
2. No se realiza la acción de intervención
3. El oponente descarta la carta de intervención
4. Volver a 4 del flujo básico

7.2.2. Deshacer Intervención

En cualquier momento y siempre que la intervención no haya sido cancelada el jugador puede cancelar su acción, deshaciendo toda modificación que se haya podido hacer al estado

7.3. Precondiciones

- El jugador dispone al menos de una acción de intervención restante
- El jugador dispone de la carta de intervención a jugar
- {Cancelar intervención} El oponente dispone de la misma carta de intervención a jugar

7.4. Postcondiciones

- El jugador no dispone de la carta de intervención utilizada
- El jugador cuenta con una acción de intervención menos
- {Cancelar intervención} El oponente no dispone de la carta de intervención

7.5 Puntos de Inclusión

- Passenger (2)
- Agent Infiltration (2)
- Afterburner Ignition (2)
- Sun Tunneling (2)
- Inner Loading (2)
- Outer Loading (2)
- On-boar Transmitter (2)
- Transmitter Gate (2)
- Hyperstorm release (2)

7.6 Puntos de Extensión

Ninguno

7A. Passenger

Caso particular relativo a las intervenciones de tipo pasajero. El jugador obtiene 3 megagalax si juega esta intervención mientras se encuentra en el planeta mostrado en la carta. A diferencia del resto de intervenciones no puede ser cancelada

7A.1. Flujo Básico

{Intervención}

1. El jugador jugador selecciona la carta de tipo pasajero correspondiente al planeta donde se encuentran
2. El jugador obtiene 3 megagalax
3. El jugador descarta la carta pasajero utilizada

7A.2. Flujos Alternativos

Los mismos que los de Intervención excepto Cancelar Intervención

7A.1. Cuota cubierta

Si el jugador alcanza la cuota de 70 megagalax durante esta acción, se ejecutará el caso de uso Fin Juego

7A.3. Precondiciones

- Las mismas que las de Intervención
- El jugador se encuentra directamente sobre el planeta de destino de la carta pasajero

7A.4. Postcondiciones

- Las mismas que las de Intervención
- El jugador obtiene 3 megagalax

7A.5. Puntos de Inclusión

Ninguno

7A.6. Puntos de Extensión

- Fin Juego

7B. Agent Infiltration

Caso particular relativo a la intervención agent infiltration. El jugador realiza una acción planetaria. La única diferencia, es que esta acción planetaria no se descontará de la cuota de acciones planetarias del presente turno.

7B.1. Flujo Básico

{Intervención}

1. El jugador selecciona la carta Agent Infiltration
2. El jugador escoge una acción planetaria

   {Acción planetaria}
3. El jugador descarta la carta Agent Infiltration utilizada.

7B.2. Flujos Alternativos

Los mismos que los de Intervención, además de los de la propia acción planetaria

7B.3. Precondiciones

- Los mismos que los de intervención
- Los mismos que los de la acción planetaria elegida

7B.4. Postcondiciones

- Los mismos que los de intervención
- Los mismos que los de la acción planetaria elegida

7B.5. Puntos de Inclusión

Ninguno

7B.6. Puntos de Extensión

Ninguno

7C. Afterburner Ignition

Caso particular relativo a la intervención Afterburner Ignition. El jugador realiza una acción de vuelo adicional a la del presente turno (no se descuenta). Funciona exactamente igual que el caso de uso Vuelo.

7C.1. Flujo básico

{Intervención}

1. El jugador selecciona la carta Afterburner Ignition
2. El jugador inicia una acción de vuelo

   {Vuelo}
3. El jugador descarta la carta Afterburner Ignition utilizada

7C.2. Flujos alternativos

Los mismos que los de Intervención, además de los de Vuelo

7C.3. Precondiciones

- Las mismas que las de Intervención
- Las mismas que las de Vuelo

7C.4. Postcondiciones

- Las mismas que las de Intervención
- Las mismas que las de Vuelo

7C.5. Puntos de Inclusión

Ninguno

7C.6. Puntos de Extensión

Ninguno

7D. Sun Tunneling

Caso particular relativo a la intervención de Sun Tunneling. El jugador puede transportarse desde la órbita del planeta más cercano al Sol (Coronis) a la del más lejano (Dahkit) usando una unidad de energía. A causa de esto, este Caso también será considerado una extensión de Vuelo. Por motivos de coherencia, tras realizar esta acción, no se podrá deshacer o cancelar el vuelo

7D.1. Flujo Básico

{Vuelo}

1. El jugador juega la carta Sun Tunneling
2. El jugador mueve su nave a la órbita de Dahkit
3. El jugador descarta la carta Sun Tunneling utilizada

{Vuelo}

7D.2. Flujos Alternativos

Los mismos que los de Intervención

7D.3. Precondiciones

- Las mismas que las de Intervención
- Durante un vuelo
- El jugador dispone de al menos una unidad de energía
- La nave del jugador se encuentra en la órbita de Coronis

7D.4. Postcondiciones

- Las mismas que las de Intervención
- El jugador dispone de una unidad de energía menos
- La nave del jugador se encuentra en la órbita de Dahkit

7D.5. Puntos de Inclusión

Ninguno

7D.6. Puntos de Extensión

Ninguno

7E. Inner Loading

Caso particular relativo a la intervención Inner Loading. El jugador carga una pila de recursos de cualquiera de los dos planetas más interiores (Coronis, Naginther). Esta intervención funciona exactamente igual que Cargar Contenedor, pero pudiéndose realizar desde cualquier lugar y sin consumir una acción planetaria.

7E.1. Flujo básico

{Intervención}

1. El jugador juega la carta Inner Loading
2. El jugador selecciona uno de los dos planetas

   {Cargar Contenedor}
3. El jugador descarta la carta Inner Loading utilizada

7E.2. Flujos alternativos

Los mismos que los de Intervención, además de los de Cargar Contenedor

7E.3. Precondiciones

- Las mismas que las de Intervención
- El jugador dispone almenos de un contenedor libre

7E.4. Postcondiciones

- Las mismas que las de Intervención
- Las mismas que las de Cargar Contenedor

7E.5. Puntos de Inclusión

Ninguno

7E.6. Puntos de Extensión

Ninguno

7F. Outer Loading

Caso particular relativo a la intervención Outer Loading. Funciona exactamente igual que inner loading pero con los dos planetas más exteriores (Talmazzar, Dahkit)

7F.1. Flujo básico

{Intervención}

4. El jugador juega la carta Outer Loading
5. El jugador selecciona uno de los dos planetas

   {Cargar Contenedor}
6. El jugador descarta la carta Outer Loading utilizada

7F.2. Flujos alternativos

Los mismos que los de Intervención, además de los de Cargar Contenedor

7F.3. Precondiciones

- Las mismas que las de Intervención
- El jugador dispone al menos de un contenedor libre

7F.4. Postcondiciones

- Las mismas que las de Intervención
- Las mismas que las de Cargar Contenedor

7F.5. Puntos de Inclusión

Ninguno

7F.6. Puntos de Extensión

Ninguno

7G. On-board Transmitter

Caso particular relativo a la intervención On-board Transmitter. El jugador intercambiará el contenido de uno de sus contenedores por el de uno de los del oponente. Esta intervención perjudica directamente al oponente

7G.1. Flujo Básico

{Intervención}

1. El jugador juega la carta On-board Transmitter Activation
2. El jugador selecciona uno de los contenedores del oponente
3. El jugador intercambia las pila de recursos de uno de sus contenedores por la del contenedor elegido del oponente
4. El jugador descarta la carta On-board Transmitter Activation utilizada

7G.2. Flujos Alternativos

Los mismos que los de Intervención

7G.3. Precondiciones

- Las mismas que las de Intervención
- Ambos contenedores contienen una pila de recursos (no están vacíos)

7G.4. Postcondiciones

- Las mismas que las de Intervención
- Se han intercambiado las pilas de recursos de los contenedores

7G.5. Puntos de Inclusión

Ninguno

7G.5. Puntos de Extensión

Ninguno

7H. Transmitter Gate

Caso particular relativo a la intervención Transmitter Gate Activation. El jugador intercambiará la posición de su nave con la posición de la nave del oponente. Es irrelevante dónde se encuentran las naves en este momento.

7H.1. Flujo Básico

{Intervención}

1. El jugador juega la carta Transmitter Gate Activation
2. El jugador intercambia la posición de la nave del oponente con la suya
3. El jugador descarta la carta Transmitter Gate Activation utilizada

7H.2. Flujos Alternativos

Los mismos que los de Intervención

7H.3. Precondiciones

- Las mismas que las de Intervención

7H.4. Postcondiciones

- Las mismas que las de Intervención
- La posición de las naves de ambos usuarios ha sido intercambiada

7H.5. Puntos de Inclusión

Ninguno

7H.6. Puntos de Extensión

Ninguno

7I. Hyperstorm Release

Caso particular relativo a la intervención Hyperstorm release. El jugador descarta su mano, une su mazo y la pila de descarte para obtener un nuevo mazo. Esta intervención proporciona la manera de restaurar el mazo de robo una vez ha sido descartado. Esta acción forzará el fin de turno del jugador

7I.1. Flujo básico

{Intervención}

1. El jugador juega la carta Hyperstorm Release
2. El jugador descarta su mano entera incluyendo la carta Hyperstorm Release
3. El jugador junta y baraja la pilas de descarte y el mazo de robo
4. El jugador roba 5 cartas del nuevo mazo de robo

{Fin turno}

7I.2. Flujos Alternativos

Los mismos que los de Intervención

7I.3. Precondiciones

- Las mismas que las de Intervención

7I.4. Postcondiciones

- Las mismas que las de Intervención
- La pila de descarte del jugador pasa a formar parte del mazo de robo
- El jugador cambia su mano por 5 nuevas cartas
- El jugador Finaliza su Turno

7I.5. Puntos de inclusión

- Fin Turno (4)

7I.6. Puntos de extensión

Ninguno

8. Fin Turno

Este caso contempla los eventos del final del turno del jugador. El jugador obtendrá nuevas cartas dependiendo del número de tecnologías replentishment que tenga. Tras esto, el jugador pasará a ser oponente, y el oponente jugador

8.1. Flujo Básico

1. El jugador anuncia que quiere acabar su turno
2. El jugador obtiene tantas cartas nuevas como tecnologías de tipo Replentishment tenga (hasta un máximo de 5)

   {Fin Turno}
3. El jugador pasa a ser oponente
4. El nuevo jugador obtiene sus correspondiente vuelo, acciones planetarias e intervenciones

8.2. Flujos Alternativos

8.2.1 Mazo Vacío

Si el usuario no tiene cartas en su mazo, se omite el tercer paso

8.3. Precondiciones

- Durante el turno del jugador

8.4. Postcondiciones

- El jugador pasa a ser oponente y viceversa
- Si su mazo lo permite, el jugador obtiene tantas cartas como tecnologías Replentishment tenga

8.5. Puntos de Inclusión

Ninguno

8.6. Puntos de Extensión

Ninguno

9. Hyperjump Inward

Este caso de uso hace referencia a la acción especial de la tecnología Hyperjump Inward. Dicha tecnología permite al jugador hacer un salto a la órbita planeta mas interior (Coronis) al principio de su turno. Esta tecnología no tiene ningún coste.

9.1. Flujo Básico

1. El jugador anuncia que quiere utilizar su tecnología Hyperjump Inward
2. El jugador desplaza su nave a la órbita de Coronis

9.2. Flujos Alternativos

Dada la atomicidad de esta caso de uso no tiene demasiado sentido definir una acción de deshacer

9.3. Precondiciones

- El jugador dispone de la tecnología Hyperjump Inward
- No se puede utilizar tras haber realizado cualquiera de las siguientes acciónes: Acción planetarias, Vuelo, Intervención, Hyperjump Inward o Hyperjump Outward

9.4. Postcondiciones

- El jugador mueve su nave a la órbita de Coronis

9.5. Puntos de Inclusión

Ninguno

9.6. Puntos de Extensión

Ninguno

10. Hyperjump Outward

Este caso de uso hace referencia a la acción especial de la tecnología Hyperjump Outward. Funciona exactamente igual que Hyperjump Inward, sólo que tiene como destino la órbita del planeta más exterior (Dahkit)

9.1. Flujo Básico

3. El jugador anuncia que quiere utilizar su tecnología Hyperjump Outward
4. El jugador desplaza su nave a la órbita de Dahkit

9.2. Flujos Alternativos

Dada la atomicidad de esta caso de uso no tiene demasiado sentido definir una acción de deshacer

9.3. Precondiciones

- El jugador dispone de la tecnología Hyperjump Outward
- No se puede utilizar tras haber realizado cualquiera de las siguientes acciones: Acción planetarias, Vuelo, Intervención, Hyperjump Inward o Hyperjump Outward

9.4. Postcondiciones

- El jugador mueve su nave a la órbita de Dahkit

9.5. Puntos de Inclusión

Ninguno

9.6. Puntos de Extensión

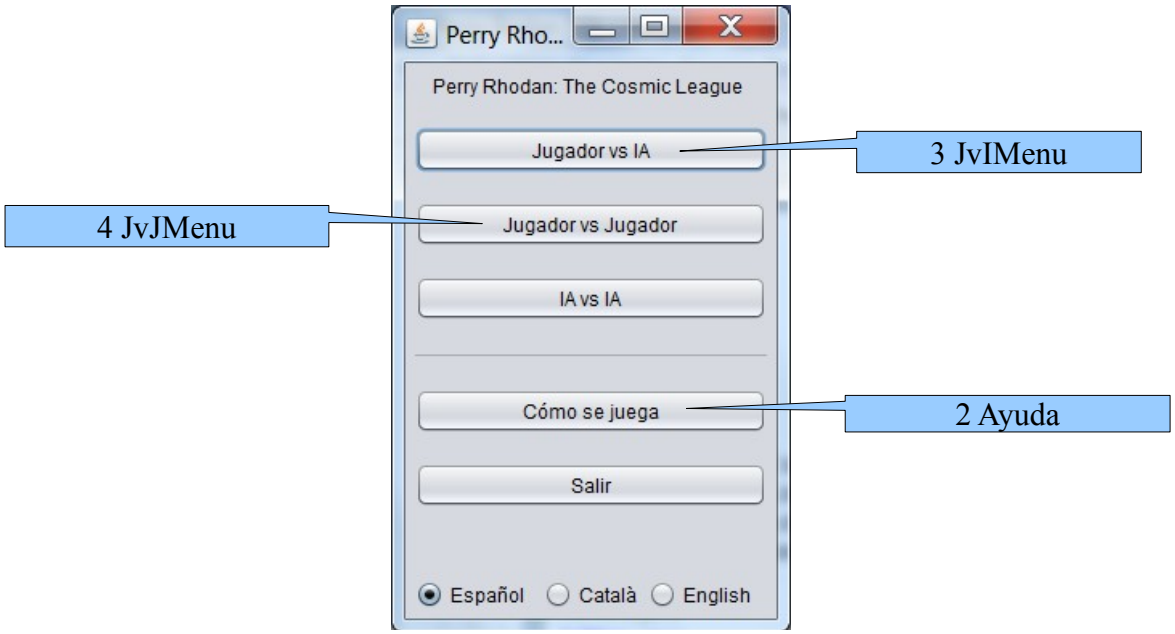
Ninguno

Diseño de Interficie Gráfica

Diseño de Interficie Gráfica....41
1. MMenu (Menú principal)....42
2. Ayuda....43
3. JvIMenu....44
4. JvJMenu....45
5. GUI_main....46
5.1. Panel Lateral....47
5.2. Vuelo....48
5.3 Log....49
6. GUI_AccPlan....50
6. GUI_SwapContainers....51
7. GUI_DoubleLoading....52
8. Más información....53

1. MMenu (Menú principal)

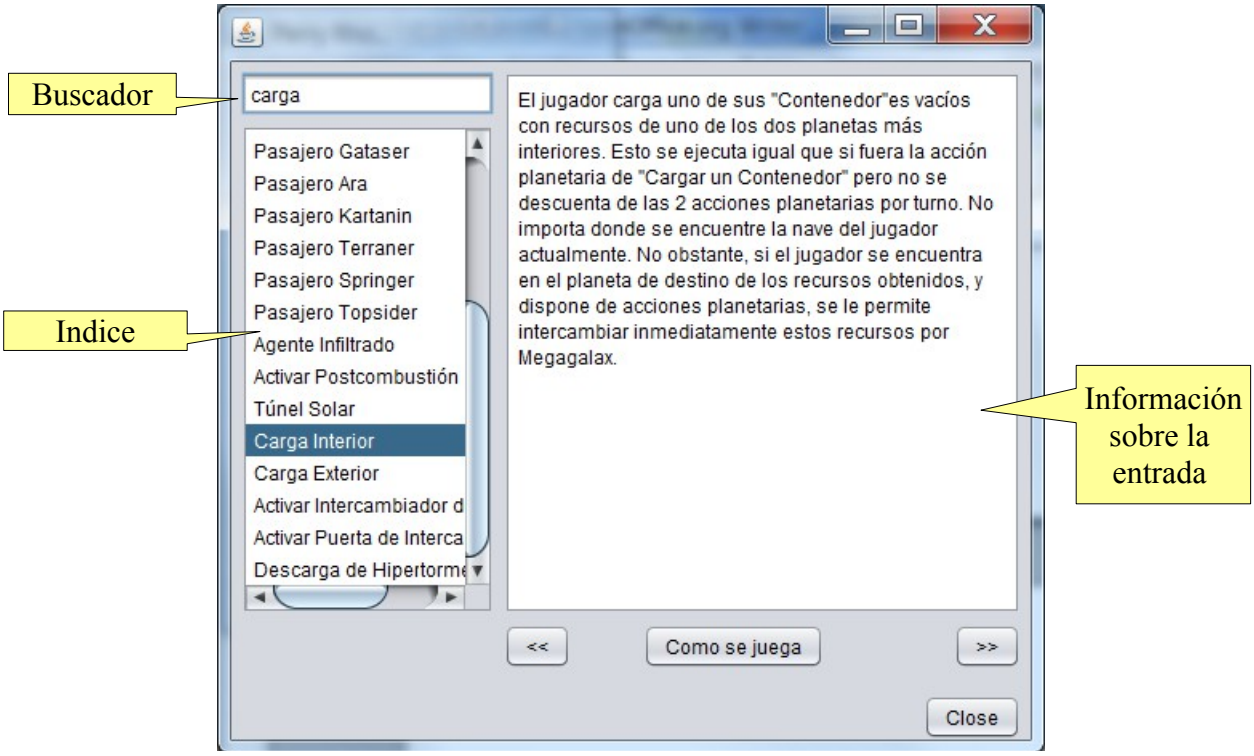
Auto-explicativo. Interficie simple y compacta donde el usuario puede seleccionar cualquiera de las opciones del caso de uso 2 (Iniciar juego). Adicionalmente, permite seleccionar el idioma que se utilizará durante el resto de la aplicación



También ofrece la posibilidad de consultar la ayuda (Caso de uso 3)

2. Ayuda

Interficie que cubre todas las funciones especificadas en el caso de uso 3 Ayuda. Esta interficie se podrá consultar tanto desde MMenu como GUI_main (ayuda in-game).



La ayuda mostrada, estará en el mismo idioma que el de la interficie (Español/Catalán/Inglés)

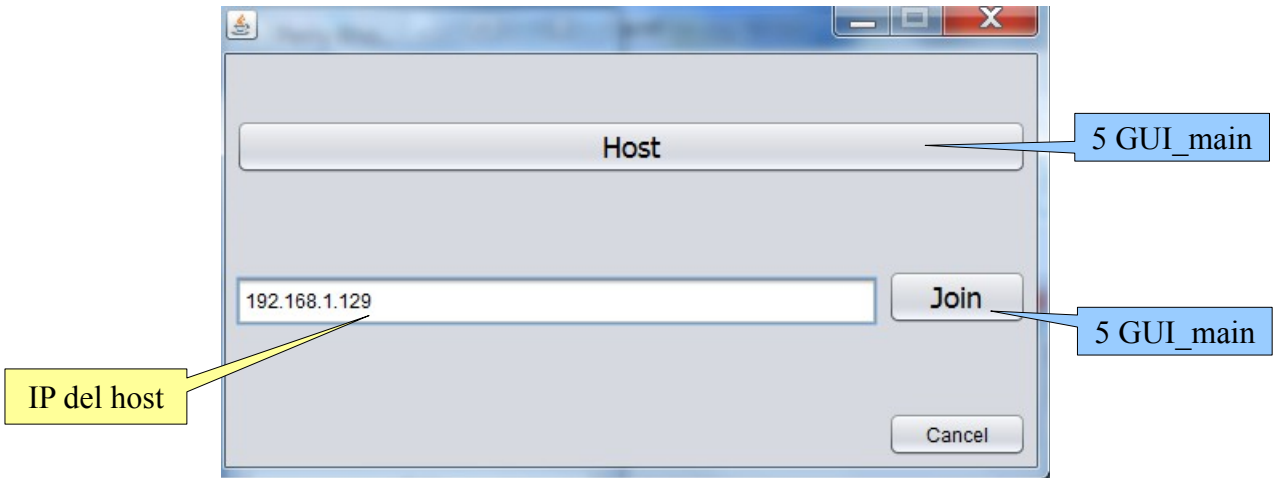
3. JvIMenu

Interficie que cubre las funciones especificadas en el flujo básico del caso de uso Iniciar Juego. El usuario selecciona los parámetros del juego, y el sistema inicializa un juego y un oponente según estos parámetros

Al igual que el resto, esta interficie también estará en el idioma seleccionado en MMenu.

4. JvJMenu

Interficie que cubre las funciones especificadas en el flujo alternativo 2.2.1 (Oponente Humano) del caso de uso Iniciar juego. Inicia un juego estándar de 70 megagalax asignándole el primer jugador al host.



Tras iniciar un juego, el host quedará bloqueado a la espera de un oponente que se una.

5. GUI_main

Interfaz principal sobre la que se desarrollará la mayor parte del juego. Proporciona al usuario la información necesaria sobre el estado del juego a la vez que cubre los casos de uso 5C (Comprar Tecnología), 6 (Vuelo), 7 (Intervención) y 8 (Fin Turno)

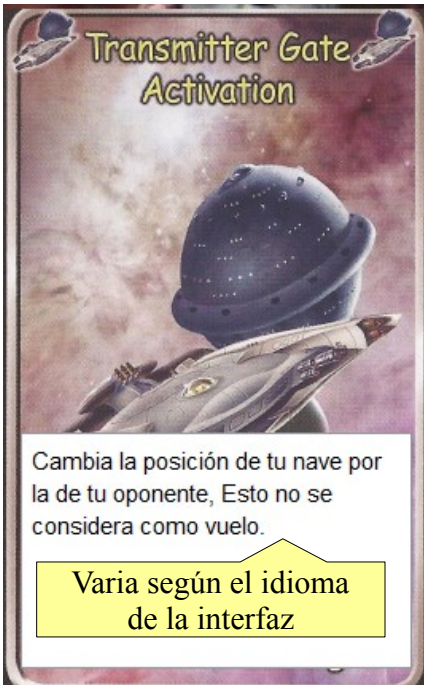
Para evitar problemas de consistencia, la interficie bloqueará todos los elementos que no pueda utilizar el usuario en ese momento[7]. Esto también implica que quedará bloqueada en su totalidad siempre que no sea el turno del jugador

7 Operaciones consultoras: checkPosible y testPlanetaria

5.1. Panel Lateral

Con el fin de optimizar el espacio ocupado por el interfaz y minimizar la información redundante, se ha dispuesto un panel multifunción en la parte lateral. Dependiendo de la posición del cursor, este panel mostrará una información diferente.

Estado del Jugador
(cursor neutro)



Descripción de la carta
(cursor sobre una carta)

Inventario del planeta
(cursor sobre un planeta)

Este panel no cubre ningún caso de uso en concreto, únicamente muestra información concreta de los elementos del juego

5.2. Vuelo

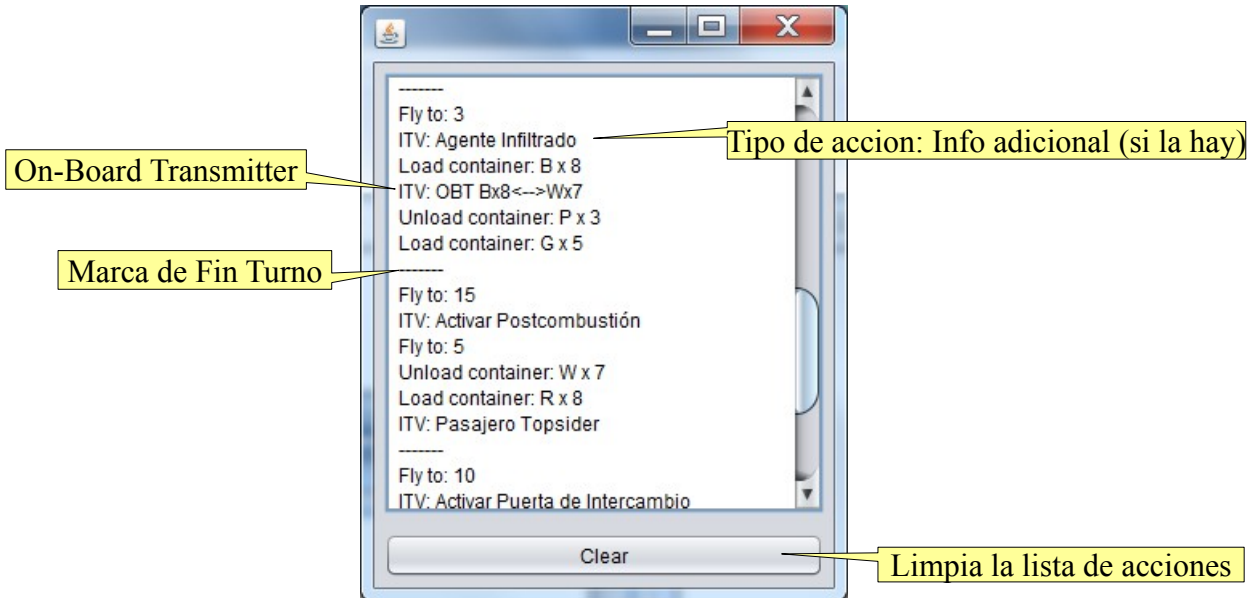
Al activar la ejecución del Caso de uso 6 (Vuelo), la interfaz activará una serie de radioButtons para limitar el rango de vuelo según las unidades de energía disponibles (NRG). Esto se hace en primer lugar para facilitar la comprensión del rango de vuelo por parte del usuario, así como para seguir la norma de limitar las acciones del usuario, a aquellas que realmente pueda hacer en ese momento.

Durante la ejecución de este caso de uso, el sistema bloqueará el resto de la interfaz[8] (Cartas, Acciones Planetarias, Fin de turno).

8 Excepción: Intervención Sun Tunneling

5.3 Log

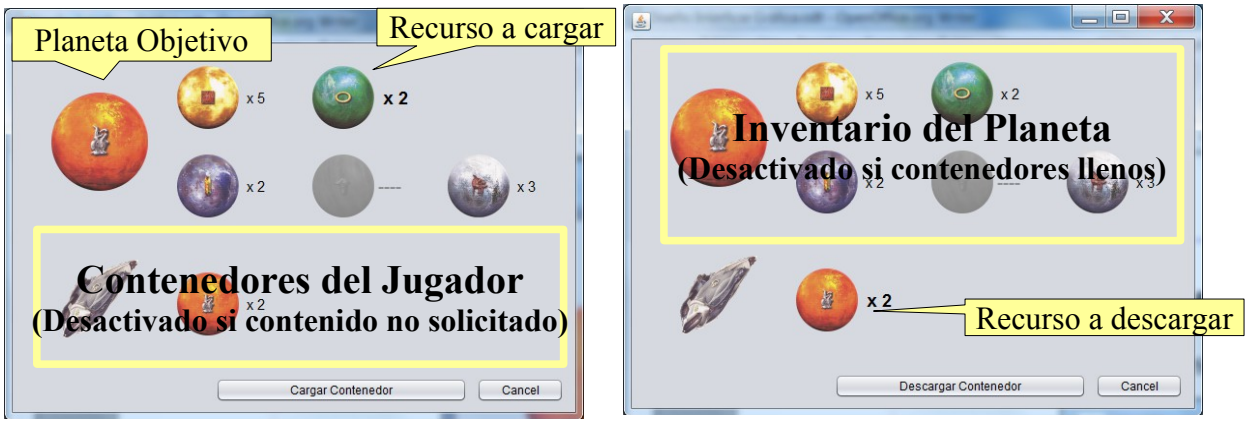
Interficie auxiliar que muestra al Jugador las acciones del oponente (ya sea Humano o IA) durante su turno. El componente Observer es el encargado de actualizar esta interfaz.



Inicialmente utilizada para propósitos de testing y debugging, se ha decidido mantener ya que también proporciona al usuario una visión de las acciones de su oponente

6. GUI_AccPlan

Interficie que cubre las funciones especificadas en los casos de uso 5A (Cargar Contenedor) y 5B (Descargar Contenedor). El jugador podrá elegir entre un recurso del inventario del planeta para efectuar una acción de carga, o uno de sus contenedores para efectuar una descarga. Dependiendo del estado del juego, alguna (o ambas) de estas dos opciones podría estar desactivada



Tal y como se especifica en el flujo alternativo 5.2.1 (Cancelar acción planetaria), el jugador también tendrá la opción de retractarse de la acción planetaria sin ningún coste

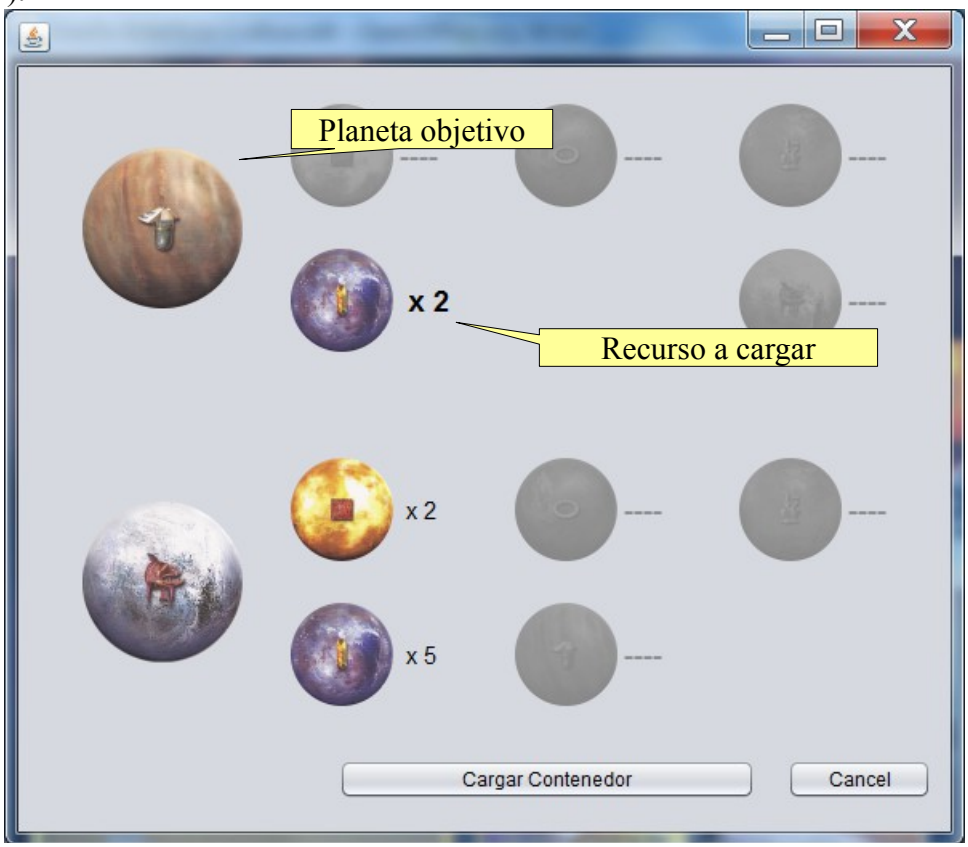
6. GUI_SwapContainers

Esta interficie cubre las funciones especificadas en el caso de uso 7G (On-Board Transmitter). Se activa al seleccionar dicha intervención, y funciona de manera similar a la interfaz GUI_AccPlan, salvo que en este caso, se tendrán que escoger un contenedor de cada tipo.

Básicamente proporciona al usuario una manera dinámica de escoger los contenedores sobre los cuales se efectuará esta intervención

7. GUI_DoubleLoading

Interficie que cubre las funciones los casos de uso 7E y 7F (Inner/Outer Loading). Se activa al seleccionar dicha intervención, y funciona de manera similar a la interfaz GUI_AccPlan, salvo que en este caso hay 2 planetas sobre los que escoger (los 2 mas cercanos/lejanos dependiendo de la intervención).



8. Más información

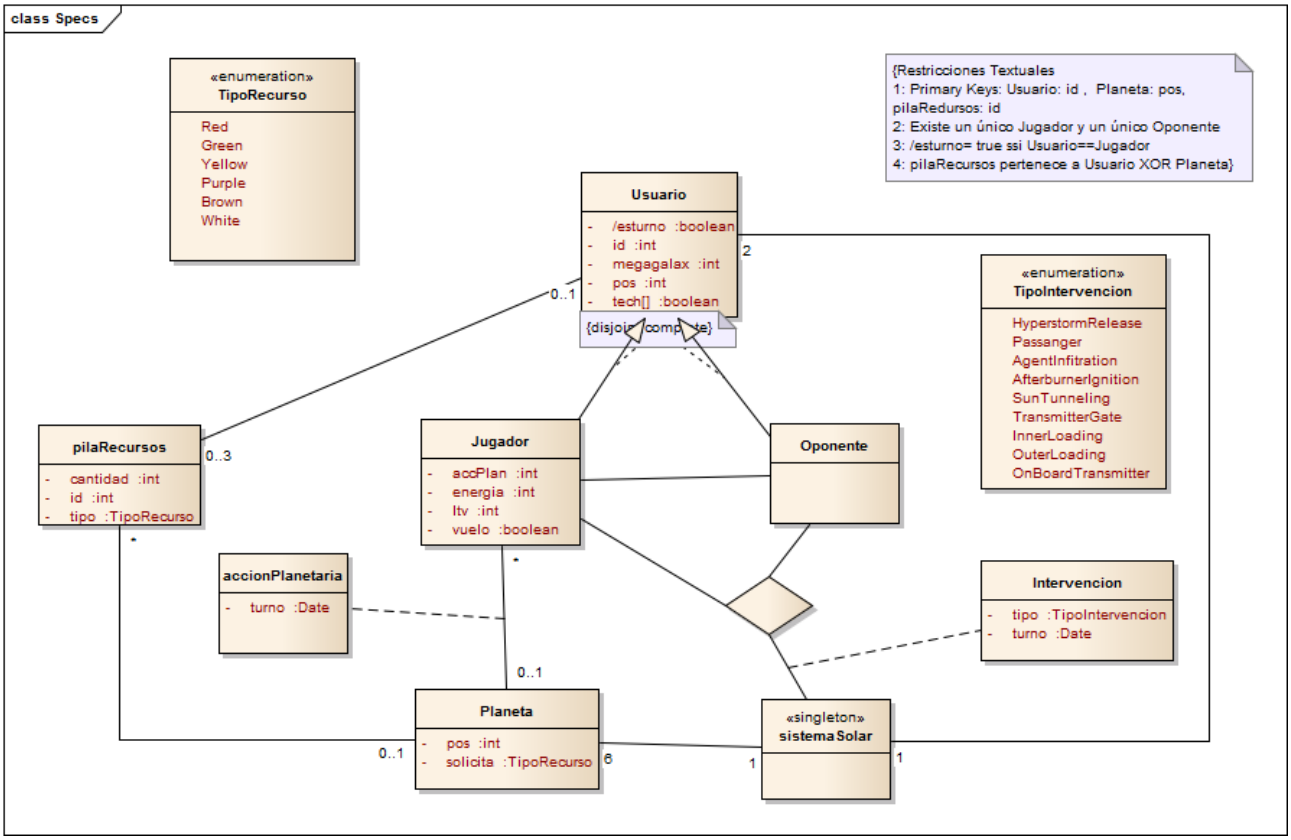
Puede ver una demostración del comportamiento de la interficie gráfica en diferentes situaciones, consultando los siguientes enlaces:

- **Jugador vs Jugador:** http://youtu.be/JHJ0tBjYh6E
- **Jugador vs IA_Random (Inglés):** http://youtu.be/p6hmaJUSyAE
- **Jugador vs IA_Greedy (Catalán):** http://youtu.be/8WP7Z4dcoo4
- **Jugador vs IA_Safe (Castellano):** http://youtu.be/29IokPPbCU4

Especificación

Especificación....54
1. Diagrama de Clases....55
2. Iniciar Juego....56
3. Consultar Ayuda....57
4. Fin Juego....58
5. Acción Planetaria....59
6. Vuelo....62
7. Intervención....64
8. Fin Turno....71

1. Diagrama de Clases



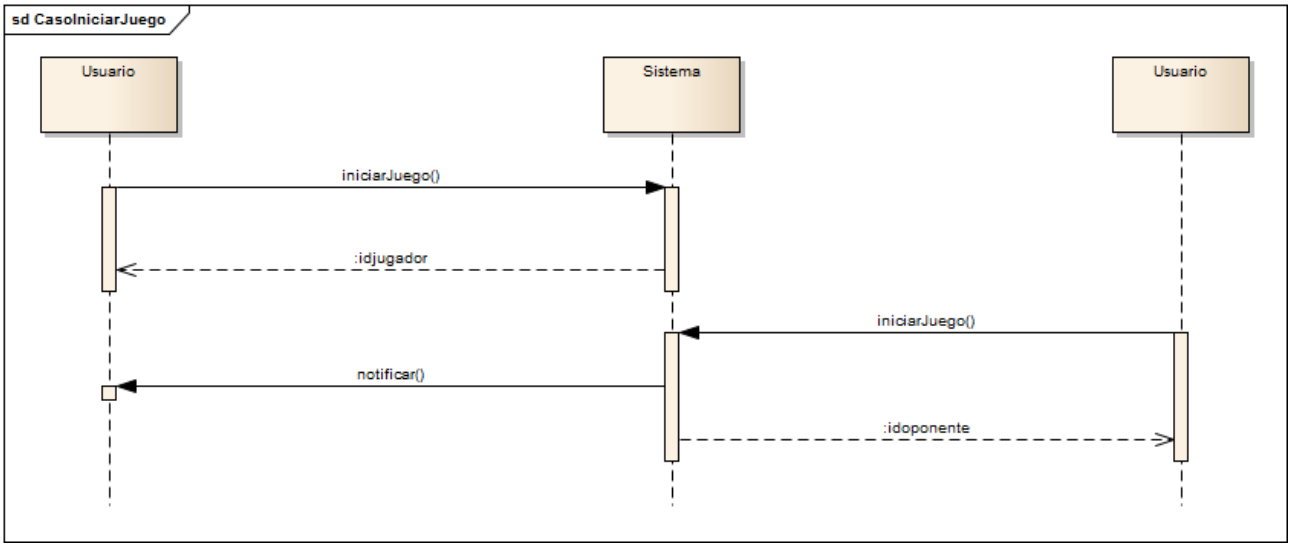
Como bien habíamos dicho en el informe preliminar, optamos por una arquitectura de 3 capas para este proyecto. En este artefacto se describe el funcionamiento de la capa lógica (Dominio)

Para evitar redundancia entre este artefacto el documento Diseño, las precondiciones han sido etiquetadas en el siguiente formato

Etiqueta: descripción de la precondición (Asignación de responsabilidades a capas)

En el documento diseño se seguirán utilizando estas etiquetas, o su conjugada negativa en el caso de las excepciones, obviando la descripción la cual seguirá siendo la misma

2. Iniciar Juego



Context iniciarJuego(): idjugador

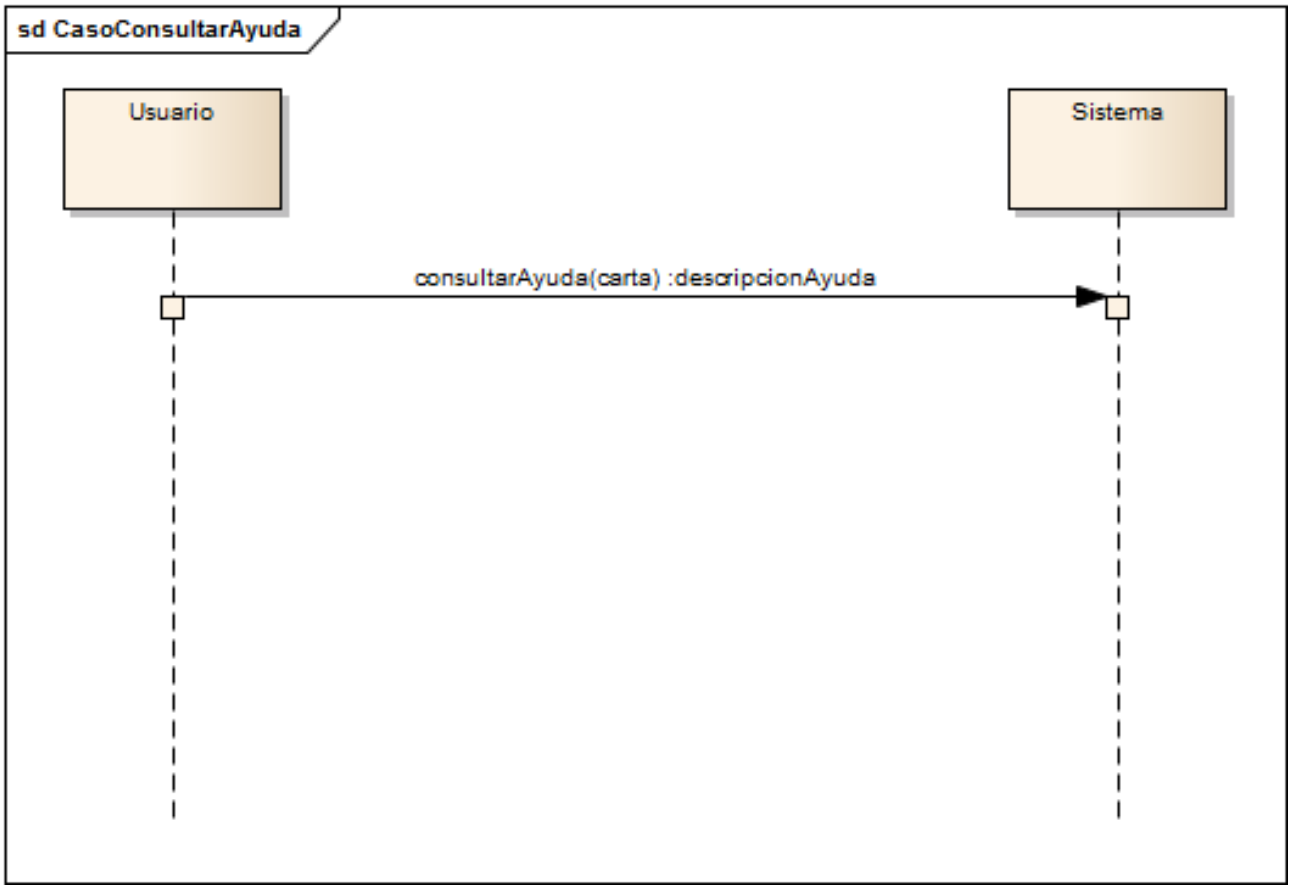
Pre:

- Init: Sistema inicializado *(ARC: Presentación)*

Post:

- Si otro usuario ha ejecutado anteriormente este caso:
  - Usuario pasa a ser Oponente *(ARC: Dominio)*
  - Juego iniciado según los parámetros establecidos en el sistema *(ARC: Presentación+Dominio+Datos)*
- Si no:
  - Usuario pasa a ser Jugador *(ARC: Dominio)*
  - Sistema bloqueado a la espera de un oponente *(ARC: Presentación+Dominio)*

3. Consultar Ayuda



Context ConsultarAyuda(carta:entero): descripciónAyuda

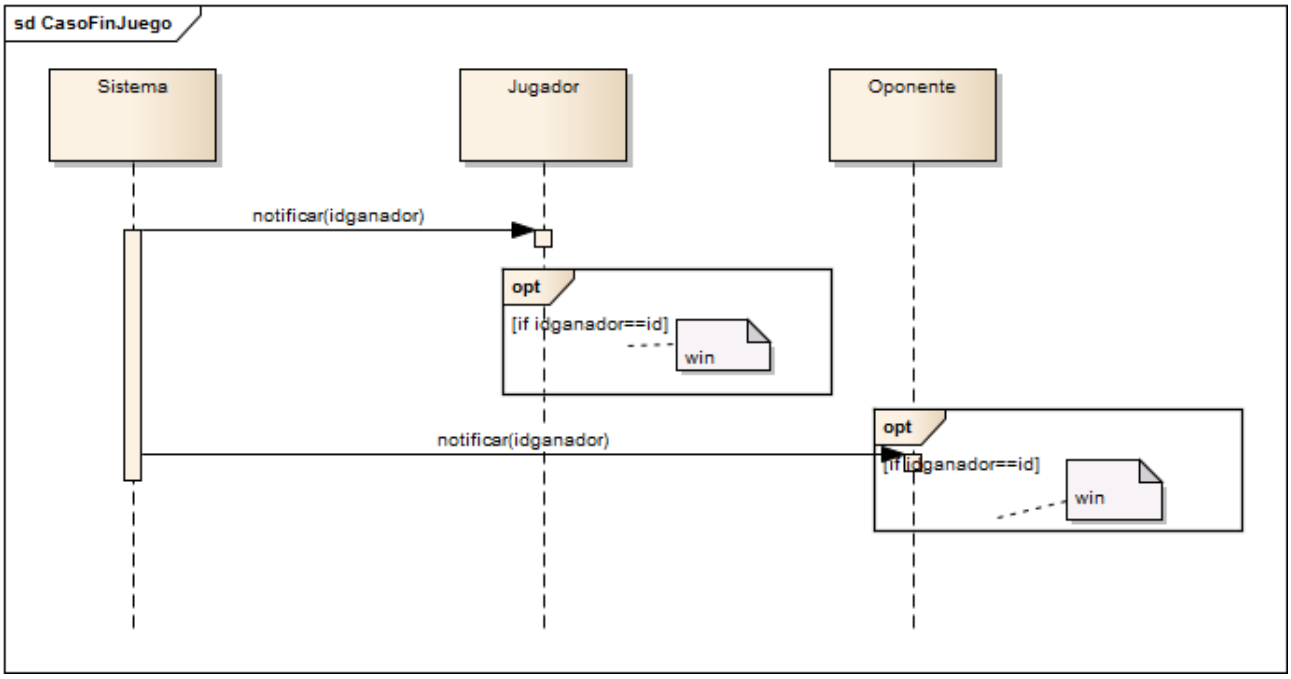
Pre:

- CartaValida: carta es un índice de carta válido *(ARC: Presentación)*

Post:

- descripciónAyuda = Ayuda referente a la carta *(ARC: Dominio+Datos)*

4. Fin Juego



Context FinJuego()

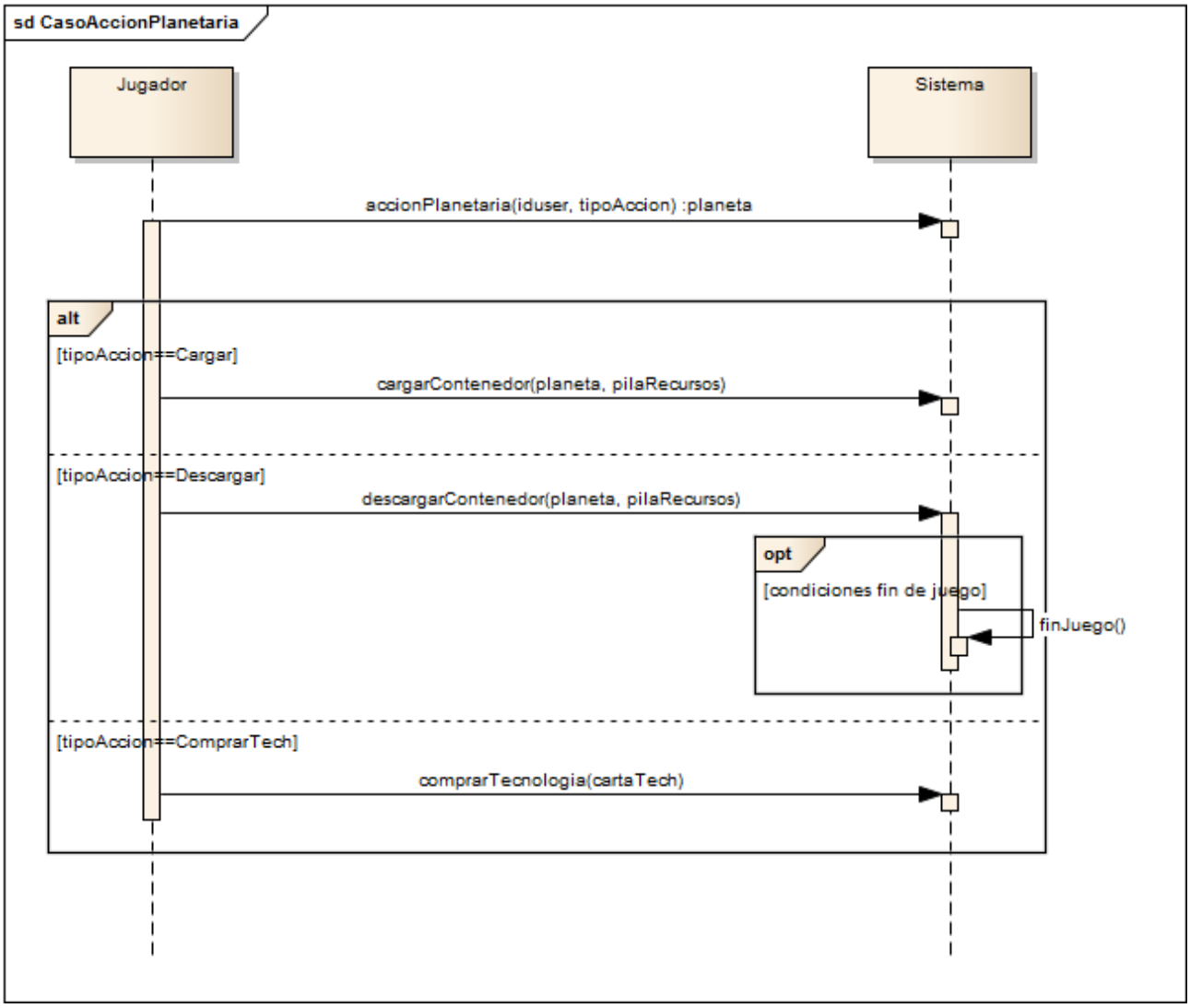
Pre:

- Fin: Se han detectado las condiciones de fin de juego *(ARC: Dominio)*

Post:

- El juego finaliza *(ARC: Dominio+Datos)*
- Se notifica a los usuarios el resultado del juego *(ARC: Presentación)*

5. Acción Planetaria



Context accionPlanetaria(iduser: entero, TipoAccion:String):idplaneta

Pre:

- Jugador: el usuario iduser es de tipo Jugador (es su turno) *(ARC:Dominio)*
- AccionPlanetaria: El jugador iduser dispone de al menos una accion planetaria *(ARC: Dominio)*
- PosCorrecta: El jugador iduser se encuentra sobre un planeta o el jugador se encuentra sobre la órbita de un planeta y dispone de la tecnología Orbital Station *(ARC: Dominio)*

Post:

- idplaneta es identificador del planeta sobre el que el jugador va a efectuar la acción planetaria *(ARC: Dominio)*
- Se inicia la acción planetaria de tipo TipoAcción *(ARC: Presentación)*

Context cargarContenedor(idplaneta:entero, pilaRecursos: entero)

Pre:

- Recursos: El planeta idplaneta tiene recursos de tipo pilaRecursos *(ARC: Dominio)*
- Libres: El jugador dispone de al menos un contenedor vacío *(ARC: Dominio)*
- AP: Acción planetaria en progreso *(ARC: Presentación)*

Post:

- El jugador dispone de una acción planetaria menos *(ARC: Dominio)*
- El jugador dispone de un contenedor vacío menos *(ARC:Dominio)*
- La pila de recursos pilaRecursos del planeta idplaneta ha pasado a ser propiedad del jugador *(ARC: Dominio+Datos)*

Context descargarContenedor(idplaneta:entero, pilaRecursos: entero)

Pre:

- RecursoAdecuado: pilaRecursos es del tipo solicitado por el planeta idplaneta *(ARC: Dominio+Datos)*
- AP: Acción planetaria en progreso *(ARC: Presentación)*

Post:

- El jugador dispone de una acción planetaria menos *(ARC:Dominio)*
- El jugador dispone nuevamente de un contenedor vacío *(ARC:Dominio)*
- El jugador obtiene tantos megagalax como indique la suma del valor de las cartas de recurso pilaRecursos: *(ARC: Dominio+Datos)*
  - Si el numero total de megagalax del jugador excede la cuota[9], se lanzará el caso de uso FinJuego
- El reverso de pilaRecursos pasa a formar parte del inventario de recursos del planeta idplaneta *(ARC: Dominio+Datos)*
  - Si tras girar las cartas de pilaRecursos se produce una pareja de dos cartas del mismo tipo, estas se retiran del juego
  - Si tras girar las cartas de pilaRecursos se producen dos o más parejas de dos cartas del mismo tipo, el jugador decide cual de ellas se retira del juego

Context comprarTecnologia(cartaTech: entero)

Pre:

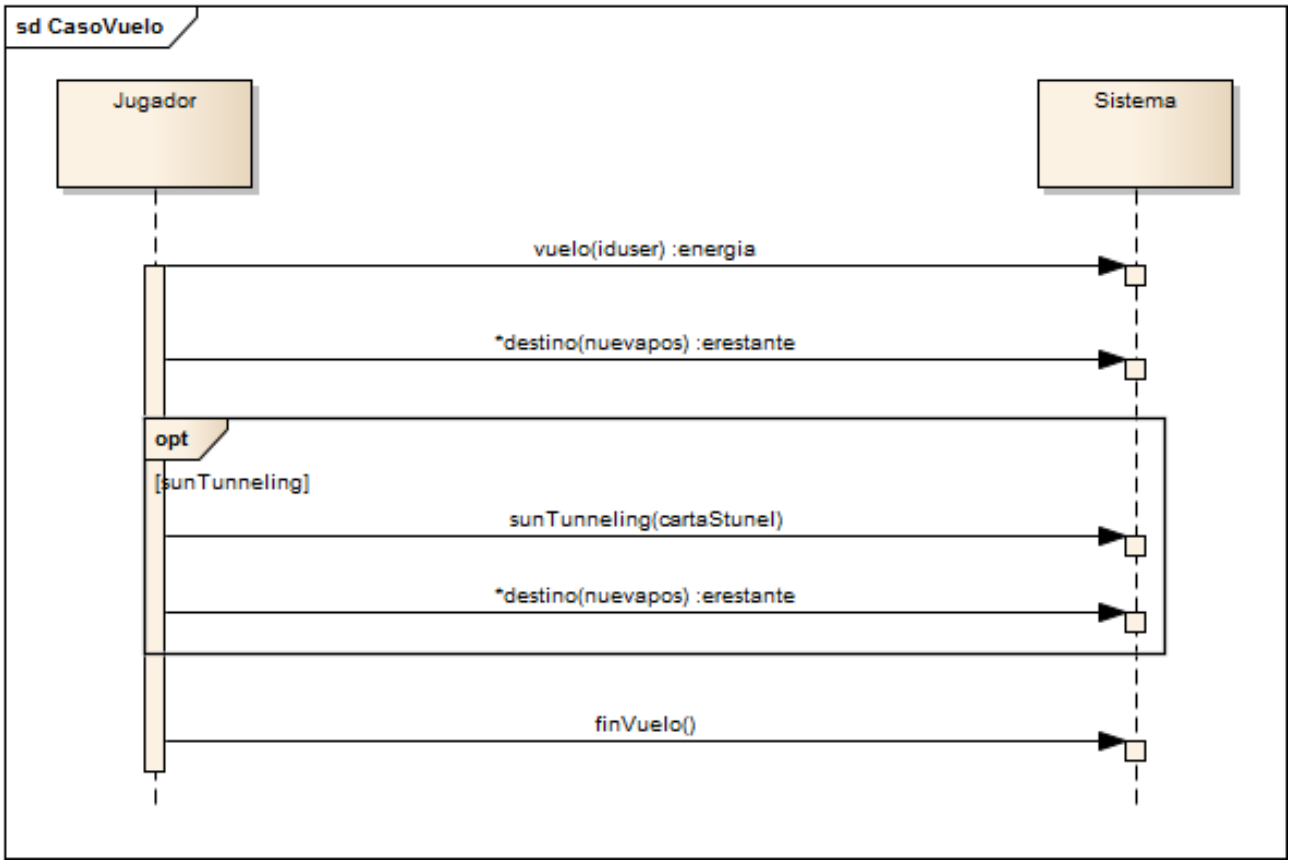
- Tech: cartaTech es una carta de tipo tecnología *(ARC: Datos)*
- MegagalaxSuficientes: el jugador dispone de megagalax suficientes para pagar la tecnología *(ARC:Dominio)*
- AP: Acción planetaria en progreso *(ARC: Presentación)*

Post:

- El jugador dispone de una acción planetaria menos *(ARC:Dominio)*
- Se añade una tecnología del tipo indicado por cartaTech al inventario del jugador *(ARC:Dominio)*
- el jugador descarta cartaTech *(ARC:Dominio)*
- Se substrae el coste de la tecnología al total de megagalax del jugador *(ARC:Dominio)*

9 Cantidad necesaria de megagalax para ganar una partida, se establece al principio de una partida (70 por defecto)

6. Vuelo



Context vuelo(iduser: entero): energia

Pre:

- Jugador: el usuario iduser es de tipo Jugador (es su turno) *(ARC:Dominio)*
- Vuelo: el jugador dispone de su acción de vuelo *(ARC: Dominio)*

Post:

- Se le asigna al jugador un numero aleatorio (>1) de unidades de energía, más una unidad energía adicional por cada tecnología de tipo fusion drive en su inventario *(ARC:Dominio)*
- Energia = numero de unidades de energía asignadas al jugador *(ARC:Dominio)*
- Se inicia un Vuelo *(ARC: Presentación)*

Context destino(nuevapos: entero):erestante

Pre:

- VP: Vuelo en progreso *(ARC: Presentación)*
- Energia: nuevapos se encuentra dentro del rango delimitado por las unidades de energía *(ARC: Dominio)*

Post:

- La posición del jugador ha pasado a ser nuevapos *(ARC: Dominio)*
- Se le substraen al jugador las unidades de energía necesarias para volar a la posición nuevapos *(ARC: Dominio)*
- Erestante: energía restante del jugador tras restar las unidades consumidas por esta acción *(ARC: Presentación)*

Context finVuelo()

Pre:

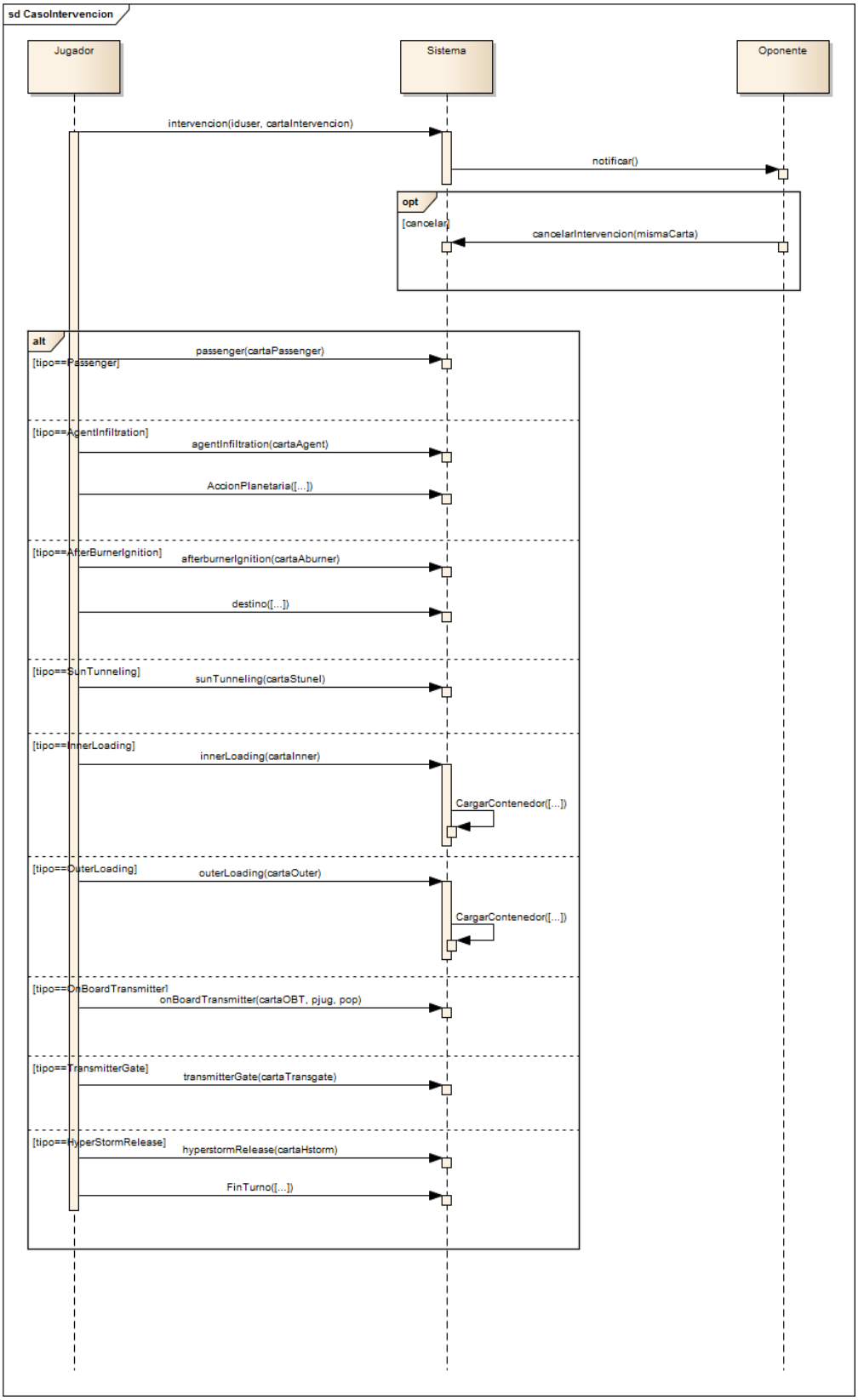
- VP: Vuelo en progreso *(ARC: Presentación)*

Post:

- El jugador no dispone de la acción de vuelo *(ARC: Dominio)*
- Termina la acción de vuelo *(ARC: Presentación)*

Sun Tunneling se considera una intervención, Y como tal está especificada en dicho apartado

7. Intervención



Context intervención(iduser:entero,cartaITV:entero):tipoIntervencion

Pre:

- Jugador: el usuario iduser es de tipo Jugador (es su turno) *(ARC:Dominio)*
- Intervenciones: el jugador dispone de al menos una intervención *(ARC: Dominio)*
- TipoAdecuado: cartaITV es de tipo Intervención *(ARC: Datos)*

Post:

- TipoIntervencion= subtipo de cartaITV *(ARC:Dominio+Datos)*
- El jugador dispone de una intervención menos *(ARC: Dominio)*
- El sistema notifica al oponente la intervención *(ARC: Presentación+Dominio)*
- Se inicia una Intervención *(ARC: Presentación)*

Context cancelarIntervención(mismaCarta:entero)

Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtitipo: mismaCarta es del mismo subtipo que la intervención en curso *(ARC: Presentación)*

Post:

- El jugador descarta mismaCarta *(ARC:Dominio)*
- El oponente descarta mismaCarta *(ARC:Dominio)*
- Se acaba la intervención *(ARC: Presentación)*

Context passenger(cartaPassenger:entero)

Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaPassenger es una carta de tipo Intervención subtipo Passenger *(ARC: Presentación)*
- PosCorrecta: el jugador se encuentra en la posición solicitada por el pasajero *(ARC: Presentación)*

Pos:

- El jugador descarta cartaPassenger *(ARC: Dominio)*
- El jugador obtiene 3 megagalax *(ARC: Dominio)*
  - Si el numero total de megagalax del jugador excede la cuota, se lanzará el caso de uso FinJuego
- Se acaba la Intervención *(ARC: Presentación)*

Context agentInfiltration(cartaAgent:entero)

Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaAgent es una carta de tipo Invervención subtipo Agent Infiltration *(ARC: Presentación)*

Pos:

- El jugador descarta cartaAgent *(ARC: Dominio)*
- El jugador obtiene 1 acción planetaria adicional*(ARC: Dominio)*
- Se acaba la Intervención *(ARC: Presentación)*

Context afterburnerIgnition(cartaAburner:entero)

Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaAburner es una carta de tipo Invervención subtipo Afterburner Ignition *(ARC: Presentación)*
- VueloConsumido: el jugador ha consumido su acción de Vuelo de el presente turno *(ARC: Dominio)*

Pos:

- El jugador descarta cartaAburner *(ARC: Dominio)*
- El jugador obtiene 1 acción de Vuelo*(ARC: Dominio)*
- Se acaba la Intervención *(ARC: Presentación)*

Context sunTunneling(cartaStunel:entero)

Pre:

- VP: Vuelo en progreso *(ARC: Presentación)*
- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaStunel es una carta de tipo Intervención subtipo SunTunneling *(ARC: Presentación)*
- Energía: El jugador dispone de al menos una unidad de energía *(ARC: Dominio)*
- PosCorrecta: El jugador se encuentra en la órbita de la posición más cercana al sol *(ARC: Dominio)*

Post:

- El jugador dispone de una unidad de energía menos *(ARC: Dominio)*
- El jugador descarta cartaStunnel *(ARC: Dominio)*
- El jugador se encuentra en la órbita del planeta más alejado del sol *(ARC: Dominio)*

Context innerLoading(cartaInner:entero)

Pre:

- Libres: El jugador dispone de al menos un contenedor vacío *(ARC: Dominio)*
- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaInner es una carta de tipo Intervención subtipo Inner Loading *(ARC: Presentación)*

Post:

- El jugador descarta cartaInner *(ARC:Dominio)*
- El sistema inicia una acción Cargar Contenedor en uno de los dos planetas interiores *(ARC: Presentación)*
- Se acaba la Intervención *(ARC: Presentación)*

Context OuterLoading(cartaOuter:entero)

Pre:

- Libres: El jugador dispone de al menos un contenedor vacío *(ARC: Dominio)*
- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaOuter es una carta de tipo Intervención subtipo Outer Loading *(ARC: Presentación)*

Post:

- El jugador descarta cartaOuter *(ARC:Dominio)*
- El sistema inicia una acción Cargar Contenedor en uno de los dos planetas exteriores *(ARC: Presentación)*
- Se acaba la Intervención *(ARC: Presentación)*

Context OnBoardTransmitter(cartaOBT:entero, pjuj:entero, pop:entero)

Pre:

- Intercambiables: los contenedores pjuj del jugador y pop del oponente no están vacíos: *(ARC: Presentación)*
- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaOBT es una carta de tipo Intervención subtipo On-board Transmitter *(ARC: Presentación)*

Post:

- El jugador descarta cartaOBT *(ARC:Dominio)*
- Se intercambia el contenido de los contenedores pjuj del jugador y pop del oponente *(ARC: Dominio)*
- Se acaba la Intervención *(ARC: Presentación)*

Context transmitterGate(cartaTGA:entero)

Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaTGA es una carta de tipo Intervención subtipo Transmitter Gate Activation *(ARC: Presentación)*

Post:

- El jugador descarta cartaTGA *(ARC:Dominio)*
- Se intercambian las posiciones del jugador y el oponente *(ARC: Dominio)*
- Se acaba la Intervención *(ARC: Presentación)*

Context hyperstormRelease(cartaHstorm:entero)

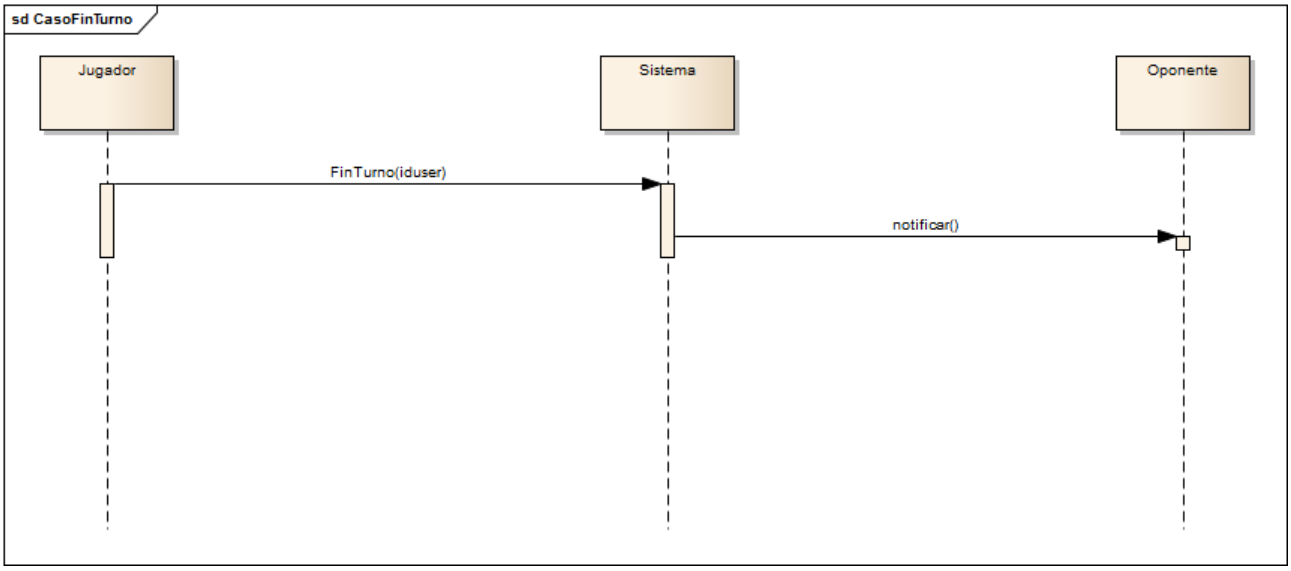
Pre:

- IP: Intervención en progreso *(ARC: Presentación)*
- Subtipo: cartaHstorm es una carta de tipo Intervención subtipo Hyperstorm Release *(ARC: Presentación)*

Post:

- El jugador descarta TODAS las cartas de su mano. La pila de descarte y robo del jugador se fusionan para formar un nuevo mazo de robo y el jugador roba 5 cartas del nuevo mazo *(ARC: Dominio)*
- Se acaba la intervención *(ARC: Presentación)*
- Lanza el caso de uso Fin Turno *(ARC: Presentación+Dominio)*

8. Fin Turno



Context finTurno(iduser:entero)

Pre:

- Jugador: el usuario iduser es de tipo Jugador (es su turno) *(ARC:Dominio)*

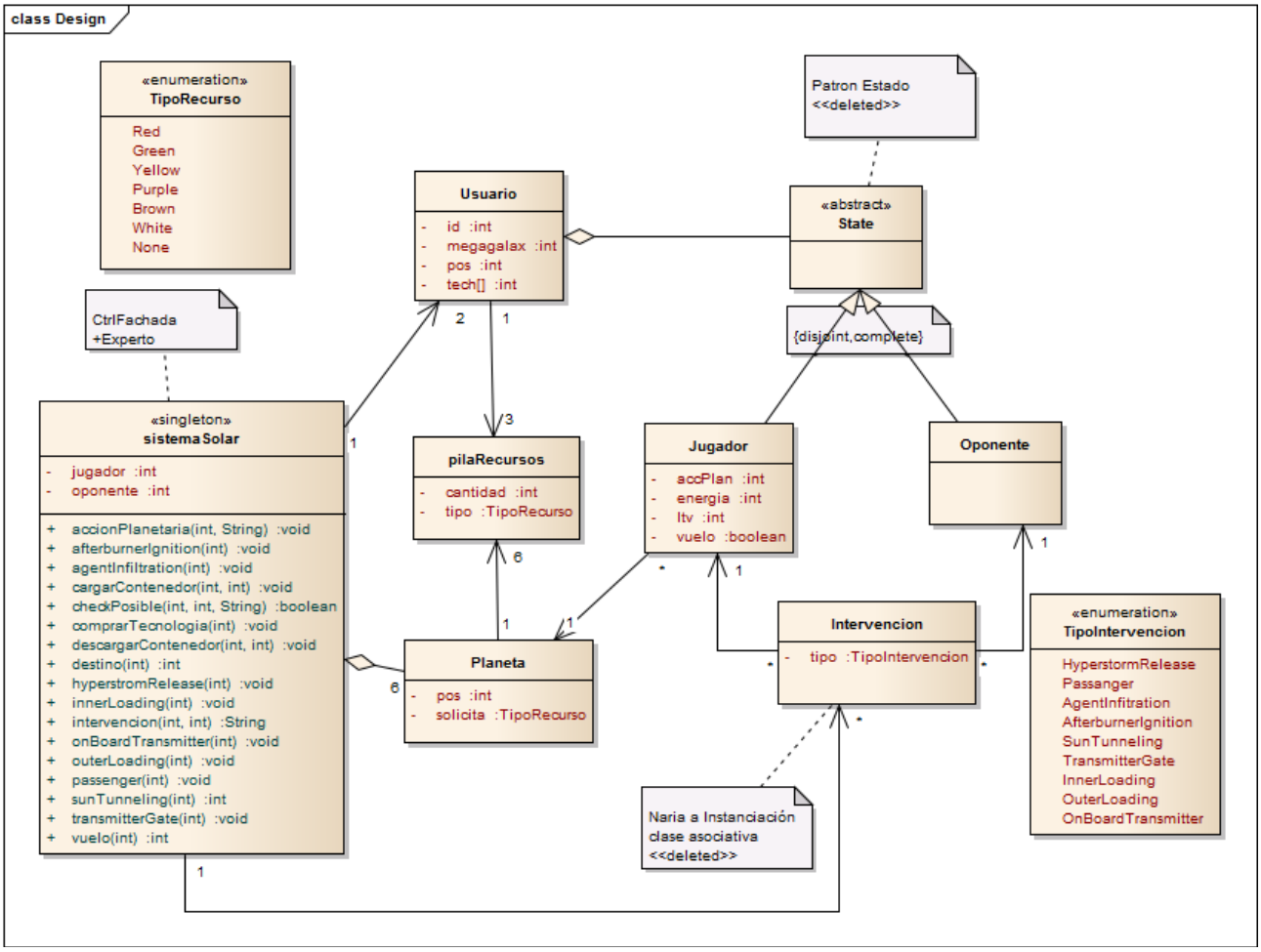
Post:

- El jugador pasa a ser oponente *(ARC:Presentación+Dominio)*
- El oponente pasa a ser jugador *(ARC:Presentación+Dominio)*
- Si su mazo lo permite el jugador roba tantas cartas como tecnologías de tipo Replentishment haya en su inventario (hasta un máximo de 5) *(ARC: Dominio)*

Diseño

Diseño....72
1. Diagrama de Clases....73
2. Iniciar Juego....74
3. Consultar Ayuda....75
4. Fin Juego....76
5. Acción Planetaria....77
6. Vuelo....81
7. Intervención....85
8. Fin Turno....96

1. Diagrama de Clases



En este documento se utilizan las etiquetas de las precondiciones usadas en el documento Especificación obviando sus descripciones.

Debido a problemas de compatibilidad con la Interficie gráfica con la técnica del multithread, el cambio de estado no se puede implementar como tal (Transacción), por lo que en su lugar, el controlador de dominio se encargará de gestionar el turno de cada jugador mediante aritmética de punteros y booleanos. La sincronización entre usuarios se hace mediante el patrón observador

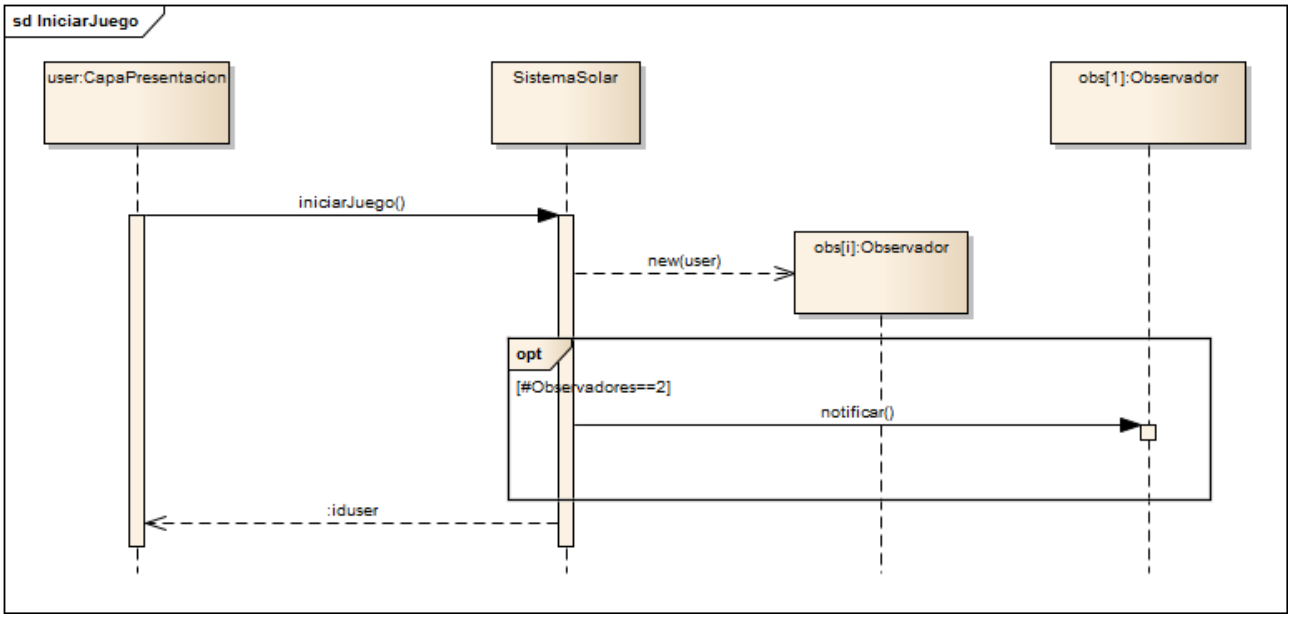
2. Iniciar Juego

Context SistemaSolar::iniciarJuego(): idjugador

Pre:

- Init: Garantizado por la capa de presentación. De otra manera, no tendría sistema al que lanzar esta función

Post: […]



El sistema crea dos observadores, uno por cada usuario que se encargarán de mantener el sincronismo entre jugadores y de actualizar al usuario de los cambios en el sistema (log)

Nótese que estos observadores son independientes del tipo de usuario en cuestión (IA, local, remoto)

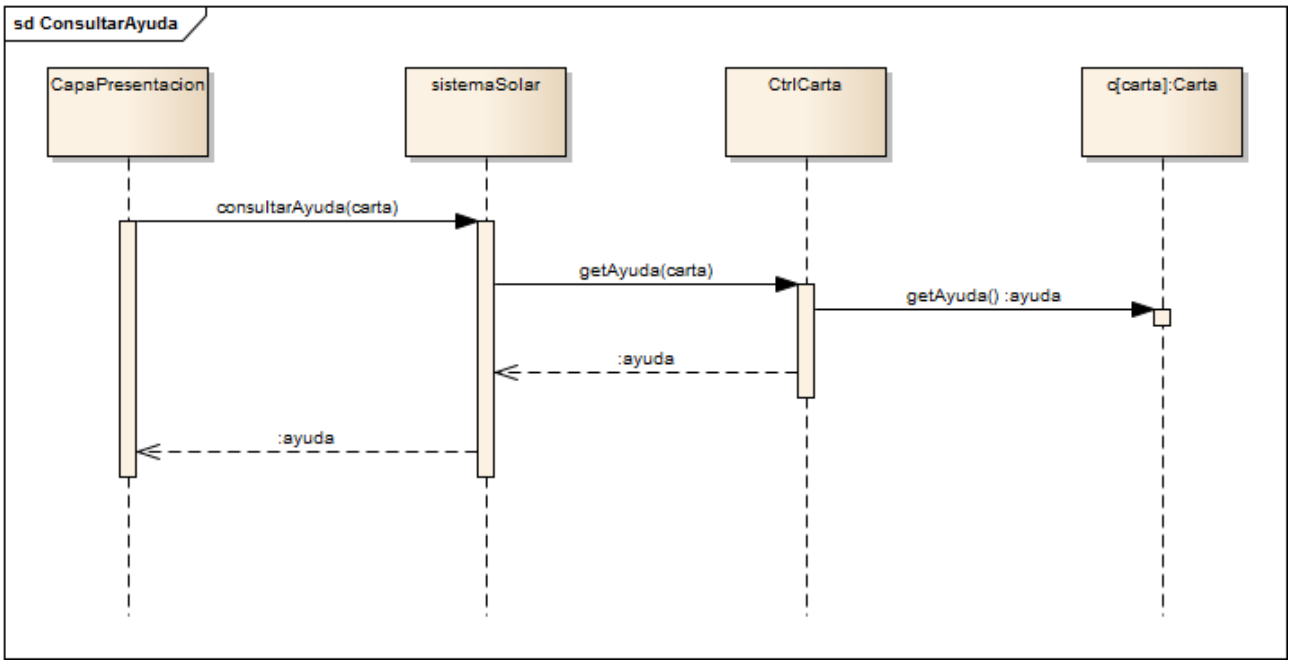
3. Consultar Ayuda

Context ConsultarAyuda(carta:entero): descripciónAyuda

Pre:

- CartaValida: La interfaz de la capa de presentación esta indexada

Post: […]



Función Trivial. Operación consultora de la base de datos.

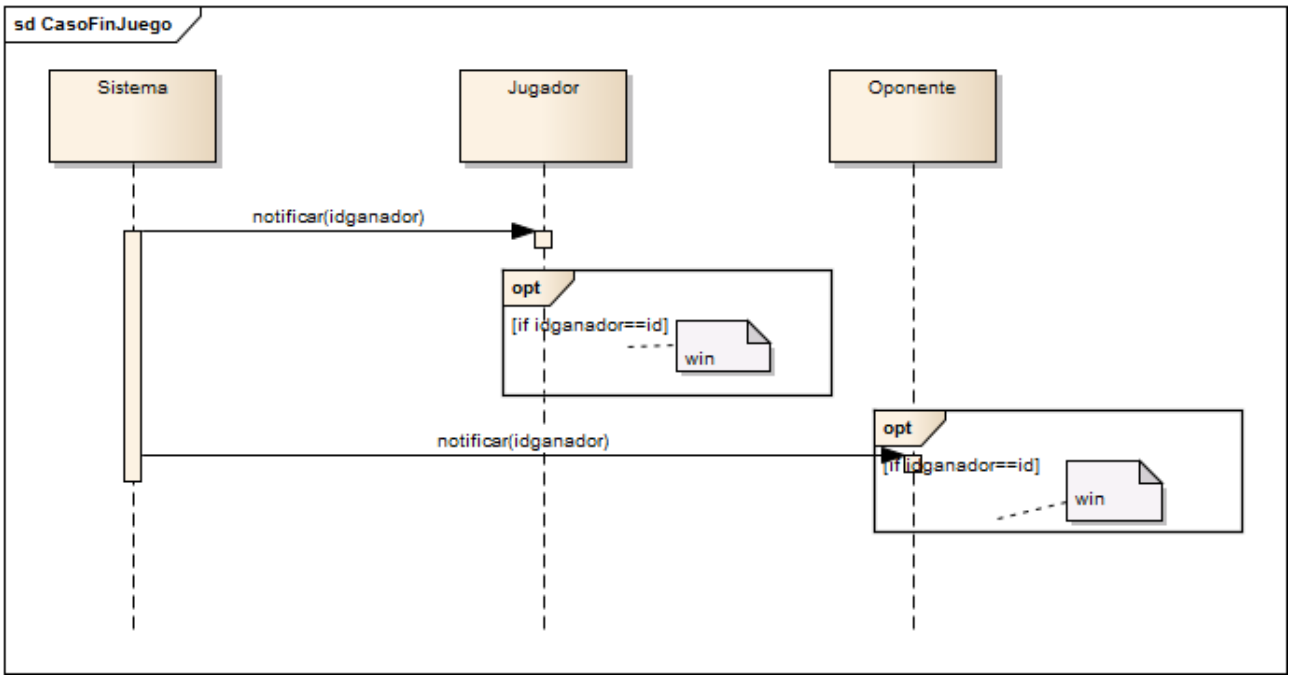
4. Fin Juego

Context SistemaSolar :: FinJuego()

Pre:

- Fin: Postcondición de passenger y descargarContenedor

Post: […]



El sistema notifica a los observadores de este hecho y estos actualizan al usuario informándole de si han ganado o han perdido

5. Acción Planetaria

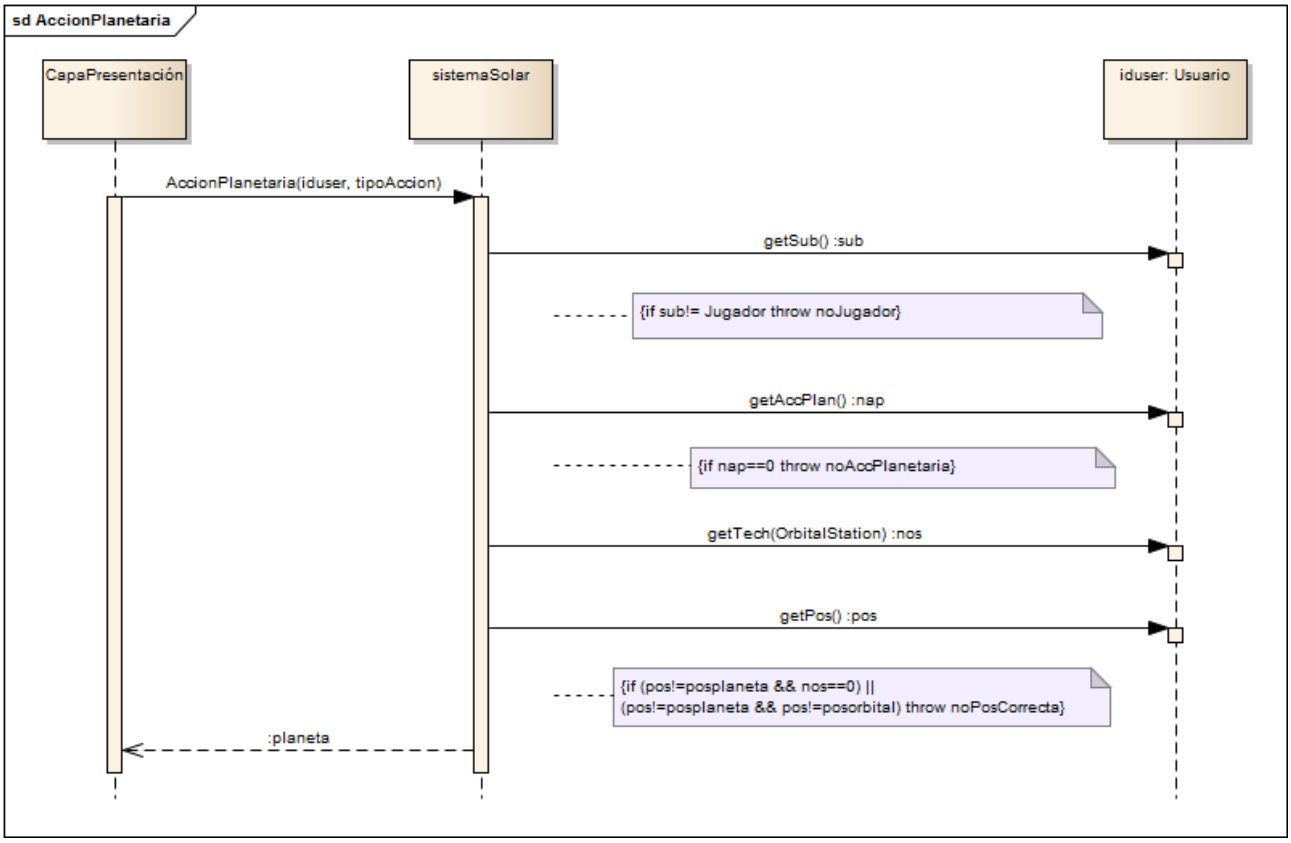
Context SistemaSolar :: accionPlanetaria(iduser: entero, TipoAccion: String):idplaneta

Pre: -----

Exc:

- noJugador: verificado por SistemaSolar
- noAccionPlanetaria: verificado por SistemaSolar
- noPosCorrecta: verificado por SistemaSolar

Post: […]



Esta función se ejecuta al principio de cada acción planetaria, para verificar el estado del jugador. Para propósitos de usabilidad, existe una variante similar a esta función llamada **testplanetaria** que funciona exactamente igual salvo que devuelve un booleano en lugar de lanzar excepciones. La capa de presentación la utiliza para determinar que acciones puede realizar el usuario sin causar una excepción

Context SistemaSolar :: cargarContenedor(idplaneta:entero, pilaRecursos: entero)

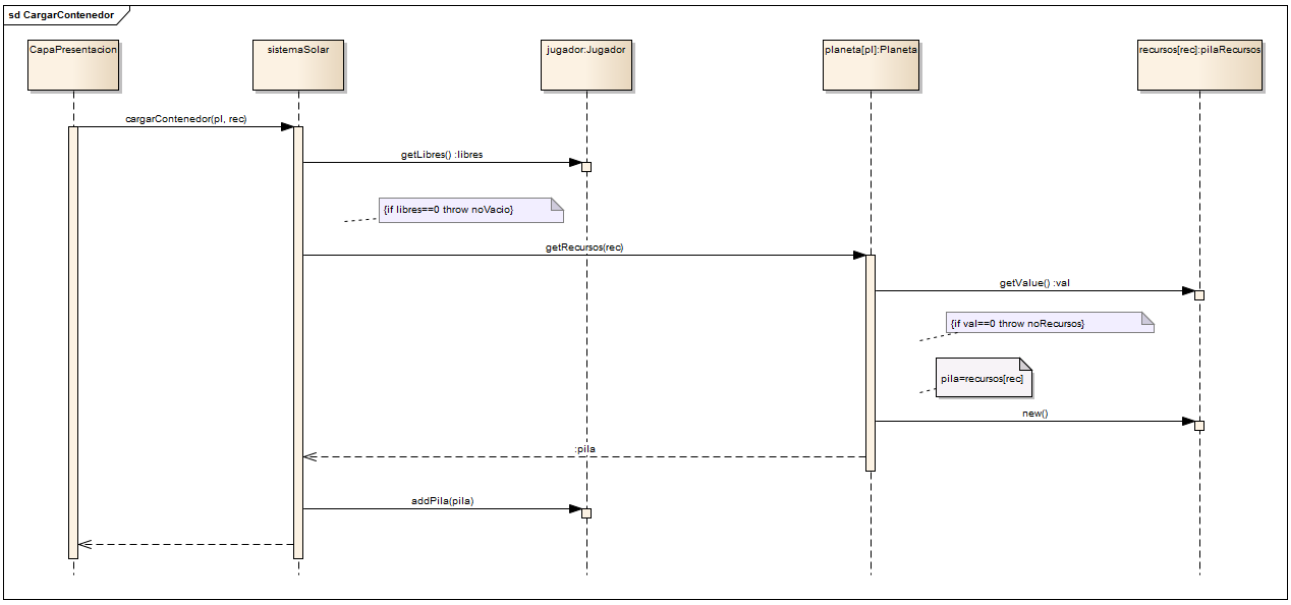
Pre:

- AP: La capa de presentación requiere de idplaneta (resultado de acciónPlanetaria) para lanzar esta función

Exc:

- noLibres: verificado por SistemaSolar
- noRecursos: verificado por Planeta

Post: […]



Sistema solar verifica los parámetros necesarios, y delega la responsabilidad de esta función a planeta (función getRecursos) ya que no tiene la visibilidad necesaria sobre pilaRecursos

Context SistemaSolar :: descargarContenedor(idplaneta:entero, pilaRecursos: entero)

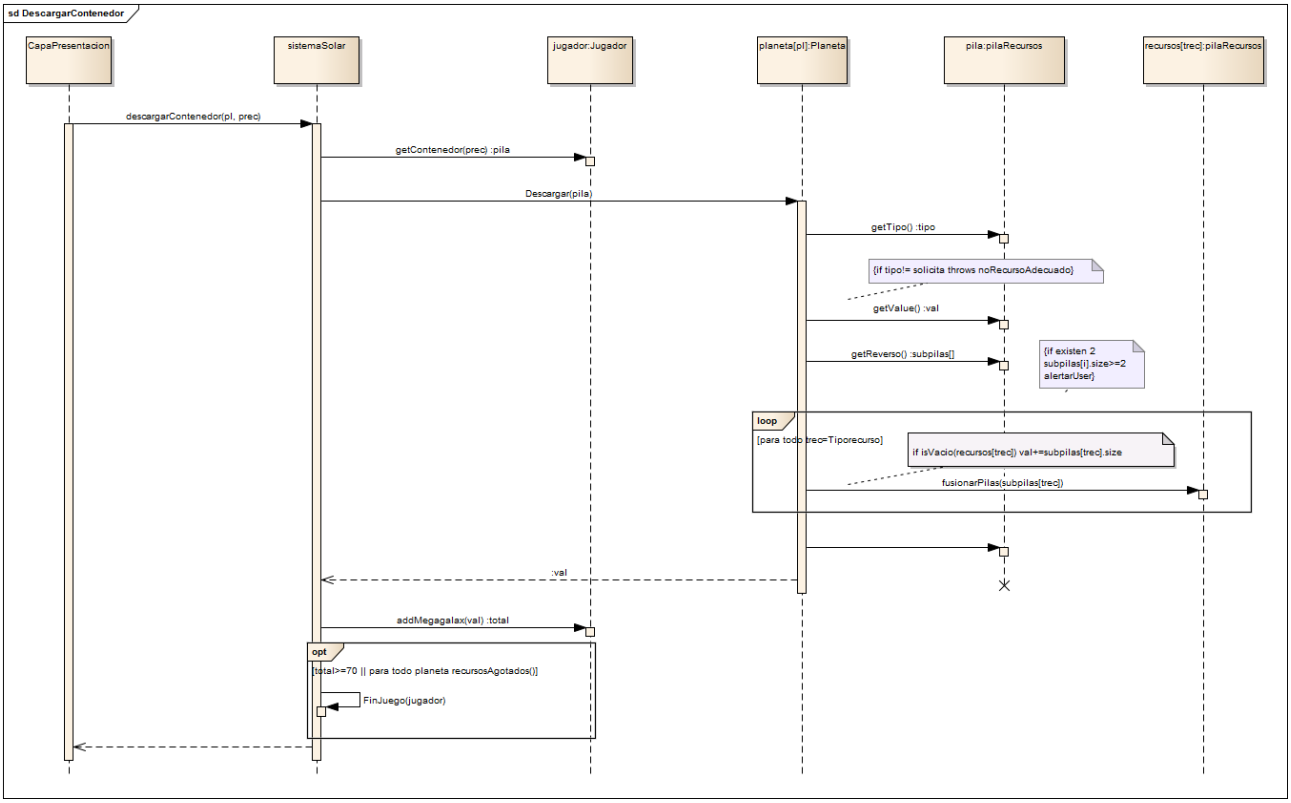
Pre:

- AP: Ídem cargarContenedor

Exc:

- noRecursoAdecuado: Verificado por pilaRecursos

Post: […]



Al igual que en el caso anterior, delegamos la responsabilidad a quien tiene los recursos necesarios para hacerla. Recordemos que en caso de que se produzca una pareja de recursos repetidos al girar la pila, esta será substraída del juego

Context SistemaSolar :: comprarTecnologia(cartaTech: entero)

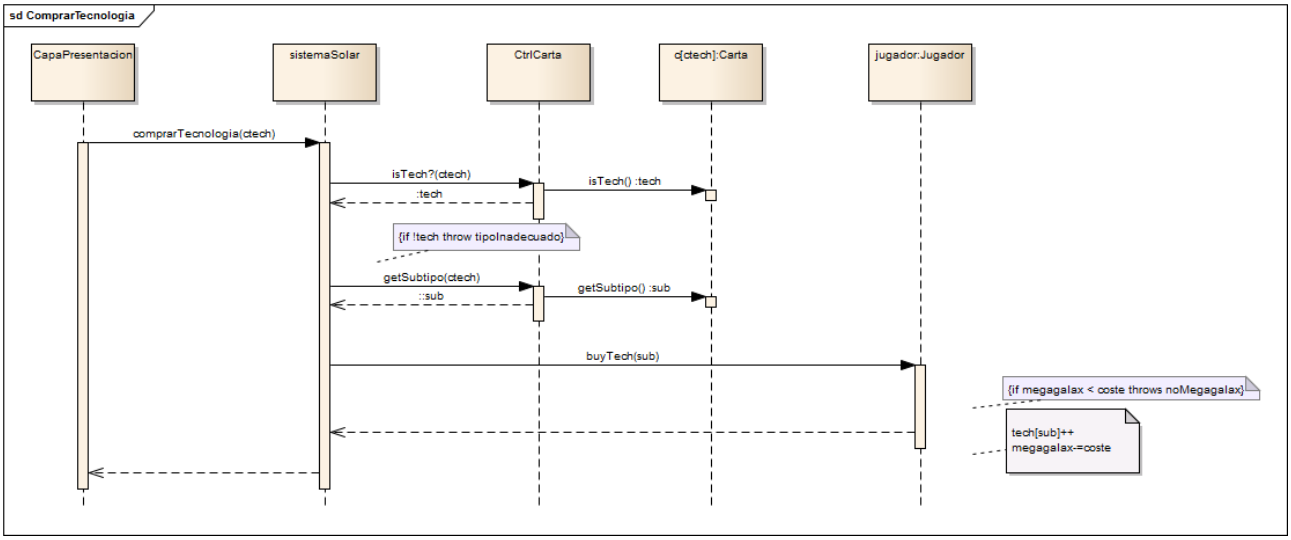
Pre:

- AP: Ídem cargarContenedor

Exc:

- noTech: verificado por SistemaSolar
- noMegagalaxSuficientes: delegado a Jugador

Post: […]



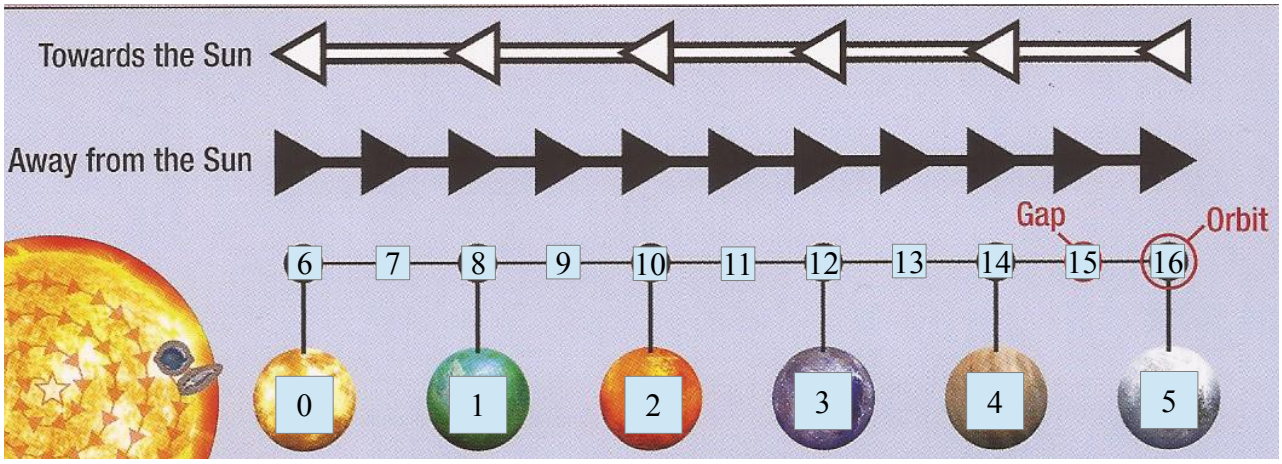
En este caso, requerimos consultar el controlador de la base de datos para verificar si la carta a jugar es de tipo tecnología. Al igual que en el caso de testplanetaria, la capa de presentación también cuenta con la opción de verificar este hecho sin causar una excepción

6. Vuelo

Convenciones:

Recordemos que el vuelo hacia el sol cuesta la mitad de energía (redondeando hacia $+\infty$) que el vuelo a la dirección opuesta.

Para facilitar la ejecución del vuelo y el cálculo de rangos, se han etiquetado a los planetas y a las posiciones orbitales de la siguiente manera



De esta manera el sistema puede determinar si el usuario se encuentra o no en una posición adecuada para realizar cierta acción, a la vez que proporciona una manera fácil de asociar órbitas a planetas y viceversa

- Posicion < 6 = planeta con órbita en (posición*2)+6
- Posicion > 6
  - par: órbita del planeta (posición – 6)/2
  - impar: espacio

Ejemplo planeta Rojo:

- Etiqueta planeta 2: (2*2)+6 = 4+6 = 10
- Etiqueta órbita 10: (10-6)/2 = 4/2 = 2

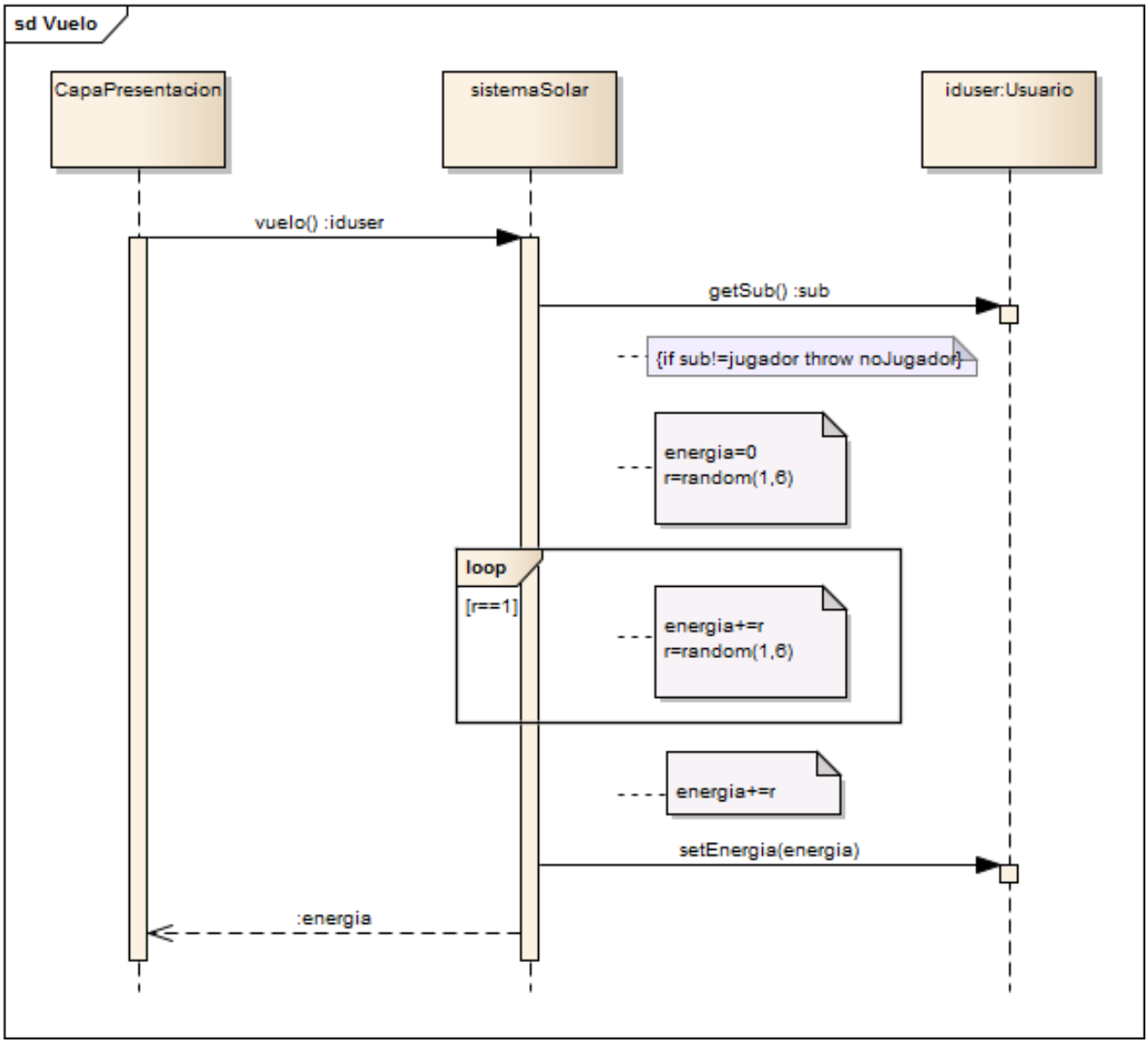
Context SistemaSolar: vuelo(iduser: entero): energia

Pre: ----

Exc:

- noJugador: verificado por SistemaSolar
- noVuelo: verificado por SistemaSolar

Post: […]



Al igual que en Acción Planetaria, esta acción se encarga de verificar el estado del usuario antes de iniciar un vuelo.

Context SistemaSolar :: destino(nuevapos: entero):erestante

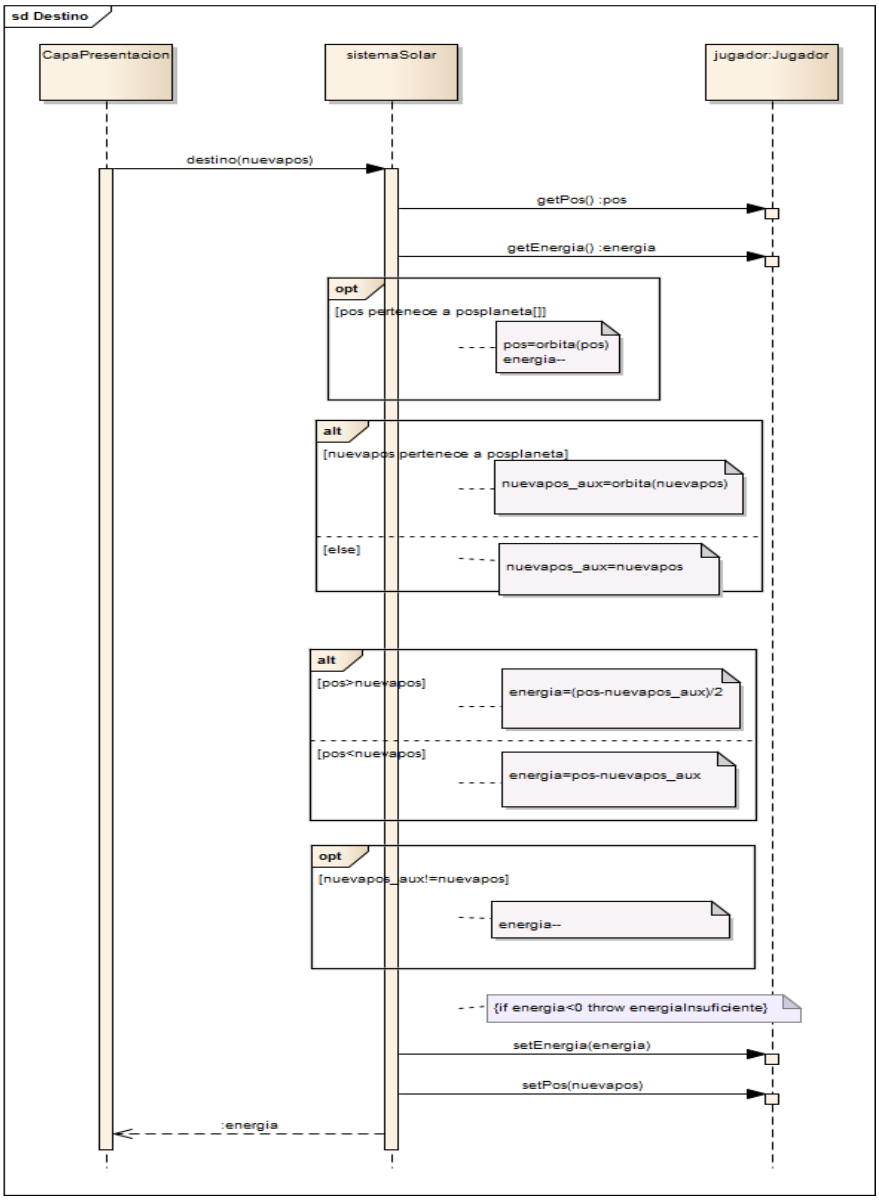
Pre:

- VP: Garantizado por la capa de presentación ya que necesita las unidades de energía para calcular el rango

Exc:

- Energia: verificado por SistemaSolar

Post: […]



Recordemos que entrar o salir de una órbita consume una unidad de energía. Para propósotos de usabilidad, existe una variante de esta función llamada **testvuelo** que dada una posición, devuelve el numero de unidades de energía necesarias para llegar a ella. La capa de presentación utiliza esta función para calcular el rango de vuelo sin causar una excepción

Context SistemaSolar :: finVuelo()

Pre:

- VP: Garantizado por la capa de presentación

Post:

- El jugador no dispone de la acción de vuelo
- Termina la acción de vuelo

Función trivial que tiene como propósito atomizar la acción de vuelo, para así prevenir posibles problemas de inconsistencia en el estado, provocados por la función destino y la intervención Sun Tunneling.

En la práctica, la Capa de Presentación simula un vuelo en varios pasos ejecutando la siguiente secuencia atómica de funciones: vuelo→destino→finVuelo

7. Intervención

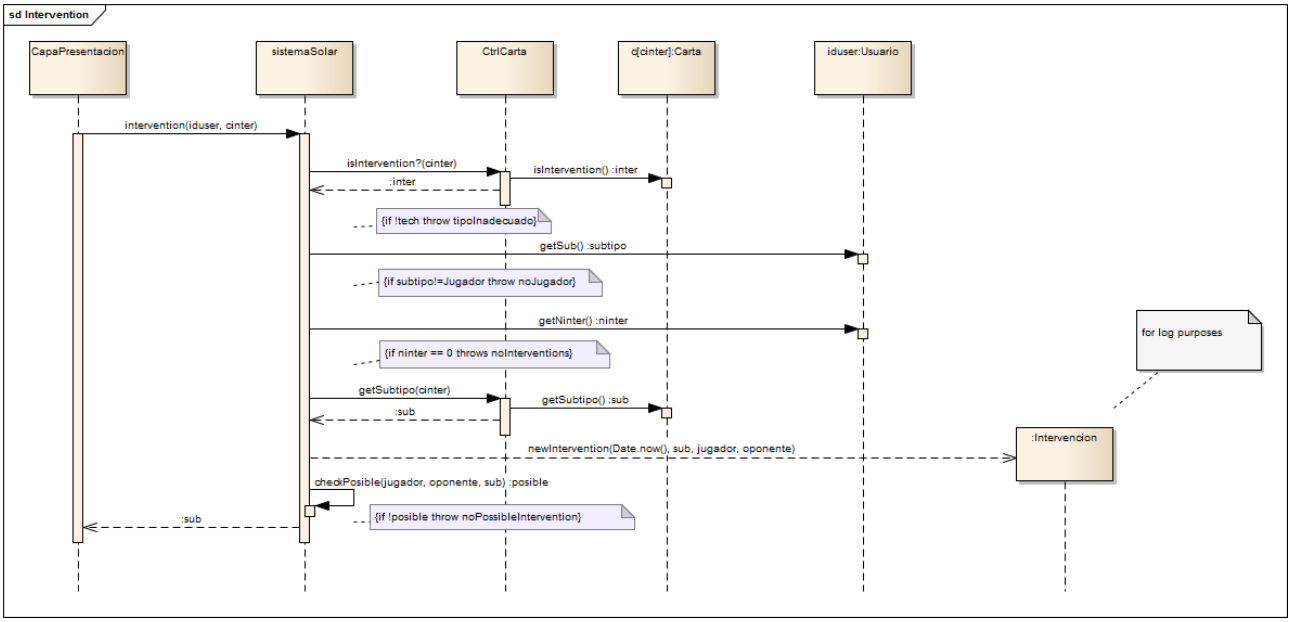
Context SistemaSolar :: intervención(iduser:entero, cartaITV:entero): tipoIntervencion

Pre: ----

Exc:

- noJugador: verificado por SistemaSolar
- noIntervenciones: verificado por SistemaSolar
- TipoInadecuado: verificado por SistemaSolar

Post: […]



En una version anterior de este programa (Transaction Scripts), la clase se le pasaba todos los parámetros a la clase Intervención y esta se encargaba de ejecutarse atómicamente. Sin embargo, se ha decidido mantener esta clase como log para testear el funcionamiento de la inteligencia artificial

Context SistemaSolar :: checkPosible(idjugador: entero, idoponente: entero, sub: tipoIntervención) : posible

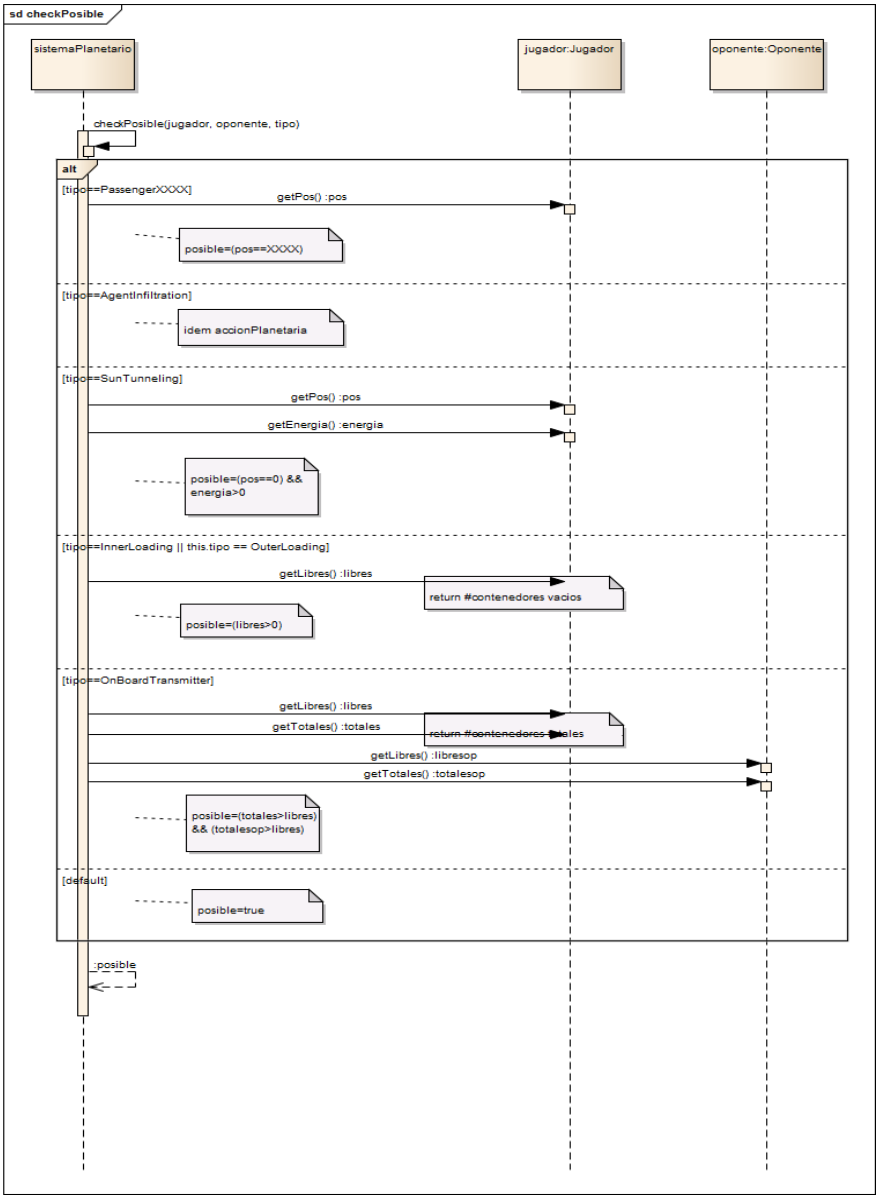
Pre:

- Subtitipo: sub es un tipo de Intervención válido (verificado por Intervención)

Exc: ----

Post:

- posible = true si es posible ejecutar la intervención de tipo sub dado el estado actual



Inicialmente, esta función era utilizada por la Capa de Presentación para verificar las condiciones de una intervención sin causar una excepción. Sin embargo, se ha decidido re-aprovechar el código para la propia función de la Capa de Dominio. Adelantamos la verificación de las condiciones para asi asegurarnos de no caer en un estado inconsistente

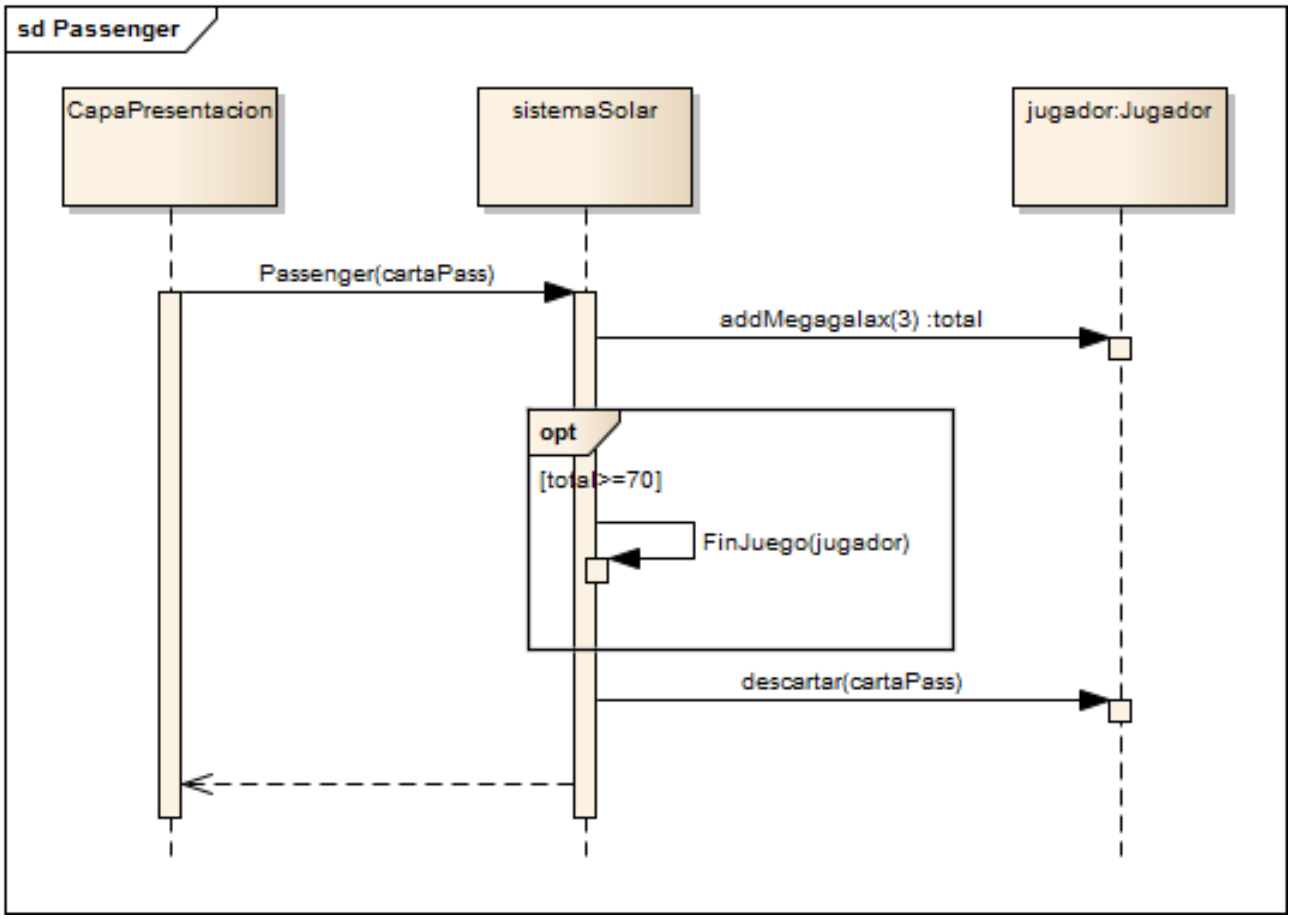
Context SistemaSolar :: passenger(cartaPassenger:entero)

Pre:

- IP: Garantizada por la Capa de Presentación
- Subtipo: Garantizada por la función intervención
- PosCorrecta: Garantizada por checkposible

Exc: ----

Pos: [...]



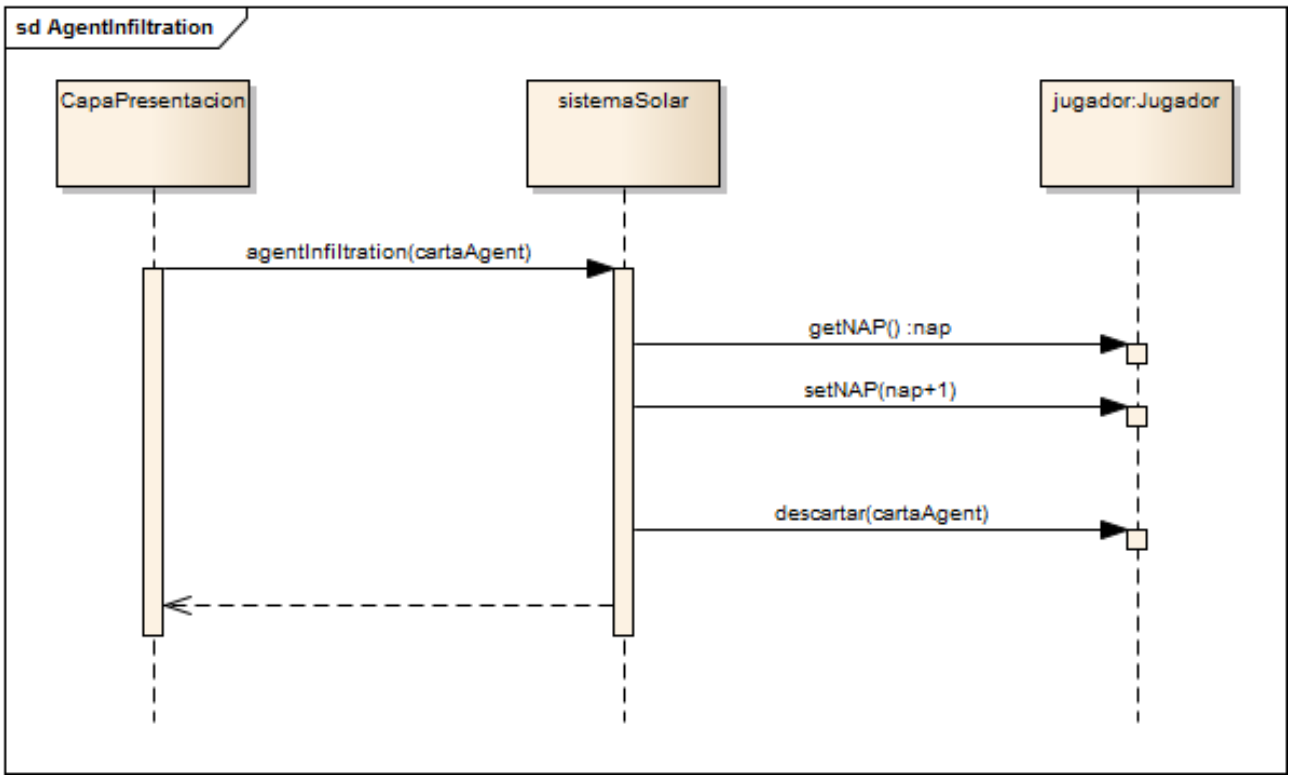
Función trivial. Recordemos que podemos alcanzar las condiciones de fin de juego al añadir los megagalax del jugador, por lo que se ha de verificar.

Context SistemaSolar :: agentInfiltration(cartaAgent:entero)

Pre:

- IP: Garantizado por Capa de Presentación
- Subtipo: Garantizado por intervención

Pos: […]



Otra función trivial. Añadimos una acción planetaria adicional para el posterior uso del jugador

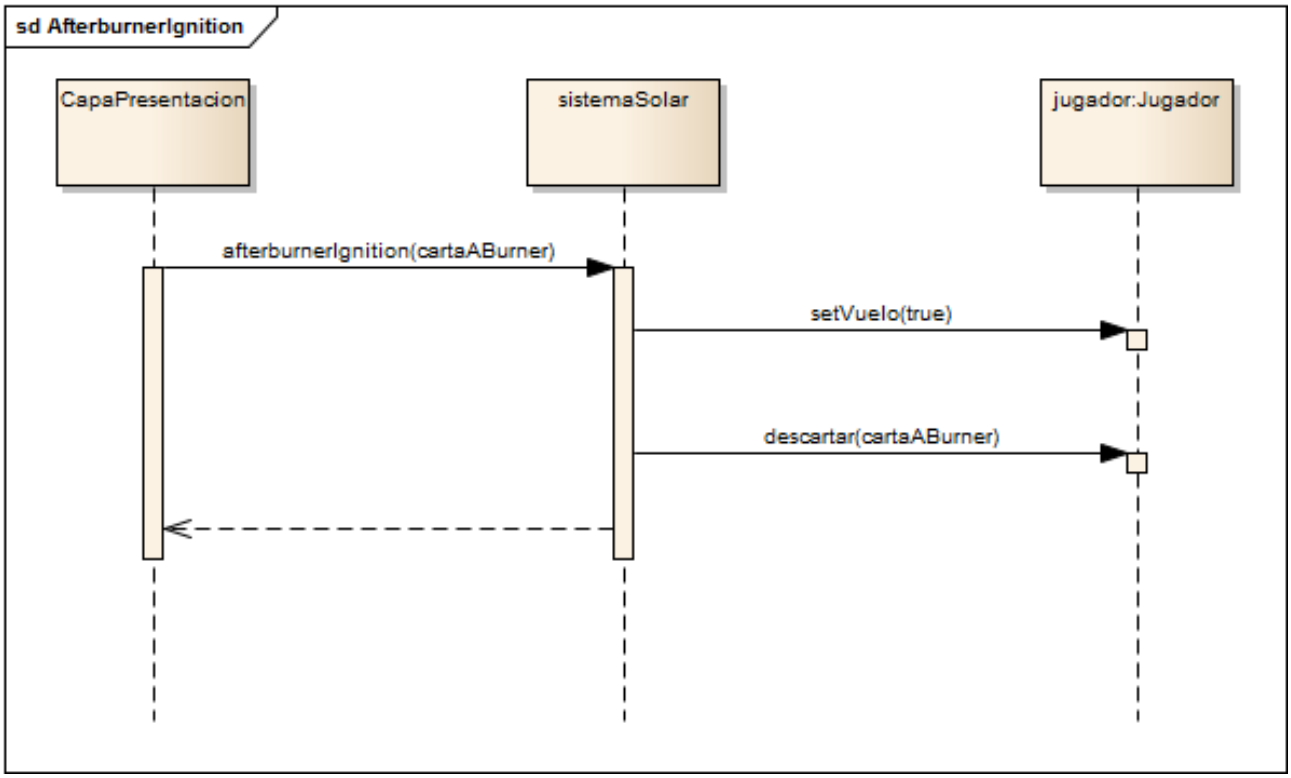
Context SistemaSolar :: afterburnerIgnition(cartaAburner:entero)

Pre:

- IP: Garantizado por Capa de presentación
- Subtipo: Garantizado por intervención
- VueloConsumido: Garantizado Por checkPosible

Exc: ----

Pos: […]



Mismo caso que en agent infiltration pero con Vuelo. Hay que tener en cuenta que es necesario que el jugador haya consumido su acción de vuelo, de otra forma, no obtendría ningún beneficio

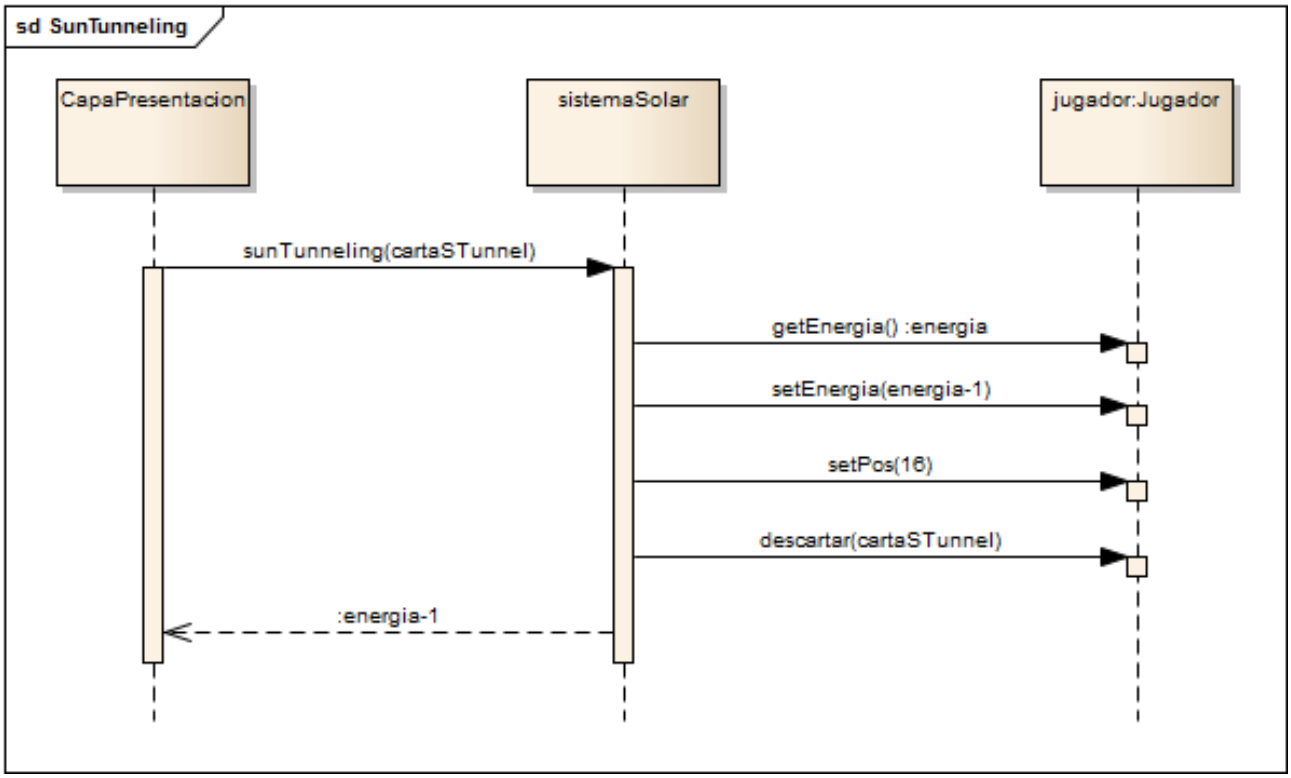
Context SistemaSolar :: sunTunneling(cartaStunel:entero)

Pre:

- VP: Garantizada por Capa de Presentación
- IP: Garantizada por Capa de Presentación
- Subtipo: Garantizada por intervención
- Energía PosCorrecta: Garantizadas por checkPosible

Exc: ----

Post: […]



Gran parte de la complejidad de esta función recae en su verificación. Esta función se ha de ejecutar durante un vuelo **y** durante una intervención, cuando el sistema esta en un estado bastante comprometido.

En la practica, la capa de presentación simula un vuelo en varios pasos ejecutando de manera atómica esta secuencia de funciones:

vuelo(...)→destino(1)→intervencion(...)→SunTunneling(...) →destino(x)→finVuelo()

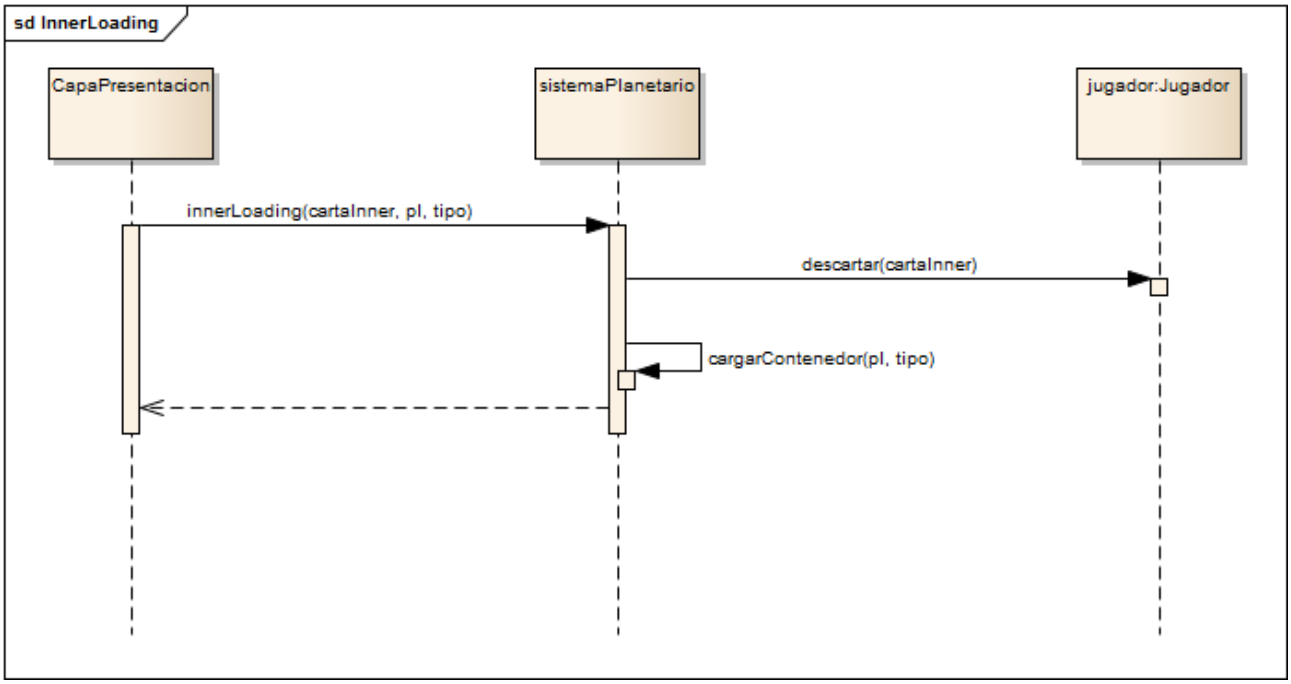
Context SistemaSolar :: innerLoading(cartaInner:entero)

Pre:

- Libres: Garantizado por checkPosible
- IP: Garantizado por Capa de Presentación
- Subtipo: Garantizado por Intervención

Exc: ----

Post: […]



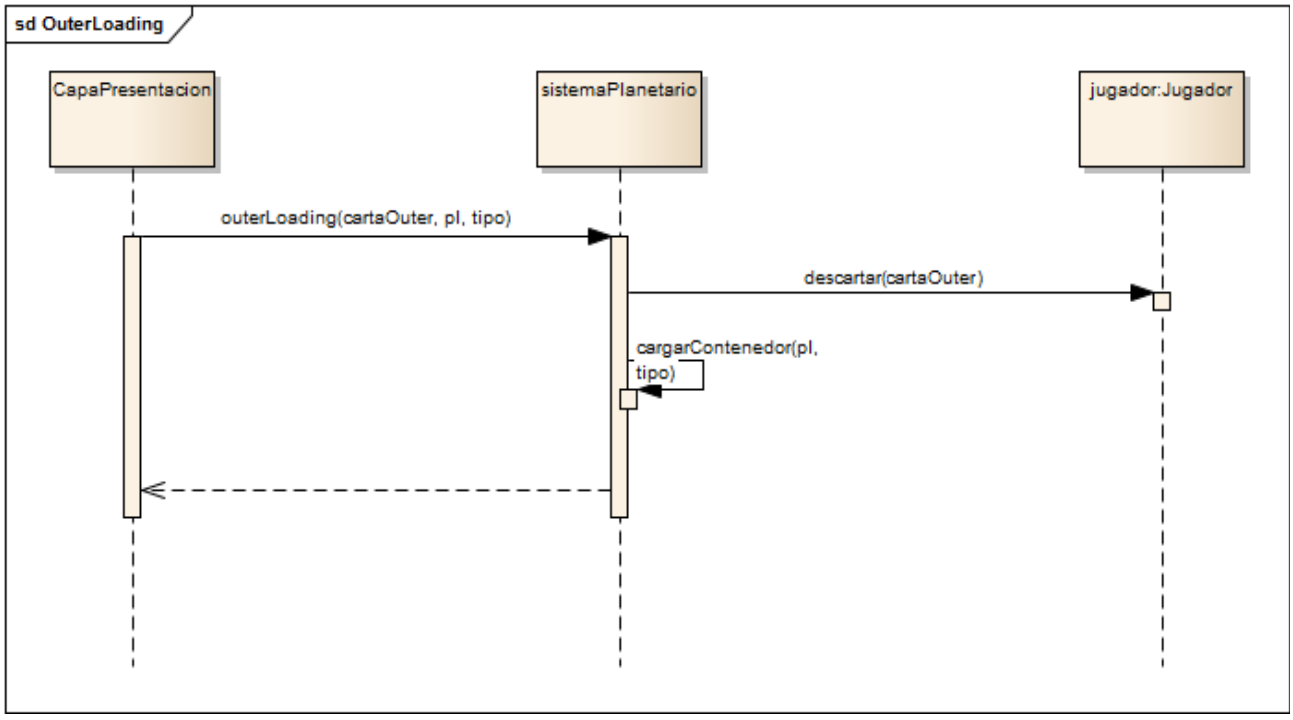
Context SistemaSolar :: outerLoading(cartaOuter:entero)

Pre:

- Libres: Garantizado por checkPosible
- IP: Garantizado por Capa de Presentación
- Subtipo: Garantizado por Intervención

Exc: ----

Post: […]



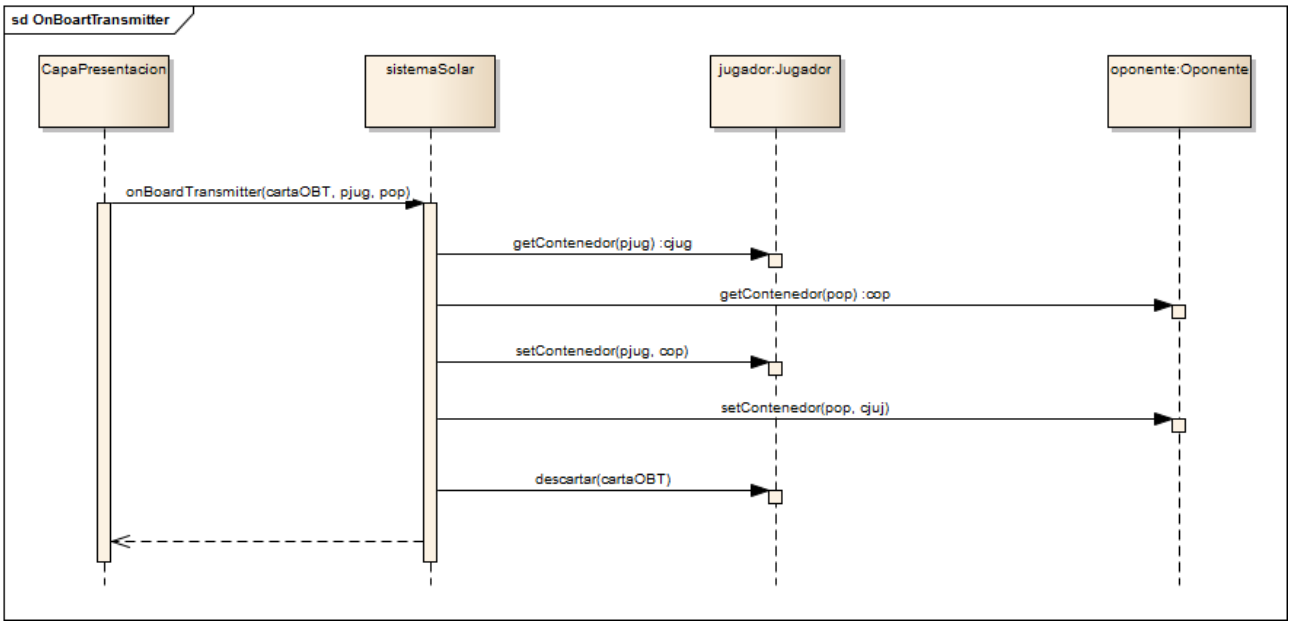
En la práctica, Inner y Outer loading estan implementados por la misma función, para re-aprovechar código

Context SistemaSolar :: OnBoardTransmitter(cartaOBT:entero, pjuj:entero, pop:entero)

Pre:

- Intercambiables: Garantizado por Capa de Presentación (consultoras estándar)
- IP: Garantizada por Capa de Presentación
- Subtipo: Garantizado por Intervención

Post: […]



Función relativamente simple basada en operaciones con punteros. La subinterfaz de la capa de presentación responsable de escoger pjug y pop cuenta utiliza consultoras estándar para asegurarse que no son vacías

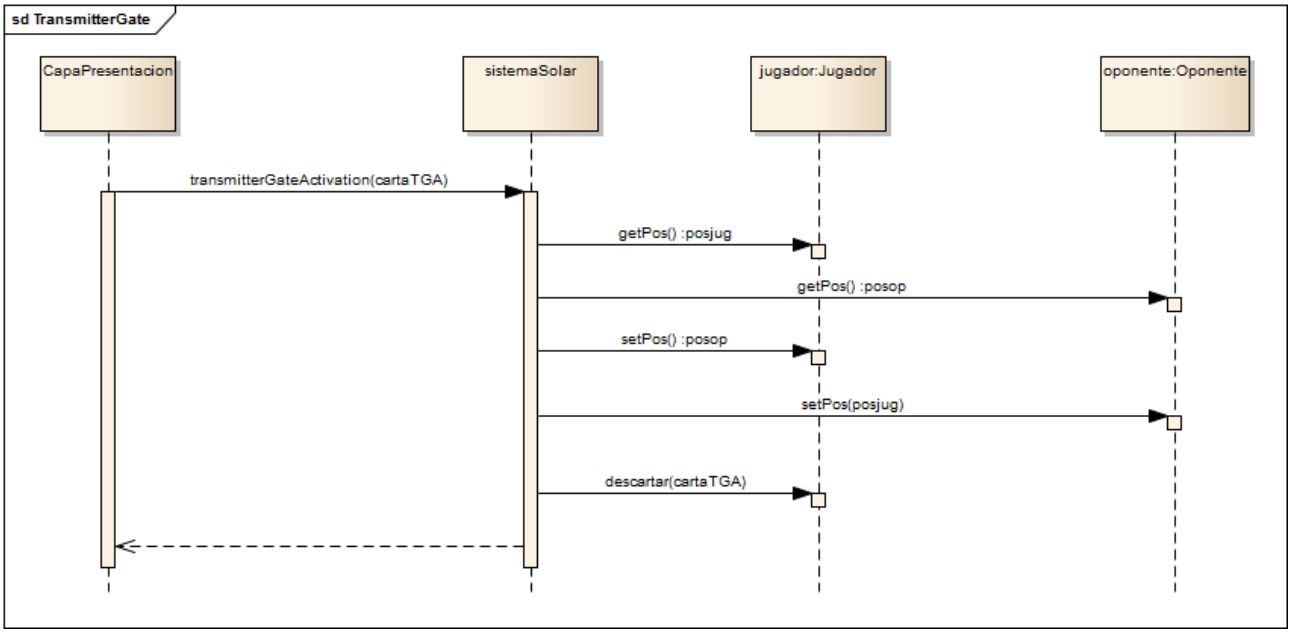
Context SistemaSolar :: transmitterGate(cartaTGA:entero)

Pre:

- IP: Garantizada por Capa de Presentación
- Subtipo: Garantizada por Intervención

Exc: ----

Post: […]



Función trivial. Intercambia las posiciones de los usuarios sin ningún tipo de restricción

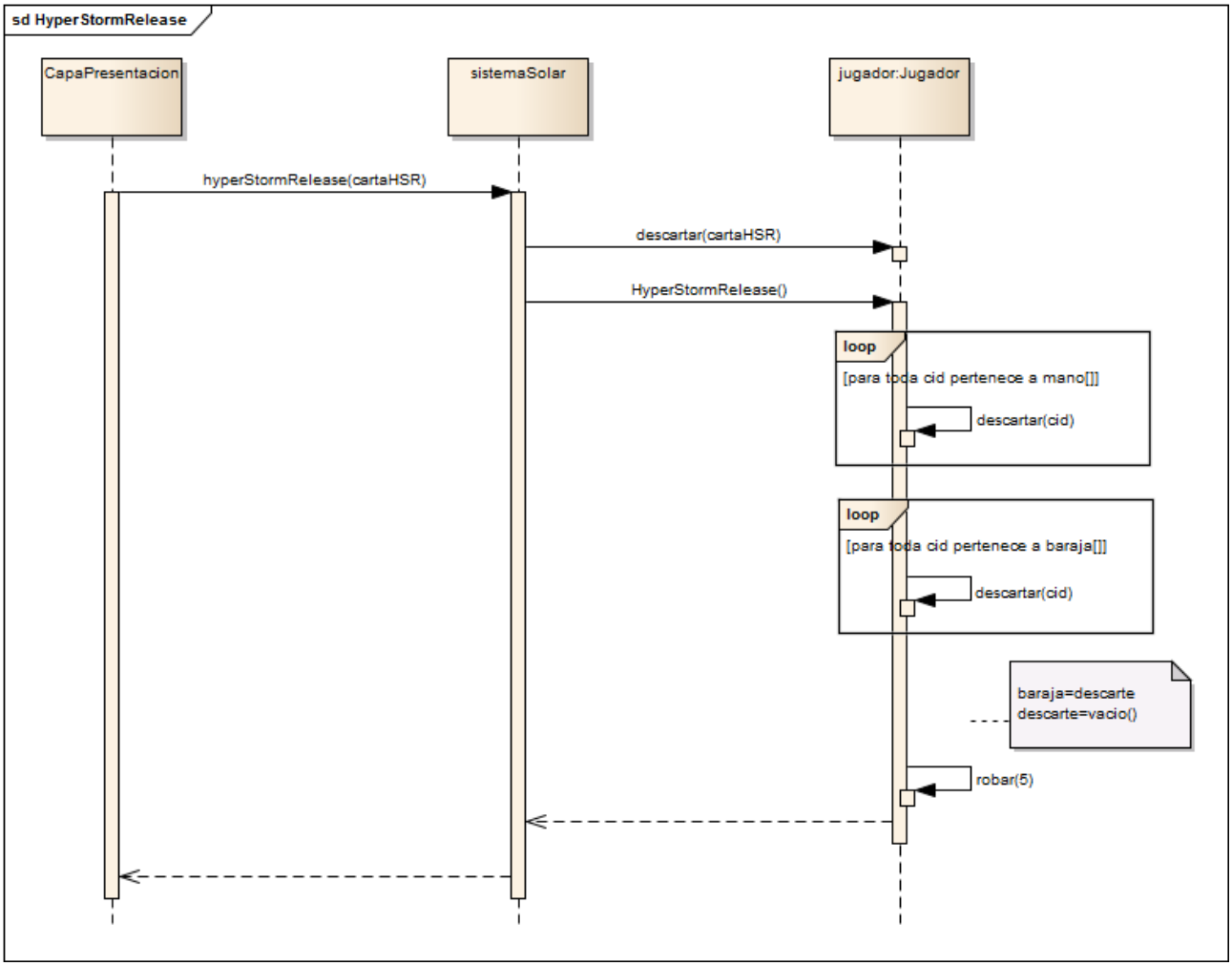
Context SistemaSolar :: hyperstormRelease(cartaHstorm:entero)

Pre:

- IP: Garantizada por Capa de Presentación
- Subtipo: Garantizada por intervención

Exc: ----

Post: […]



Esta función es delegada a la clase Usuario (Jugador) quien es la que posee la visibilidad necesaria para realizarla.

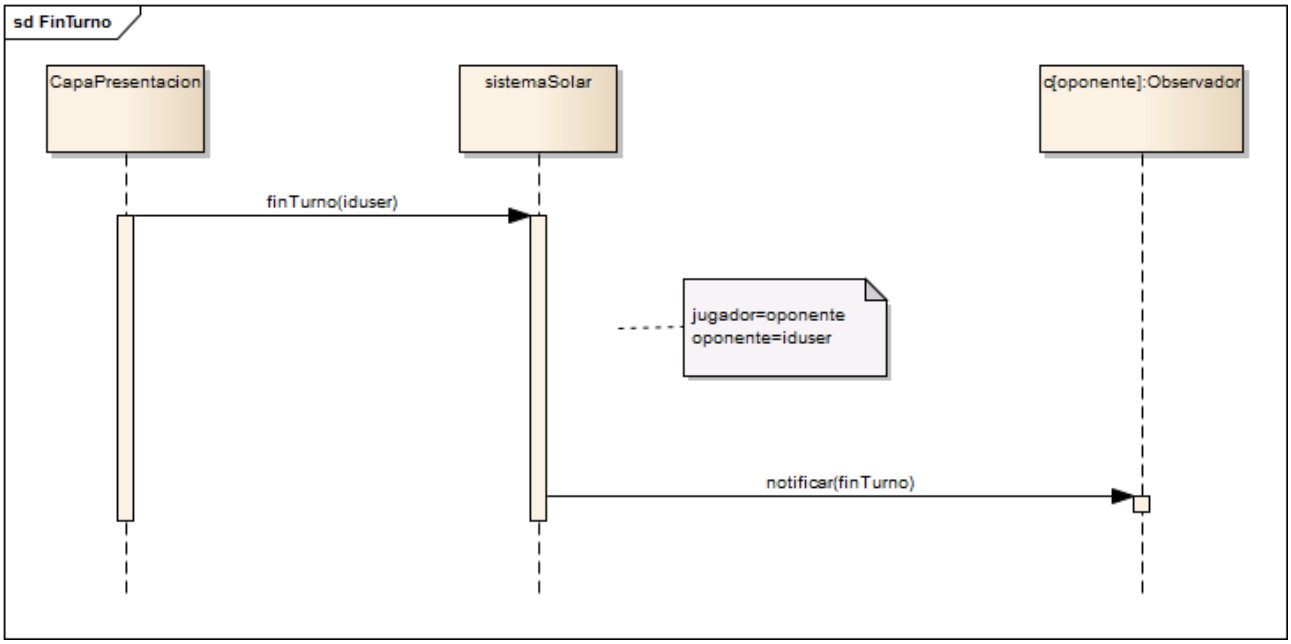
8. Fin Turno

Context finTurno(iduser:entero)

Pre:

- Jugador: El oponente esta bloqueado por la capa de presentación

Post: […]



Tras efectuar esta acción la capa de presentación se bloquea hasta que el observador es notificado de nuevo, por lo que es imposible que el usuario pueda realizar cualquier acción en este estado (salvo cancelar intervenciones)

Implementación

Implementación.....97
1. Presentación.....98
2. Dominio.....98
3. Datos.....99

1. Presentación

Para implementar la capa de Presentación hemos utilizado las funciones y objetos de la librería Java.swing.

Las cartas, planetas y demás elementos del juego han sido digitalizados e incorporados al juego, con la ayuda de la clase ImageIcon de dicha librería, para simular una experiencia similar a la del propio juego de mesa.

En pos de la utilidad, la interfaz cuenta con elementos de verificación, que se encargan de bloquear y ocultar todas aquellas funcionalidades del programa, que el usuario no puede usar en ese momento. Esto también implica que la interfaz se bloqueará completamente cuando no sea el turno del usuario. A pesar de que esto ha aumentado la redundancia en la verificación del programa, también ha supuesto en una sinergia positiva a la seguridad del conjunto.

Para ver más detalles acerca del funcionamiento de la capa de presentación consultar el artefacto Diseño de la Interficie Gráfica

2. Dominio

La lógica del programa ha sido implementada en Java estándar siguiendo el diagrama de clases de Diseño. La capa de presentación consta de las siguientes clases:

CtrlDominio (a.k.a. SistemaSolar): Conjunto de elementos que representan el estado del juego Principal punto de comunicación con la capa de presentación y encargado de arbitrar la comunicación entre usuarios.

CtrlDom_0: Stub de CtrlDominio. Hereda de la clase java.Remote, necesaria para implementar la comunicación online por Remote Method Invocation

Jugador: Conjunto de elementos que representa el estado de un Usuario: tecnologías poseidas, cartas, mazo, pila de descarte, pilas de recursos y acciones e intervenciones del presente turno. CtrlDominio es el encargado de determinar que Jugador tiene el turno mediante punteros

Planeta: Encargado de implementar las funciones relativas a las acciones planetarias

PilaRecursos: Conjunto de elementos que representan una pila de recursos: tipo, valor y cartas de recurso que la componen

Observador: Clase abstracta encargada de reportar al usuario los cambios sucedidos en CtrlDominio. Tambien hereda de la clase java.Remote, necesaria para implementar el observador remoto por java RMI

Observador_Humano: Implementación de la clase Observador adaptada para la comunicación con una interficie gráfica

Observador_IA: Implementación de la clase Observador adaptada para la comunicación con una Inteligencia Artificial

Observador_Remoto: funciona igual que Observador_Humano, pero con funciones de comunicación online.

3. Datos

La capa de datos propiamente dichas no existe, ya que en ningún momento hay persistencia de datos.

Al principio de la ejecución del programa, los Controladores de la capa de datos leen y parsean dos archivos de texto que contienen la información relativa a los recursos y cartas en el siguiente formato

indice del recurso: Tipo del anverso (valor 3) / Tipo del reverso (valor 2)

indice de la carta: Nombre: Descripción de la carta: Información del apartado de Ayuda

Ejemplo:

2: Y / R

4: Motor de fusión : Suma 1 a las unidades de energía disponibles por vuelo. : Cada "Motor de fusión" incrementa el numero de unidades de energía por vuelo en 1.

A partir de estas entradas se crea una tabla indexada sobre la cual se efectuarán las operaciones consultoras de la Capa de Datos.

Para ver una lista completa de las cartas y recursos consultar el apartado Base de Datos en el Anexo de este documento.

Estudio de la Inteligencia Artificial

Estudio de la Inteligencia Artificial....100
1. Estudio del factor de ramificación....101
2. Núcleo y Nodos....102
2.1. Poda....102
2.2. Vuelo....103
3. Tipos de heurístico....104
3.1. IA_Random....105
3.2. IA_Greedy....106
3.3. IA_Safe....108
4.Testing....109
4.1. Oscilación....110
4.2. Porcentaje de Victorias....115

1. Estudio del factor de ramificación

Debido a la naturaleza de este juego, una inteligencia artificial propiamente dicha, requeriría una búsqueda heurística (cada turno) y un minimax (global). Sin embargo, debido al gran factor de ramificación resultaría imposible realizar tal operaciones en un tiempo de computación aceptable

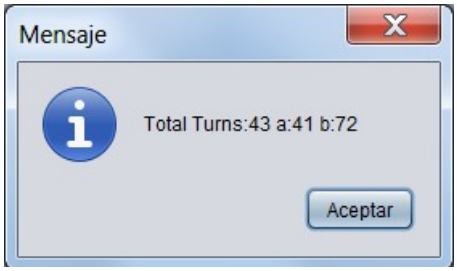
Fr=((#recursos diferentes del planeta+#contenedores del jugador)$^{\#acciones\ planetarias}$) * (#ndestinos_posibles$^{\#vuelos}$) * (#cartasjugador$^{\#intervenciones}$)

=((5+3)^2*17^1*5^2)=64*17*25=27200

Hay que considerar que el orden en que se realizaran esas acciones alteraría el producto del conjunto. Además, factor multiplicativo de las intervenciones sería mucho mayor ya que en ocasiones implican nuevas acciones planetarias o vuelos.

Por ejemplo: la intervención Afterburner Ignition, aumentaría el componente de vuelo de 17 a 289. En caso de realizarse una segunda intervención afterburner ignition, este factor aumentaría hasta 4913 (En el peor de los casos... También habría que tener en cuenta que este factor se reduciría según las unidades de energía proporcionadas por la propia acción de vuelo)

Y si tenemos en cuenta que el numero medio de turnos de una partida oscila entre los 30-80...



Partida aleatoria Greedy(a) vs Safe(b)

Para solucionar esto, se ha llegado a la conclusión de que reducir el espacio de búsqueda únicamente a las acciones que puede realizar el jugador en ese mismo instante. Consideramos únicamente la búsqueda de la siguiente acción y de las modificaciones que podría realizar al estado del juego. Esto reduce el factor de ramificación a:

Fr=((#recursos del planeta+#contenedores del jugador)*#acciones planetarias)+ (#destinos_posibles*#vuelos)+(#cartas del jugador*#intervenciones)

=(5+3)*2+17*1+5*2=43

Al no considerar acciones pasadas o futuras, el indeterminismo producido por las intervenciones se reduce notablemente.

Siguiendo el ejemplo anterior: La intervención Afterburner ignition, aumentaría a 34 la componente de vuelo ya que no consideraríamos una posible acción de vuelo anterior o siguiente

2. Núcleo y Nodos

Siguiendo el estudio anteriormente mencionado, se ha limitado el espacio de búsqueda a 33 acciones:

- **1-6** Cargar contenedor con recursos de tipo X
- **7-9** Descargar contenedor X
- **10-14** Usar la carta X
- **15-32** Volar hasta la posición X
- **33** Fin de turno (por defecto)

Durante su turno, la ia generará el estado correspondiente a cada una de estas 33 acciones y elegirá una de ellas dependiendo del resultado de la función heurística; repitiendo este bucle hasta que la opción de Fin de turno sea seleccionada por defecto.

2.1. Poda

Tal y como se especificaba en el documento "Análisis de requisitos", estas acciones requieren de unas condiciones específicas en el estado del jugador o el juego para ser realizadas. Esto conlleva a una poda en la generación de nodos que agilizará aún más el cálculo del heurístico mayor. A saber:

- **1-6** Condiciones generales Acción planetaria[10], Al menos un contenedor libre, el planeta dispone de recursos de tipo X
- **7-9** Condiciones generales Acción planetaria, contenedor con recursos del tipo solicitado
- **10-14 Tecnología:** Condiciones generales Acción planetaria, megagalax del jugador > coste
- **10-14 Intervención:** Intervenciones disponibles > 0, Condiciones concretas según el tipo de la intervención[11]
- **15-32** El jugador dispone de su acción de vuelo
- **33** por defecto

10 El Jugador se encuentra sobre un planeta (o en la órbita de este si el jugador dispone de la tecnología Orbital Station) y dispone de almenos una Acción planetaria restantes

11 Ver Análisis de requisitos. Punto 5: Intervenciones

2.2. Vuelo

La acción "Volar hasta la posición X" cuenta con la incertidumbre[12] de no saber el número de unidades de energía disponibles a priori (se calculan aleatóriamente) y por extensión el rango de vuelo. Esto implica que en algunos casos, la acción de vuelo seleccionada no se pueda realizar.

Para solucionar esto, el sistema volará hasta la posición más cercana posible al objetivo dentro del rango especificado por las unidades de energía.

Adicionalmente, la inteligencia artificial de tipo "Safe" tiene en cuenta este hecho, por lo que pondera el valor del heurístico del vuelo en función de el numero de unidades de energía disponible que requeriría volar hasta esa posición

12 En teoría el sistema sí que podría calcularla, pero para ser justos con el usuario, añadimos este tipo de incertidumbre ya que el usuario no cuenta con la posibilidad de ver el futuro (deshacer la acción de vuelo)

3. Tipos de heurístico

Tras analizar las acciones posibles, se ha de definir una función heurística que determine numéricamente el valor de cada nodo, para así poder encontrar la acción adecuada en cada momento. Para ello, tenemos que llevar a cabo un ejercicio de abstracción para determinar que elementos proporcionan valor a los nodos. Esto nos lleva a clasificar las acciones posibles en dos tipos:

1. Proporcionan beneficios de manera directa o indirecta: Todos los tipos de acción planetaria
2. Ofrecen la posibilidad de obtener beneficios en un futuro: Vuelo

Dependiendo del tipo de acción, los parámetros que proporcionan valor al heurístico serán diferentes. Esto implica que tendremos que implementar 2 heurísticos diferentes

En el primer caso, los parámetros que proporcionan valor a la acción sería:

- Numero de puntos actual del jugador y la diferencia de puntos respecto al estado anterior (ya sea negativa o positiva)
- Valor del los recursos almacenados en los contenedores del jugador tras realizar esa acción (puntos potenciales)
- Estado del planeta tras realizar dicha acción

En el segundo caso, estos parámetros serían:

- Valor de los recursos del planeta al que se pretende volar (para una posterior acción de carga)
- Coste de volar hasta dicho planeta.
- Tipo de recurso solicitado por el planeta en cuestión y el tipo de contenido de los recursos del jugador (para una posterior acción de descarga)

En lo que respecta a las intervenciones, todas ellas pueden ser traducidas a uno de estos dos tipos de acciones[13] o combinación, por lo que aprovecharemos los heurísticos implementados. Sin embargo, puesto que las intervenciones requerirán que el usuario se desprenda de una de sus cartas, se añadirá una pequeña penalización al heurístico, en concepto de coste de la carta.

13 Primer tipo: Passenger, Agent infiltration, Afterburner Ignition, Inner/Outer Loading, On-board Transmitter
Segundo tipo: Afterburner Ignition, Sun Tunneling, Transmitter Gate Activation

3.1. IA_Random

En esta implementación, el heurístico se calcula de manera completamente aleatoria ignorando completamente los parámetros anteriormente mencionados. A pesar de ello, se le ha dado cierta prioridad al primer tipo de heurístico debido a la voracidad del juego.

- h_1=*Random[0..50]*
- h_2=*Random[0..30]*
- *bonificación/Penalización intervención=Random[-25..25]*

Implementación simple que representaría la cota inferior en lo que respecta a la eficiencia de los heurísticos. Suponiendo una muestra de 10000 partidas de dos IA_Random enfrentadas[14].

- **Numero medio de turnos de una partida:** 342 turnos
- **Porcentaje de victorias:** 49.56%
- **Porcentaje de partidas terminadas por time out (1000 turnos):** 0.35%

Una de las particularidades de la función aleatoria es que tarde o temprano, siempre acaba encontrando una solución. De no haber establecido un límite de turnos, ese 0.35% de casos, habrían acabado convergiendo. Por ello, para asegurarnos la convergencia de el resto de heurísticos, añadimos un pequeño componente de aleatoriedad (*Random[0..5]*) a los heurísticos restantes

14 Ver apartado testing para comparaciones más exhaustivas entre IAs

3.2. IA_Greedy

En esta implementación, el heurístico se calcula vorazmente, buscando aquellas acciones que proporcionen mayores beneficios inmediatos. Dada la naturaleza voraz de este juego, esta era la opción más obvia

- h_1=Dependiendo de la diferencia de puntos respecto al estado anterior
  - Si diferencia positiva: $h_1=3*(\Sigma_0^{ncont} valor_contenedorjugador[i] - max(valor_recursoplaneta) + diferencia*2)$
  - Si diferencia negativa: $h_1=(puntos_totales+diferencia/2)$
- $h_2=\Sigma_0^{nrecursos} valor_recursoplaneta[i]+18*ncontenedores_recursosolicitado$
- *Bonificación/Penalización intervención=Random(-10..40)*

Respecto al primer heurístico, el hecho de separar en 2 subcasos dependiendo de la diferencia de puntos respecto al estado interior, se debe a que una diferencia negativa, implica que estamos gastando puntos comprando una tecnología; mientras que una positiva, implica que estamos obteniendo puntos al descargar un contenedor. Al cargar un contenedor, obtenemos puntos potenciales, mientras que al descargarlo, convertiremos esos puntos potenciales en puntos reales, los cuales obviamente tendrán más peso en el heurístico que los potenciales. Finalmente, cabría mencionar el hecho de que si existe en el planeta un recurso con un valor mayor al que estamos cargando, obviamente no estaremos realizando la acción más voraz, de ahí la penalización que se le resta al primer heurístico. Por ejemplo:

- *Dado un planeta con 2 tipos de recursos diferentes:*
  - *valor 9, distancia de al destino irrelevante*
  - *valor 3, distancia al destino irrelevante*

El heurístico del primer caso sería 3(9-3+0)=18, mientras que el del segundo caso seria 3*(3-9+0)=-18. Por lo que acabaríamos cargando el primer tipo de recurso*

El segundo heurístico es bastante más intuitivo: El jugador siempre volará hacia el planeta en el que tenga la posibilidad de obtener un mayor número de puntos, es decir, el planeta que tenga más recursos para cargar, o el planeta donde pueda descargar más recursos. Lógicamente, este último tendrá un peso[15] mucho mayor, ya que de no liberarse el contenedor, no podría volver a ser usado y el jugador quedaría bloqueado.

Suponiendo una muestra de 50000 partidas de dos IA_Greedy enfrentadas[16].

- **Numero medio de turnos de una partida:** 52.63 turnos
- **Porcentaje de victorias del primer jugador:** 54.21%
- **Porcentaje de partidas terminadas por time out:** 0.11%

Esta heurístico tiende a funcionar bastante bien en las partidas contra humanos, aunque puede ser

15 Se le ha decidido dar peso 18 a este factor ya que es el número máximo de recursos que puede llegar a tener un planeta (30 cartas de recurso * 3 (anverso)/5 tipos recursos posibles) la posibilidad de descargar un contenedor, simpre tendra un peso mayor que la posibilidad de cargarlo

16 Ver apartado testing para comparaciones más exhaustivas entre IAs

anulado fácilmente por jugadores experimentados ya que es bastante predecible. Debido a la voracidad de éste, en igualdad de condiciones, proporciona una pequeña bonificación al jugador que juega en primer lugar

3.3. IA_Safe

Al contrario que IA_Greedy, en esta implementación el heurístico se calcula reduciendo la incertidumbre de las acciones futuras al mínimo. Esto implica, que buscamos reducir la distancia que tendrá que recorrer en los vuelos y, por consiguiente, la posibilidad que quedarse sin energía en mitad de un vuelo

- h_1=Dependiendo de la diferencia de puntos respecto al estado anterior
  - Si diferencia positiva: $h_1=|pos_jugador-pos_solicita[maxrecurso]|*(\Sigma_0^{ncont} valor_contenedorjugador[i] + diferencia*2)$
  - Si diferencia negativa: $h_1=(puntos_totales+diferencia)/2$
- $h_2=1/2*\Sigma_0^{nrecursos} valor_recursoplaneta[i]+18*ncontenedores_recursosolicitado - costevuelo$
- *Bonificación/Penalización intervención=Random(-20..30)*

En lo que respecta al primer heurístico, separaremos igualmente entre comprar tecnologías y cargar/descargar recursos. A pesar de que buscaremos cargar el recurso de mayor valor, este máximo vendrá ponderado por la distancia que lo separa al jugador del planeta de destino del recurso. Por ejemplo:

- *Dado un planeta con 2 tipos de recursos diferentes:*
  - *valor 9, distancia de al destino 4*
  - *valor 3, distancia al destino 1*

El heurístico de cargar el primer contenedor sería 1(9+0)=9, mientras que el del segundo sería 4*(3+0)=12. Por lo que acabaremos cargando el segundo tipo de recurso*

El segundo heurístico es bastante similar al de IA_Greedy, pero añadiendo el coste del vuelo al calculo. Cuanto menor sea el coste del vuelo, menos posibilidades habrá de que el jugador no disponga de unidades de energía suficientes para realizarlo[17]. Sin embargo, la posibilidad de descargar contenedores, seguirá teniendo más peso por las razones anteriormente mencionadas

Dada una muestra de 50000 partidas entre dos IA_Safe enfrentadas[18].

- **Numero medio de turnos para terminar una partida:** 51.09 turnos
- **Porcentaje de victorias del primer jugador:** 55.64%
- **Porcentaje de partidas acabadas en time out:** 0.002%

Al reducir la incertidumbre, IA_Safe es un heurístico mucho más estable y difícil de anular que los 2 anteriores. Sin embargo en la práctica, debido a la naturaleza voraz del juego, esta estabilidad no representa una mejora substancial sobre IA_Greedy, quien aparentemente sigue manteniendo una eficiencia similar

17 Recordemos el punto 2.2

18 Ver apartado testing para comparaciones más exhaustivas entre IAs

4.Testing

En este apartado se realizarán comparaciones entre IA dadas muestras de 50000 partidas: consecutivas.

Leyenda

- **Tm:** numero medio de turnos de la partida y [máximo..mínimo]
- **V1:** Porcentaje de victorias del primer jugador
- **O:** Porcentaje de partidas acabadas en time out (1000 turnos)
- **R**: Porcentaje de partidas acabadas por agotamiento de recursos
- **E:** Porcentaje de empates

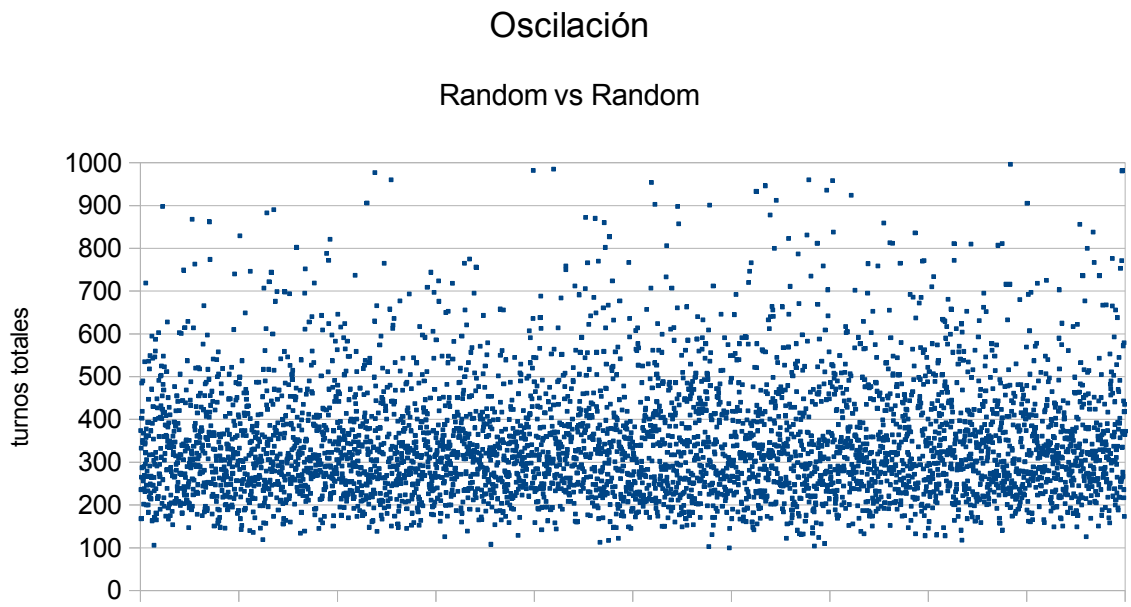
J1 \ J2	Random	Greedy	Safe
Random	**Tm:** 342 [77..998] **V1:** 49.56% **O:** 0.35% **R:** 3.38% **E:** 0.085%	**Tm:** 69.97[25..997] **V1:** 1.82% **O:** 0.19% **R:** 0.32% **E:** 0.002%	**Tm:** 52.92 [23..159] **V1:** 0.002% **O:** 0% **R:** 0.017% **E:** 0%
Greedy	**Tm:** 69.38 [24..991] **V1:** 98.14% **O:** 0.19% **R:** 0.29% **E:** 0.002%	**Tm:** 52.63 [26..620] **V1:** 54.21% **O:** 0.11% **R:** 7.73% **E:** 0.22%	**Tm:** 51.22 [25..100] **V1:** 46.27% **O:** 0% **R:** 8.12% **E:** 0.26%
Safe	**Tm:** 51.83 [22..207] **V1:** 99.996% **O:** 0% **R:** 0.015% **E:** 0%	**Tm:** 50.73 [22..236] **V1:** 61.57% **O:** 0% **R:** 7.49% **E:** 0.26%	**Tm:** 51.09 [24..221] **V1:** 55.64% **O:** 0.002% **R:** 5.81% **E:** 0.19%

4.1. Oscilación

En este apartado estudiaremos la desviación respecto a la media que existe en el numero de turnos necesarios para terminar una partida dependiendo de el tipo de IA.

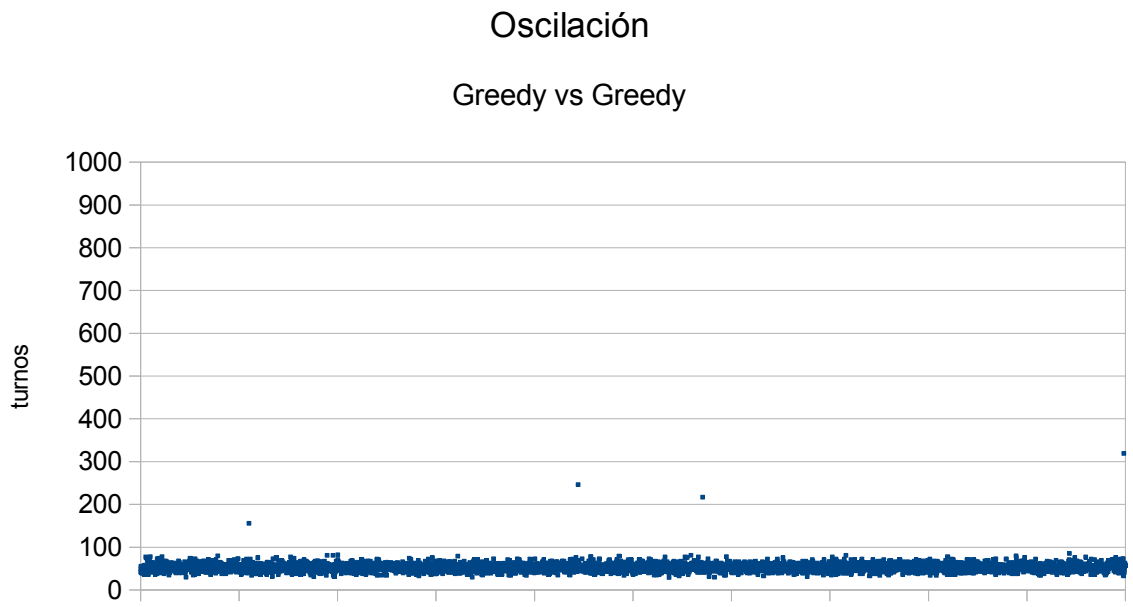
4.1.1 IA_Random

Una función heurística basada en la aleatoriedad no puede ser estable, así que, como era de esperar, es la que ofrece una dispersión mayor respecto a la media (342 turnos). Nótese que de no haber establecido el time out de 1000 turnos, la dispersión se propagaría mucho más allá

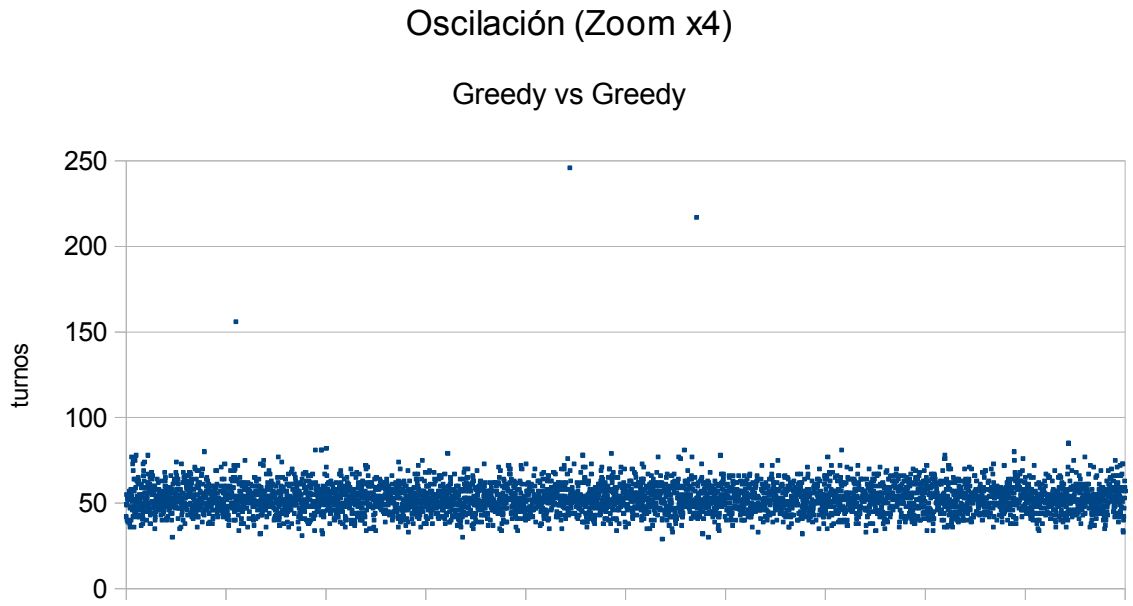


4.2.2. IA_Greedy

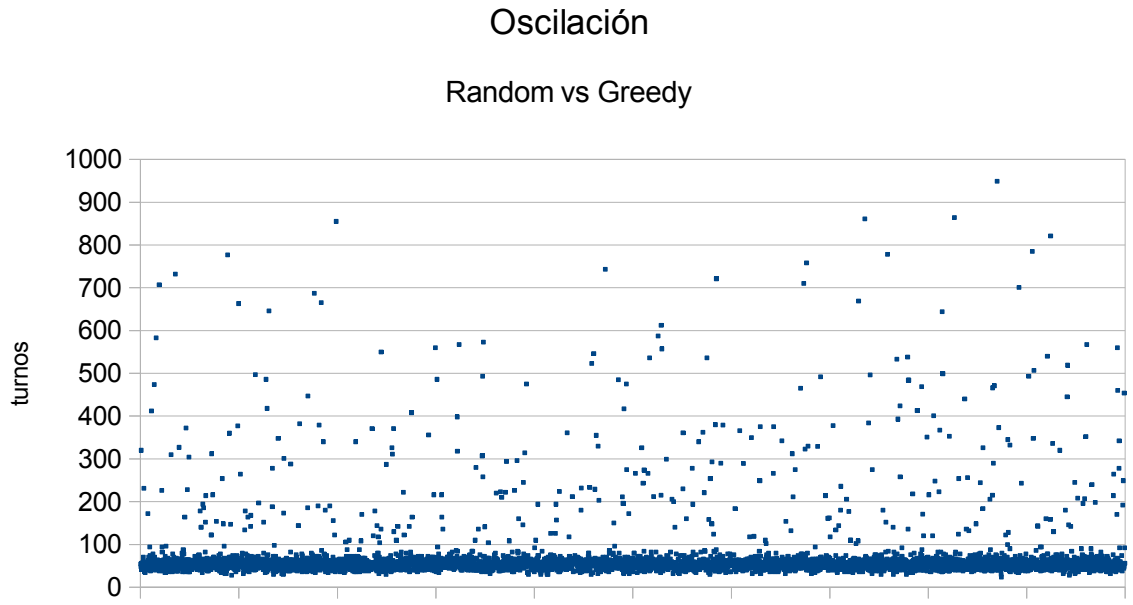
Ofrece una mejora significativa en lo que a la dispersión se refiere, pero todavía se detectan algunos outlayers. Además, debido a la voracidad de esta IA, al enfrentar al enfrentar a dos oponentes de este tipo, la probabilidad de llegar a un punto muerto se reduce aún más



En el siguiente gráfico, se aumenta la escala del eje Y para facilitar la comprensión y comparación de los resultados. Observamos que a excepción de los outlayers, el numero medio de turnos para terminar una partida oscila entre 30 y 80

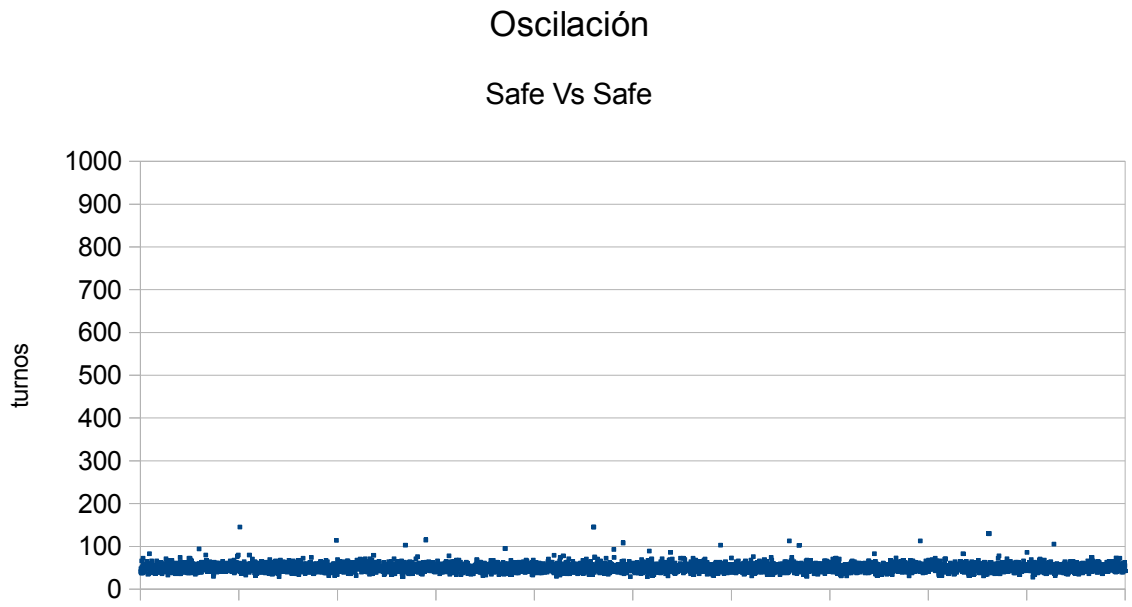


Sin embargo, esta voracidad, también hace que la estabilidad del heurístico se resienta... y si además, añadimos la IA_Random, que ya de por sí es inestable, a la ecuación, la situación empeora notablemente.

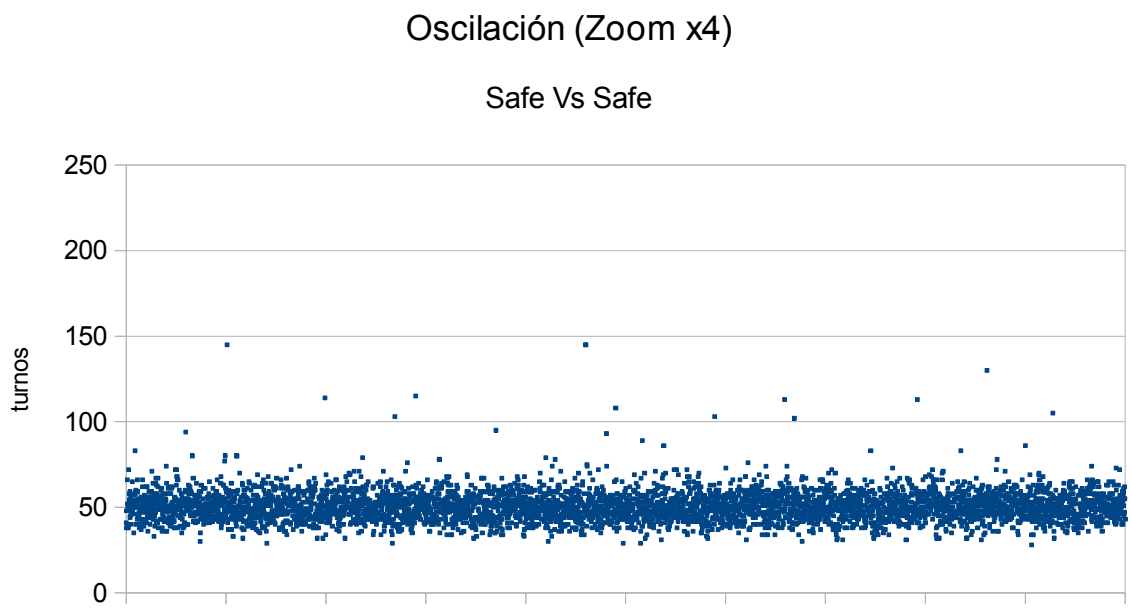


4.1.3. IA_Safe

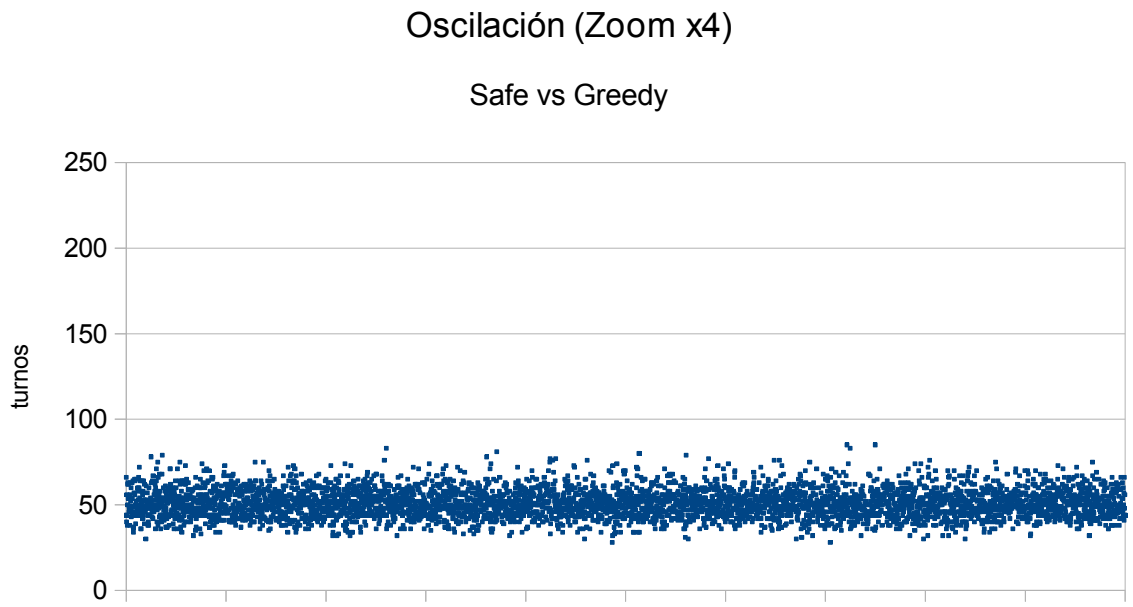
El objetivo de esta IA era el de reducir la incertidumbre de las acciones futuras. Esto ha reducido al mínimo los outlayers del algoritmo.

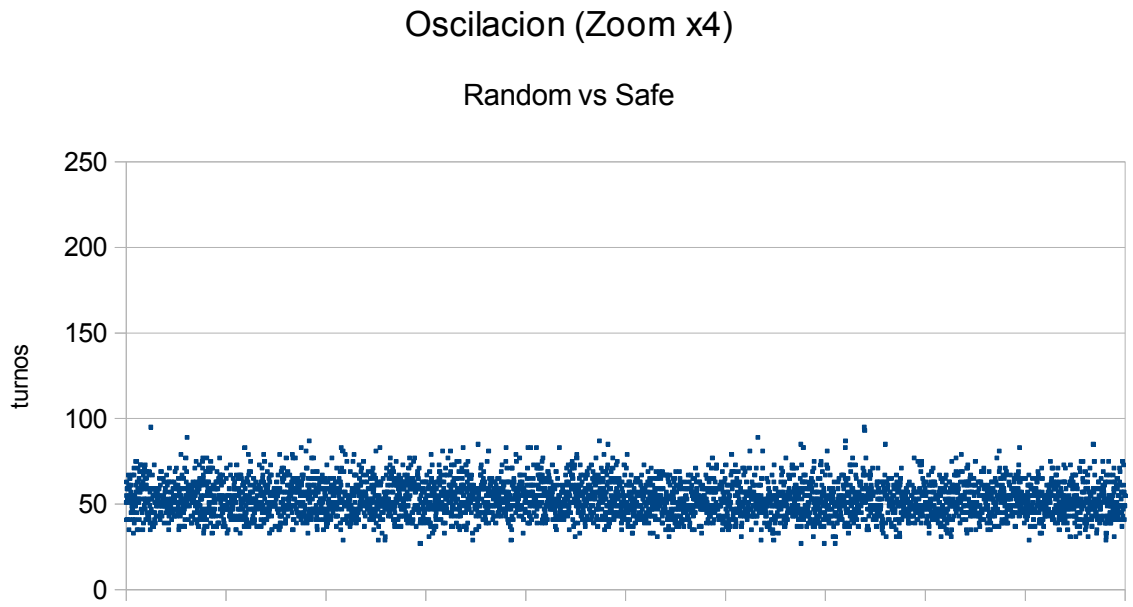


Al contrario que la IA_Greedy, al enfrentar dos IAs de este tipo, se reduce el movimiento de recursos entre planetas, es decir, que las IAs se anulan entre sí provocando en ocasiones un punto muerto que hace que el numero de turnos necesarios para terminar una aumente. Esto provoca una dispersión respecto a la media ligeramente mayor que en el caso IA_Greedy vs IA_Greedy.



No obstante, al reducir la incertidumbre, aumentamos la estabilidad; prueba de ello es que si forzamos el movimiento de recursos enfrentándola contra cualquier otro tipo de IA, esta dispersión desaparece completamente





4.2. Porcentaje de Victorias

En este apartado estudiaremos los porcentajes de victorias de los diferentes tipos de IAs y buscaremos las causas de estos basándonos en las peculiaridades de cada heurístico

4.2.1. IA_Random vs

Una de las particularidades de los números aleatorios es que son justos, es decir, no ofrecen ninguna ventaja a ninguno de los jugadores en particular. Por ello el porcentaje de victorias para cada jugador al enfrentar a dos inteligencias artificiales de este tipo, tiende al 50%

49.56% primer jugador y 50.35% segundo jugador (0.09% empates)

Como era de esperarse, las probabilidades de victoria de una función aleatoria frente a un heurístico con algun tipo de inteligencia son bastante deplorables

- **Random vs Greedy:** 1.82% victorias, 0.002 % empates
- **Random vs Safe:** 0.002 % victorias, 0 % empates

Recordemos que IA_Random agravia la inestabilidad de IA_Greedy[19], de ahí que aumenten sus posibilidades de victoria

4.2.2. IA_Greedy vs

La voracidad del algoritmo se complementa con la voracidad del juego, otorgando una pequeña bonificación a este tipo de IA si juega en primer lugar. De ahí que al enfrentar a dos IAs de este tipo, aquella que juega en primer lugar se la que obtenga mejores resultados

54.21% primer jugador y 45.57% segundo jugador (0.22% empates)

Por lo demás, ofrece un rendimiento bastante respetable, a pesar de ser voraz:

- **Greedy vs Random:** 98.14% victorias, 0.002% empates
- **Greedy vs Safe:** 46.27% victorias, 0.26% empates

4.2.3. IA_Safe vs

Como vimos en el apartado anterior, este tipo de IA es el más estable. Esto se traduce a una mayor resistencia a las condiciones adversas provocadas por el resto de IAs, es decir, que "anula" a sus oponentes.

- **Safe vs Random:** 99.994% victorias, 0% empates
- **Safe vs Greedy:** 61.57% victorias, 0.26% empates

A pesar de ello, este tipo de IA se basa en un máximo ponderado, así que, en cierto modo, actua de forma voraz. Por lo tanto, al enfrentarlo contra una IA del mismo tipo, se puede observar un comportamiento similar al mostrado por IA_Greedy

55.64% primer jugador, 44.17% segundo jugador (0.19% empates)

19 Ver apartado 4.1.2 Oscilación IA_Greedy

Progreso del Proyecto

Progreso del Proyecto....116
1. Introducción....117
2. Tareas....118
2.1. Análisis de requisitos....118
2.2. Especificación....118
2.3. Diseño....118
2.4. Implementación....118
2.5. Validación....119
2.6. Documentación....119
3. Desviaciones....121
3.1 Interficie gráfica....122
3.2 Transaction Script → Domain Model....122
4. Recursos....123
4.1 Asignación de recursos a tareas....124
4.2. Coste....125
5. Situación actual....126
6. Mejoras....127
6.1. Inteligencia artificial....127
6.2. Jugador vs Jugador....127

1. Introducción

Como ya se había especificado en el informe preliminar, este proyecto se ha desarrollado utilizando utilizando el método de desarrollo de software en cascada

Análisis de Requisitos → Especificación → Diseño → Implementación → Testing

1. Análisis de Requisitos: Se examina a conciencia el conjunto del juego y se determinan los elementos principales que lo componen amén de las restricciones que definen y condicionan el flujo del juego.
2. Especificación: Se realiza una abstracción de los elementos anteriormente definidos y se definen los casos de uso principales así como los flujos de acciones de éstos.
3. Diseño: Se establece la separación por capas y se determina que elemento realizará cada acción. Una vez hecho esto se podrán concretar los casos de uso anteriormente definidos,
4. Implementación: Se plasma en código aquello definido en las anteriores etapa.
5. Validación (o Testeo): Se pondrá a prueba el programa para asegurarnos de que cumple los requisitos establecidos.
6. Finalmente se procederá a hacer una compilación de los documentos generados por cada una de las etapas realizando las correcciones y adecuación de formato correspondientes.

2. Tareas

Siguiendo el esquema en cascada, hemos subdividido cada etapa del proyecto en una serie de subetapas o tareas para calcular más fácilmente la duración del proyecto:

2.1. Análisis de requisitos

1. **Análisis Previo del Problema:** Estudiar las reglas, elementos del juego, determinar el flujo normal de una partida, etc.
2. **Determinar elementos básicos del juego:** Hacemos un ejercicio de abstracción para determinar y clasificar los elementos que componen el juego (Jugador, Planeta, etc.)
3. **Clasificación y estudio de restricciones:** Estudio en profundidad de las reglas del juego en el que determinamos qué, cómo, cuándo y a quién afecta cada regla
4. **Análisis de requisitos:** artefacto que resume las tareas anteriores

2.2. Especificación

5. **Elaborar Modelo conceptual:** una vez determinados los elementos básicos, los clasificamos en objetos y operaciones
6. **Definición de Casos de Uso:** Determinar los flujos de acciones que se producen durante el juego y sus condicionantes
7. **Asignación de responsabilidades a capas:** Determinamos a qué capa pertenece cada acción en un esfuerzo por distribuir responsabilidades
8. **Casos de uso:** artefacto que resume todo el proceso de especificación

2.3. Diseño

9. **Diseñar Diagrama de clases de dominio:** Concretamos el modelo conceptual de la anterior fase y aplicamos patrones de diseño
10. **Diseñar screens de la Capa de Presentación:** Concretamos los elementos sobre los que interactuará el usuario en la capa de presentación y las acciones asociados a ellos
11. **Concretar casos de uso:** Traducimos los casos de uso abstractos de la etapa anterior a casos de uso concretos sobre nuestra capa de presentación
12. **Asignación de responsabilidades a objetos:** determinamos qué acciones hace cada objeto y qué requiere para hacerlas
13. **Elaborar diagramas de secuencia:** concretamos el flujo de cada una de las acciones y los elementos que comunican
14. **Diseño:** artefacto que resume todo el proceso de diseño

2.4. Implementación

15. Capa de presentación

1. **Digitalización del material:** escaneamos los elementos físicos del juego para integrarlos en nuestro programa
2. **Creación de la interfaz de usuario:** Implementamos la capa de presentación

16. Capa de datos
    1. **Elaboración del sistema de ayuda:** creamos el sistema de ayuda, a partir del material previamente digitalizado
    2. **Serialización de las clases de datos:** creamos la base de datos, indexamos los elementos y añadimos la información referente a ellos
17. Capa de dominio:
    1. **Implementación de las clases básicas del juego:** traducimos a código los diagramas de clases y secuencia de la etapa anterior
    2. **Inclusión del sistema Jugador vs Jugador:** añadimos al programa la lógica necesaria para implementar esta función (juego online)
    3. **Implementación de la inteligencia artificial:** implementamos una inteligencia artificial con las funciones especificadas en la etapa anterior

2.5. Validación

18. **Testeo del sistema jugador vs jugador**
19. **Testeo de la inteligencia artificial**
20. **Testing:** artefacto que resume esta etapa

2.6. Documentación

21. **Recapitulación de documentos:** unificamos el formato de los artefactos
22. **Actualización de la memoria del proyecto:** añadimos a la memoria los cambios en la planificación que hayan podido ocurrir durante el proyecto

Dadas las tareas anteriormente mencionadas, estimamos la planificación siguiente

Nombre de tarea	Duración
Analisis de requisitos	**9 días**
Análisis Previo del Problema	2 días
Determinar elementos básicos del juego	3 días
Clasificación y Estudio de restricciones	4 días
Análisis de requisitos	0 días
Especificación	**17 días**
Elaborar Modelo conceptual UML	3 días
Definición de Casos de uso abstractos	7 días
Asignación de responsabilidades a capas	1 día
Análisis del comportamiento de la IA	7 días
Casos de uso	0 días
Diseño	**14 días**
Diseñar Diagrama de clases de Dominio	3 días
Diseñar screens Capa de presentación	2 días
Concretar Casos de Uso	3 días
Asignación de responsabilidades a objetos	1 día
Elaborar diagramas de secuencia	5 días
Diseño	0 días
Implementación	**35 días**
Capa de presentación	**9 días**
Digitalización de las cartas y planetas	2 días
Creación de la interfaz de usuario	7 días
Capa de datos	**4 días**
Elaboración del sistema de ayuda	2 días
Serialización de las clases de datos	2 días
Capa de dominio	**22 días**
Implementación de las clases básicas del juego	7 días
Inclusión del sistema de Jugador vs Jugador	4 días
Implementación de la inteligencia arfificial	15 días
Validación	**26 días**
Testeo del sistema jugador vs jugador	15 días
Testeo de la inteligencia arficicial	15 días
Testing	0 días
Documentación	**7 días**
Recapitulación de documentos	2 días
Actualización de la memoria del proyecto	5 días

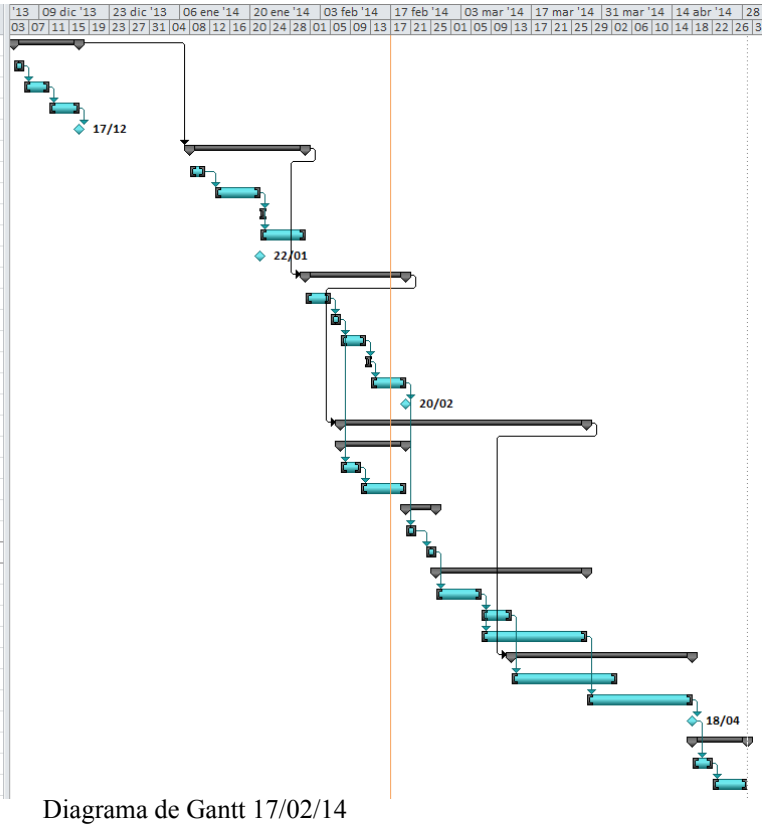


Diagrama de Gantt 17/02/14

3. Desviaciones

A lo largo del proyecto se han producido una serie de circunstancias adversas que han comprometido el desarrollo del proyecto.

- **10/2 Sistema multilingüe.** Añadir "Diccionarios" traductores de tecnologías e intervenciones para hacer a la capa de dominio independiente
- **21/2 Simplificar calculo de órbitas** respecto a posiciones planetarias (pos*2+6)
- **14/3 Diseño de Interficie gráfica demasiado complejo**, se necesita algo mas intuitivo para el usuario (limitar acciones del usuario a acciones que realmente puede hacer)
    - **26/3 Nueva interficie gráfica diseñada** incluido panel consultor a la izquierda de la interficie
    - **28/3 consultor de tecnologías** y recursos ocupa demasiado espacio. fusionar con panel lateral
- **6/4 Acciones planetarias poco intuitivas.** Delegar acciones planetarias a subinterfaz gráfica
- **18/04 Transaction Scripts no funciona.** Pasar el programa a Domain Model
    - **9/4 Si se realiza otra acción durante vuelo**, el sistema queda en un estado inconsistente. Atomizar acción de vuelo
    - **10/4 Sun Tunneling incompatible con acciones de vuelo atómicas.** Rediseñar operación vuelo para "simular" acciones de vuelo de varias fases
    - **12/4 Intervenciones no siempre son atómicas,** Algunas requieren de feedback del usuario. Aplicar el mismo concepto de la subinterfaz de las acciones planetarias a las Intervenciones Inner/Outer Loading y OnBoardTransmitter
    - **16/4 java.Swing no acepta multithreading.** Rediseñar comunicación entre jugadores por java RMI en lugar de sockets
- **7/5 Añadir subinterficie de log** para obtener algún feedback de las acciones realizadas por la IA (testing)
- **10/5 Función heurística tiene llanuras.** añadir componente aleatorio a función heurística

La mayoría de ellas sólo han sido problemas menores que han acabado implicando un pequeño retraso. Otras sin embargo, han requerido de la inclusión de nuevas tareas y en ocasiones de la repetición de tareas que ya habían sido acabadas

3.1 Interficie gráfica

“Diseño de la interficie gráfica demasiado complejo”, es un problema descubierto durante la primera iteración de la etapa de testing jugador vs jugador:

La interficie gráfica tenia demasiados elementos que consultar. Esto dificultaba la comprensión del jugador, y aumentaba notablemente el tiempo de ejecución, al tener que mantener actualizada esta interficie. Finalmente optamos por una interficie gráfica más compacta en la que se unificaban varios elementos a consultar en un panel lateral multifunción, y se delegaban las operaciones medianamente complejas a subinterfaces. Sin embargo, este hecho añadió 2 nuevas tareas al planning

1. Rediseñar la interficie gráfica: 20 horas
2. Adaptar el controlador a la nueva interficie: 12 horas
3. Repeticion del testing jugador vs jugador: +10 horas

3.2 Transaction Script → Domain Model

El contratiempo que supuso un mayor impacto en el proyecto ha sido la falta de compatibilidad de los controladores de tipo Transaction Script con nuestro programa:

Este problema se descubrió en la segunda iteración de la etapa testing jugador vs jugador a raiz de una serie de problemas aparentemente no relacionados que se iban acumulando sin saber exactamente la causa. En un principio, este programa había sido diseñado según el método de diseño Transaction Script (transiciones, acciones atómicas, estado). Sin embargo, la interficie gráfica java.Swing era incompatible con el método de transiciones por threads que habíamos utilizado hasta ahora. A causa de eso, tuvimos que optar por su alternativa, Domain Model en un solo thread. Un efecto colateral de esto, fue que la comunicación online entre jugadores se pasó de sockets a remote method invocation. Tareas añadidas:

1. Migrar a Domain model: 20 horas
2. Adaptar a Domain Model: 16 horas
3. Repetición del testing jugador vs jugador: + 15 horas

4. Recursos

Para realizar este proyecto han sido necesarios los siguientes recursos:

- **Analista:** Encargado de las tareas de Análisis de requisitos y Especificación. Requiere conocimientos básicos sobre UML, y una capacidad de abstracción
- **Arquitecto:** Encargado del Diseño. Conocimientos avanzados de UML y patrones de diseño
- **Diseñador GUI:** Encargado de diseñar la interficie gráfica de usuario. Conocimientos de usabilidad y estética amén de los conocimientos requeridos para diseñar la Interficie de Java. Tener en cuenta, que este profesional, necesitará un equipo más potente que los programadores, por lo que el coste de esté será mayor
- **Programador:** Encargado de Implementar el código. Conocimientos y experiencia en la plataforma en cuestión (Java). Cabe destacar, que hay que añadir al coste de este profesional, el coste del equipo que utilizará para implementar el programa
- **Tester:** Encargado de probar el programa y buscar bugs y fallos. Al igual que en el caso del programador, hay que añadir el coste del equipo utilizado
- **Becario:** Encargado de las tareas mundanas tales como escanear los recursos o unificar el formato de los artefactos

Suponemos que estos profesionales están trabajando en nuestra cartera de proyectos, por lo que eludimos los costes de contratación. Respecto a la jornada laboral, trabajarán a media jornada (4 horas/día). El coste de éstos es el siguiente:

Profesional	**Coste por hora**
Becario	15,00 €
Programador Java	23,00 €
Tester	20,00 €
Diseñador GUI	34,00 €
Analista	27,00 €
Arquitecto	30,00 €

4.1 Asignación de recursos a tareas

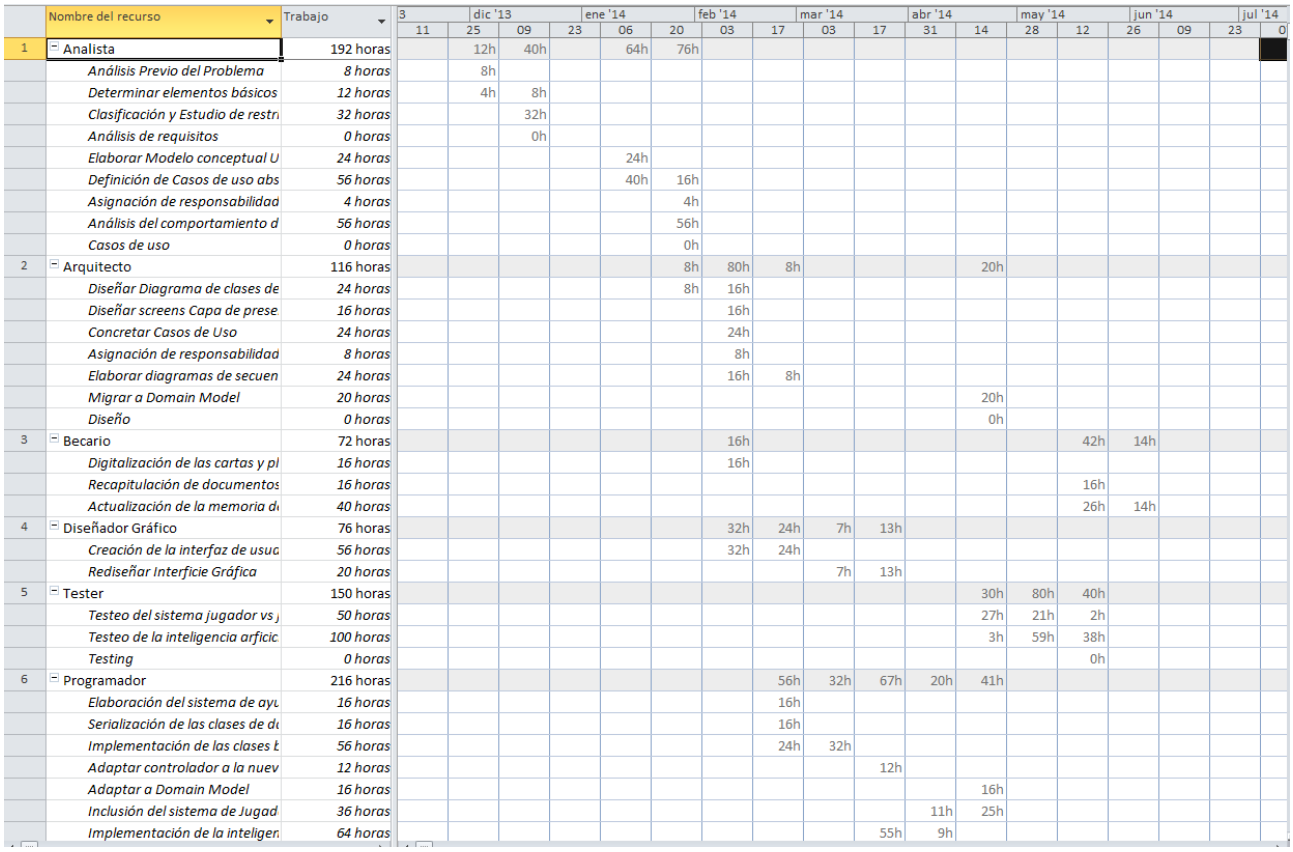
Los recursos han sido asignados a las tareas de la siguiente manera:

	Nombre de tarea	Duración	Comienzo	Trabajo	Fin	Predecesora	Nombres de los recursos
1	**Perry Rhodan: The Cosmic League**	**249,5 días**	**mié 04/12/13**	**822 horas**	**mar 27/05/14**		
2	**Analisis de requisitos**	**18 días**	**mié 04/12/13**	**52 horas**	**mar 17/12/13**		
3	Análisis Previo del Problema	2 días	mié 04/12/13	8 horas	mié 04/12/13		Analista
4	Determinar elementos básicos del juego	6 días	vie 06/12/13	12 horas	mar 10/12/13	3	Analista
5	Clasificación y Estudio de restricciones	8 días	mié 11/12/13	32 horas	lun 16/12/13	4	Analista
6	Análisis de requisitos	0 días	mar 17/12/13	0 horas	mar 17/12/13	5	Analista
7	**Especificación**	**36 días**	**mié 08/01/14**	**140 horas**	**vie 31/01/14**	**2**	
8	Elaborar Modelo conceptual UML	6 días	mié 08/01/14	24 horas	vie 10/01/14		Analista
9	Definición de Casos de uso abstractos	14 días	lun 13/01/14	56 horas	mar 21/01/14	8	Analista
10	Asignación de responsabilidades a capas	2 días	mié 22/01/14	4 horas	mié 22/01/14	9	Analista
11	Análisis del comportamiento de la IA	14 días	jue 23/01/14	56 horas	vie 31/01/14	9	Analista
12	Casos de uso	0 días	vie 31/01/14	0 horas	vie 31/01/14	11	Analista
13	**Diseño**	**111,25 días**	**vie 31/01/14**	**116 horas**	**vie 18/04/14**		
14	Diseñar Diagrama de clases de Dominio	6 días	vie 31/01/14	24 horas	mar 04/02/14		Arquitecto
15	Diseñar screens Capa de presentación	4 días	mié 05/02/14	16 horas	jue 06/02/14	14	Arquitecto
16	Concretar Casos de Uso	6 días	vie 07/02/14	24 horas	mar 11/02/14	15	Arquitecto
17	Asignación de responsabilidades a objetos	2 días	mié 12/02/14	8 horas	mié 12/02/14	16	Arquitecto
18	Elaborar diagramas de secuencia	6 días	jue 13/02/14	24 horas	lun 17/02/14	17	Arquitecto
19	Migrar a Domain Model	9 días	lun 14/04/14	20 horas	vie 18/04/14	17	Arquitecto
20	Diseño	0 días	vie 18/04/14	0 horas	vie 18/04/14	19	Arquitecto
21	**Implementación**	**105,25 días**	**vie 07/02/14**	**308 horas**	**mar 22/04/14**		
22	**Capa de presentación**	**55,25 días**	**vie 07/02/14**	**92 horas**	**mar 18/03/14**		
23	Digitalización de las cartas y planetas	4 días	vie 07/02/14	16 horas	lun 10/02/14	15	Becario
24	Creación de la interfaz de usuario	14 días	mar 11/02/14	56 horas	mié 19/02/14	23	Diseñador Gráfico
25	Rediseñar Interficie Gráfica	5 días	vie 14/03/14	20 horas	mar 18/03/14	24	Diseñador Gráfico
26	**Capa de datos**	**12 días**	**mar 18/02/14**	**32 horas**	**mar 25/02/14**		
27	Elaboración del sistema de ayuda	4 días	mar 18/02/14	16 horas	mié 19/02/14	18	Programador
28	Serialización de las clases de datos	4 días	lun 24/02/14	16 horas	mar 25/02/14	27	Programador
29	**Capa de dominio**	**79,25 días**	**mié 26/02/14**	**184 horas**	**mar 22/04/14**		
30	Implementación de las clases básicas	14 días	mié 26/02/14	56 horas	jue 06/03/14	28	Programador
31	Adaptar controlador a la nueva interficie	3 días	mar 18/03/14	12 horas	jue 20/03/14	25	Programador
32	Adaptar a Domain Model	4 días	vie 18/04/14	16 horas	mar 22/04/14	19	Programador
33	Inclusión del sistema de Jugador vs Jugad	9 días	jue 10/04/14	36 horas	jue 17/04/14	31	Programador
34	Implementación de la inteligencia arfifici	16 días	jue 20/03/14	64 horas	mar 01/04/14	31	Programador
35	**Validación**	**36,75 días**	**mar 22/04/14**	**150 horas**	**vie 16/05/14**	**21**	
36	Testeo del sistema jugador vs jugador	12,5 días	mar 22/04/14	50 horas	vie 16/05/14	33	Tester
37	Testeo de la inteligencia arficicial	25 días	mar 22/04/14	100 horas	vie 16/05/14	34	Tester
38	Testing	0 días	vie 16/05/14	0 horas	vie 16/05/14	37	Tester
39	**Documentación**	**14 días**	**vie 16/05/14**	**56 horas**	**mar 27/05/14**		
40	Recapitulación de documentos	4 días	vie 16/05/14	16 horas	mar 20/05/14	38	Becario
41	Actualización de la memoria del proyecto	10 días	mar 20/05/14	40 horas	mar 27/05/14	40	Becario

Las tareas producto de una desviación han sido marcadas en rojo.

4.2. Coste

En el siguiente gráfico se muestra la hoja de recursos, es decir, la cantidad de trabajo que realiza un recurso a lo largo del proyecto.



	Nombre del recurso	Trabajo	3	dic '13			ene '14			feb '14			mar '14			abr '14			may '14			jun '14			jul '14
			11	25	09	23	06	20	03	17	03	17	31	14	28	12	26	09	23	0
1	Analista	192 horas		12h	40h		64h	76h												
	Análisis Previo del Problema	*8 horas*		8h																
	Determinar elementos básicos	*12 horas*		4h	8h															
	Clasificación y Estudio de restri	*32 horas*			32h															
	Análisis de requisitos	*0 horas*			0h															
	Elaborar Modelo conceptual U	*24 horas*					24h													
	Definición de Casos de uso abs	*56 horas*					40h	16h												
	Asignación de responsabilidad	*4 horas*						4h												
	Análisis del comportamiento d	*56 horas*						56h												
	Casos de uso	*0 horas*						0h												
2	Arquitecto	116 horas						8h	80h	8h				20h						
	Diseñar Diagrama de clases de	*24 horas*						8h	16h											
	Diseñar screens Capa de prese	*16 horas*							16h											
	Concretar Casos de Uso	*24 horas*							24h											
	Asignación de responsabilidad	*8 horas*							8h											
	Elaborar diagramas de secuen	*24 horas*							16h	8h										
	Migrar a Domain Model	*20 horas*												20h						
	Diseño	*0 horas*												0h						
3	Becario	72 horas							16h							42h	14h			
	Digitalización de las cartas y pl	*16 horas*							16h											
	Recapitulación de documentos	*16 horas*														16h				
	Actualización de la memoria d	*40 horas*														26h	14h			
4	Diseñador Gráfico	76 horas							32h	24h	7h	13h								
	Creación de la interfaz de usua	*56 horas*							32h	24h										
	Rediseñar Interficie Gráfica	*20 horas*									7h	13h								
5	Tester	150 horas												30h	80h	40h				
	Testeo del sistema jugador vs j	*50 horas*												27h	21h	2h				
	Testeo de la inteligencia arficic	*100 horas*												3h	59h	38h				
	Testing	*0 horas*														0h				
6	Programador	216 horas								56h	32h	67h	20h	41h						
	Elaboración del sistema de ayu	*16 horas*								16h										
	Serialización de las clases de d	*16 horas*								16h										
	Implementación de las clases b	*56 horas*								24h	32h									
	Adaptar controlador a la nuev	*12 horas*										12h								
	Adaptar a Domain Model	*16 horas*												16h						
	Inclusión del sistema de Jugad	*36 horas*											11h	25h						
	Implementación de la inteligen	*64 horas*										55h	9h							

A partir de esta información, podremos calcular fácilmente el coste del proyecto para calcular el coste del proyecto

Recuros	**Trabajo**	**Coste por hora**	**Total Euros**
Analista	192 horas	27,00 €	5.184,00 €
Arquitecto	116 horas	30,00 €	3.480,00 €
Becario	72 horas	15,00 €	1.080,00 €
Diseñador GUI	76 horas	34,00 €	2.584,00 €
Tester	150 horas	20,00 €	3.000,00 €
Programador	216 horas	23,00 €	4.938,00 €
Total	**822 horas**		**20.296,00 €**

5. Situación actual

En el siguiente diagrama de gantt actualizado, en el que se muestran las tareas a lo largo del tiempo, dependencias entre ellas y los recursos asignados a lo largo del tiempo

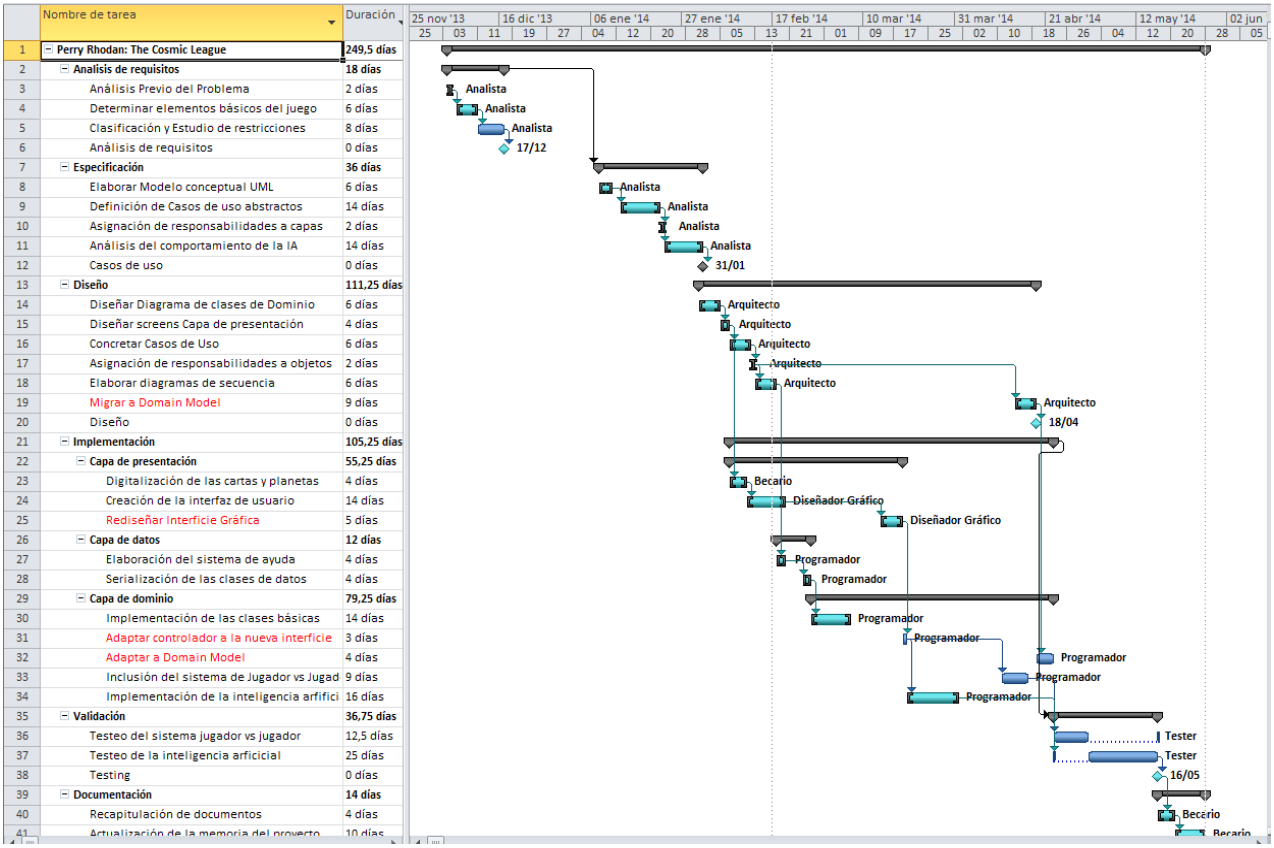


Diagrama de Gantt 12/06/14

Como podemos ver, se ha producido un retraso de aproximadamente un mes (20%) respecto a la planificación inicial.

La desviación respecto al presupuesto no ha sido demasiado grave, pero aún así, se podría haber evitado de haber tenido en cuenta las eventualidades de Java.Swing durante la etapa de Diseño. No obstante, también es cierto que el desarrollo en cascada es propenso a producir este tipo de desviaciones (e incluso mayores[20]).

20 Fuente: *Magne Jorgensen, Kjetil Molokken-Ostvold – How large are software cost overruns? A review of the 1994 Chaos Report*

6. Mejoras

El proyecto está acabado y en un estado satisfactorio. Sin embargo, a continuación se exponen una serie de mejoras propensas a ser implementadas en futuras versiones del proyecto:

6.1. Inteligencia artificial

- **Aumentar la inteligencia:** Actualmente la inteligencia artificial funciona por algoritmos voraces. Como hemos visto en el estudio de la IA, estos algoritmos funcionan bastante bien, pero son algo predecibles y fáciles de anular. Una posible mejora consistiría en hacer que la IA pensara almenos 2 o tres pasos por delante. Esto elevaría al cuadrado la complejidad del heurístico, pero representaría una mejora considerable en la "inteligencia" de la IA
- **Mas tipos de IA:** Una mejora fácil y rápida sería la de añadir más tipos de heurístico, y por consiguiente más tipos de IA. Es más, de la misma manera que hemos añadido una componente aleatoria a los tipos safe y greedy, podríamos desarrollar infinidad de IAs mixtas añadiendo componentes greedy o safe y calibrando el peso de estos en el heurístico.

6.2. Jugador vs Jugador

- **Matchmaking:** Actualmente, para jugar contra otro usuario, el jugador host hace de servidor, y el oponente tiene que introducir la dirección IP del host para establecer una conexión. Una posible mejora sería establecer un servidor de matchmaking al que los jugadores se tengan que conectar para jugar online. Este servidor emparejaría peticiones de jugadores y crearía una partida remota para ellos sin tener que pasar por el engorro de la IP.
- **Chat:** Una mejora rápida y sencilla que consistiría en establecer un pequeño sistema de comunicación por chat en el que los usuarios puedan intercambiar mensajes diversos mientras están jugando
- **Jugador (vs Jugador)ⁿ:** De igual manera, que en el caso del matchmaking, podríamos establecer partidas de múltiples usuarios que competirían entre sí. Al hacer esto, la estrategia del juego aumenta considerablemente porque los varios jugadores pueden aliarse para ayudarse entre sí o bloquear a un jugador en concreto.

Anexo

Glosario

Usuario:

Representación de la persona que utiliza el sistema. Puede ser jugador u oponente dependiendo de si és su turno o no

Jugador:

Usuario que tiene el turno actualmente. Tiene derecho a una acción de vuelo, 2 acciones planetarias y 2 intervenciones. Su turno terminará cuando las haya agotado o cuando él así lo decida

Oponente:

Usuario que no tiene el turno. Tiene derecho a cancelar las intervenciones del jugador si cuenta con las cartas necesarias.

Juego:

Conjunto de elementos que representan el estado de una partida: megagalax acumulados por les jugadores y cantidad de recursos en cada planeta.

Condiciones de Victoria:

Se alcanzará la victoria cuando uno de los 2 jugadores haya alcanzado la quota de megagalax. De agotarse los recursos antes de producirse el caso anterior el vencedor será el que haya acumulado un mayor número de megagalax.

Megagalax:

Representación de los puntos del juego. El juego terminará cuando uno de los 2 jugadores haya acumulado un número determinado de éstos (70 por defecto)

Regursos:

Principal fuente de megagalax del juego. Al ser intercambiados en el planeta que muestran en el dibujo de la carta, se obtienen tantos megagalax como indica el numero en ésta.

Planetas:

Punto de interaccion de los jugadores. En ellos se pueden realizar acciones planetarias. De menor a mayor distancia respecto al Sol:

1. Coronis (Y:Amarillo)
2. Naginther (G:Verde)

3. Vinau II (R:Rojo)
4. Taghelia (P:Lila)
5. Talmazzar (B:Marron)
6. Dahkit (W:blanco)

Acción planetaria:

Puede ser cargar un contenedor, descargar un contenedor o comprar tecnología. El jugador dispone de 2 de ellas

Cargar un contenedor:

Acción de traspasar una pila de recursos de un planeta a uno de los contenedores libres del jugador

Descargar un contenedor:

Acción de traspasar una pila de recursos de uno de los contenedores del jugador al planeta solicitante de ese recurso

Comprar una tecnología:

Acción de activar una tecnología. Cuesta tantos megagalax como tecnologías de las que disponia el jugador al principio del presente turno

Vuelo:

Acción de desplazarse por el tablero. Puede realizarse una vez por turno y termina cuando se inicia una acción planetaria. El coste de éste varía segun su dirección

Intervención:

Acción que altera el flujo normal del juego. El jugador puede realizar 2 por turno

Cancelar Intervención:

El oponente puede anular todas (excepto Passenger) las intervenciones de su oponente a costa de una carta del mismo tipo.

Tecnología:

Aumentos permanentes que proporcionan bonificaciones diversas al jugador. Cuesta una acción planetaria activarlas

Contenedor:

Puede contener una pila de recursos. Ha de vaciarse antes de volver a ser usado

Pila de recursos:

Numero indeterminado de cartas de recursos de un mismo tipo (planeta de destino) procedentes de un mismo planeta de origen

Inventario:

Conjunto de bienes que representan el estado de un usuario:

- Megagalax
- Tecnologías
- Contenedores y contenido de estos

Mano:

Conjunto de hasta 5 cartas de intervención y tecnología que pueden ser usaras por el jugador.

Mazo de robo:

Conjunto de cartas de intervención y tecnología de cada jugador, pendientes de ser usadas. Al final de cada turno el jugador robará tantas cartas como le permitan sus tecnolgías

Pila de descarte:

Conjunto de cartas de intervención y tecnología usadas o descartadas por el jugador. Una vez todas las cartas del mazo han sido descartadas, se continuará la partida normalmente

Base de Datos

Cartas de recurso

ID	Anverso (Valor 3)	Reverso (Valor 2)
1	Y	G
2	Y	R
3	Y	P
4	Y	B
5	Y	W
6	G	Y
7	G	R
8	G	P
9	G	B
10	G	W
11	R	Y
12	R	G
13	R	P
14	R	B
15	R	W

ID	Anverso (Valor 3)	Reverso (Valor 2)
16	P	Y
17	P	G
18	P	R
19	P	B
20	P	W
21	B	Y
22	B	G
23	B	R
24	B	P
25	B	W
26	W	Y
27	W	G
28	W	R
29	W	P
30	W	B

Cartas Jugador

Tecnologías

ID	Nombre	Descripción	Ampliación ayuda
1	Carta Inicial	Contenedor Repostaje	Cada Carta Inicial dispone de dos tecnologías "Contenedor" y "Repostaje". Esto permite a los jugadores transportar cartas de recursos des del principio del juego, así como robar una carta de su mazo al final de su turno.
2	Contenedor	Se usa para transportar recursos.n Un Contenedor no puede reutilizarse hasta ser descargado	Cada "Contenedor" permite transportar cualquier número de cartas de recurso, siempre y cuando tengan el mismo planeta de destino. Para poder cargar un "Contenedor" nuevamente, éste debe estar vacío. Si un jugador dispone de mas de un "Contenedor", éste será capaz de transportar más cartas de recursos a la vez - concretamente, cartas de recursos con diferentes destinos, cada una de ellas en su "Contenedor". Nótese que cargar o descargar un "Contenedor" requiere una acción planetaria completa.
3	Repostaje	Roba una carta de tu mazo al final de tu turno (mano máxima de 5)	Cada "Repostaje" permite al jugador robar una carta adicional al final de su turno, siempre que el número de cartas en su mano sea menor o igual a 5 En ningún momento un jugador podrá tener mas de 5 cartas en su mano.
4	Motor de fusión	Suma 1 a las unidades de energía disponibles por vuelo.	Cada "Motor de fusión" incrementa el numero de unidades de energía por vuelo en 1.
5	Hipersalto al interior	Sólo puede ser jugada al principio de tu turno. Mueve tu nave a la órbita del planeta más cercano al Sol. Esta acción no cuenta como vuelo.	Esta tecnología sólo podrá ser usada al principio del turno del jugador. El jugador moverá su nave a la orbita del planeta mas cercano al Sol. No se considera un vuelo, por lo que el jugador podrá lanzar el dado como haría normalmente para realizar su acción de vuelo.
6	Hipersalto al exterior	Sólo puede ser jugada al principio de tu turno. Mueve tu nave a la órbita del planeta más alejado del Sol. Esta acción no cuenta como vuelo.	Esta tecnología sólo podrá ser usada al principio del turno del jugador. El jugador moverá su nave a la orbita del planeta mas alejado del Sol. No se considera un vuelo, por lo que el jugador podrá lanzar el dado como haría normalmente para realizar su acción de vuelo.

7	Estación Orbital	Te permite realizar una acción planetaria desde la órbita del planeta. (No podrá utilizarse para descargar pasajeros)	Cada "Estación Orbital" que posea el jugador le permite ejecutar una de las 2 acciones planetarias de su turno desde la órbita del planeta, en lugar de desde el propio planeta. Nota: Debido a que "Pasajero" es una carta de intervención, no se permite utilizar una "Estación Orbital" para llevar a un pasajero a su destino.

Intervenciones

ID	Nombre	Descripción	Ampliación ayuda
8	Pasajero Gataser	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.
9	Pasajero Ara	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.
10	Pasajero Kartanin	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.
11	Pasajero Terraner	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.

12	Pasajero Springer	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.
13	Pasajero Topsider	Obtienes 3 Megagalax por descargar al pasajero mientras te encuentras en el planeta mostrado. No puede ser neutralizada.	El jugador sólo puede jugar esta carta si su nave se encuentra en el planeta mostrado como destino del pasajero de la carta. Gana 3 Megagalax por jugar esta intervención. Un intervención de tipo "Pasajero" no puede ser neutralizada.
14	Agente Infiltrado	Te permite realizar una acción planetaria adicional.	El jugador puede realizar una acción planetaria tras jugar esta carta (Cargar un "Contenedor", Descargar un "Contenedor", o Comprar una Tecnología).
15	Activar Postcombustión	Te permite realizar una acción de vuelo adicional.	El jugador puede realizar una acción de vuelo adicional - adicional a la acción de vuelo de su turno. No obstante, sólo le permite lanzar un dado y mover su nave dependiendo del resultado de éste - esta interveción no permite la ejecución de nuevas acciones planetarias o intervenciones (Básicamente, no le proporciona otro turno).
16	Túnel Solar	Te permite volar desde la órbita del planeta mas cercano al Sol hasta la órbita del más lejano, gastando 1 unidad de energía	Durante su turno, el jugador puede volar desde la órbita del planeta más cercano al Sol, a la orbita del más alejado usando una única unidad de energía. Esto no se aplica en sentido contrario. Se permite lanzar un dado a priori para determinar el número de unidades de energía disponibles y mover su nave a la órbita del planeta más cercano (al Sol). Si al jugador le quedan unidades de energía tras atravesar el Sol, puede usarlas para continuar el vuelo.
17	Carga Interior	Carga un contenedor vacío con recursos de uno de los dos planetas más cercanos al Sol. Esto no se considera como acción planetaria y puede realizarse desde cualquier lugar.	El jugador carga uno de sus "Contenedor"es vacíos con recursos de uno de los dos planetas más interiores. Esto se ejecuta igual que si fuera la acción planetaria de "Cargar un Contenedor" pero no se descuenta de las 2 acciones planetarias por turno. No importa donde se encuentre la nave del jugador actualmente. No obstante, si el jugador se encuentra en el planeta de destino de los recursos obtenidos, y dispone de acciones planetarias, se

			le permite intercambiar inmediatamente estos recursos por Megagalax.
18	Carga Exterior	Carga un contenedor vacío con recursos de uno de los dos planetas más alejados del Sol. Esto no se considera como acción planetaria y puede realizarse desde cualquier lugar.	El jugador carga uno de sus "Contenedor"es vacíos con recursos de uno de los dos planetas más exteriores. Esto se ejecuta igual que si fuera la acción planetaria de "Cargar un Contenedor" pero no se descuenta de las 2 acciones planetarias por turno. No importa donde se encuentre la nave del jugador actualmente. No obstante, si el jugador se encuentra en el planeta de destino de los recursos obtenidos, y dispone de acciones planetarias, se le permite intercambiar inmediatamente estos recursos por Megagalax.
19	Activar Intercambiador de a Bordo	Intercambia el contenido de uno de tus contenedores por el de uno de los contenedores del oponente.Ambos contenedores deben tener al menos una carta de recurso	El jugador elije uno de los "Contenedor"es de su oponente e intercambia su contenido por el de uno de sus propios "Contenedor"es. Ambos "Contenedor"es deben contener al menos una carta de recurso.
20	Activar Puerta de Intercambio	Cambia la posición de tu nave por la de tu oponente, Esto no se considera como vuelo.	El jugador intercambia la posición de su nave por la de la nave de su oponente. No importa dónde se encuentran las naves actualmente.
21	Descarga de Hipertormenta	Descarta tu mano. Combina los mazos de robo y descarte y barajas, entonces roba 5 nuevas cartas. Esta acción acaba tu turno.	El jugador descarta todas las cartas de su mano, y junta y baraja la pila de descarte y robo para crear una nuevo mazo de robo. Entonces roba 5 cartas. Su turno finaliza tras realizar esta acción.