IA para el Desarrollo de Videojuegos

Proyecto final:
Real Time Wargame.

Autores:
Antonio López Martínez-Carrasco
José María Sánchez Salas

Profesores:
Francisco Javier Marín-Blazquez Gómez
Luis Daniel Hernández Molinero

Introducción

Introducción

- Software y Hardware utilizado.

Introducción

- Software y Hardware utilizado.
  - Librería LibGDX (implementada en Java).

Introducción

- Software y Hardware utilizado.
  - Librería LibGDX (implementada en Java).
  - IDE Eclipse.

Introducción

- Software y Hardware utilizado.
  - Librería LibGDX (implementada en Java).
  - IDE Eclipse.
  - Ningún hardware específico.

Modelo

Modelo

- Clase abstracta WorldObject.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.
- Formation:

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.
- Formation:
  - Hereda de Character.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.
- Formation:
  - Hereda de Character.
  - CircularFormation.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.
- Formation:
  - Hereda de Character.
  - CircularFormation.
  - LineFormation.

Modelo

- Clase abstracta WorldObject.
  - Hereda de Sprite (librería LibGDX).
- Obstacle.
  - Hereda de WorldObject.
  - Obstáculos **estáticos**.
- Character.
  - Hereda de WorldObject.
  - Personajes del mundo.
- Formation:
  - Hereda de Character.
  - CircularFormation.
  - LineFormation.
  - StarFormation Video

Modelo – Formaciones

- Patrón de diseño Composite.

Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.

Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.
- Cierta funcionalidad se delega a las formaciones concretas:

Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.
- Cierta funcionalidad se delega a las formaciones concretas:
  - Calcular la posición de los integrantes.

Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.
- Cierta funcionalidad se delega a las formaciones concretas:
  - Calcular la posición de los integrantes.
  - Calcular el Steering a aplicar (árbitro y lista de comportamientos “a pelo”).

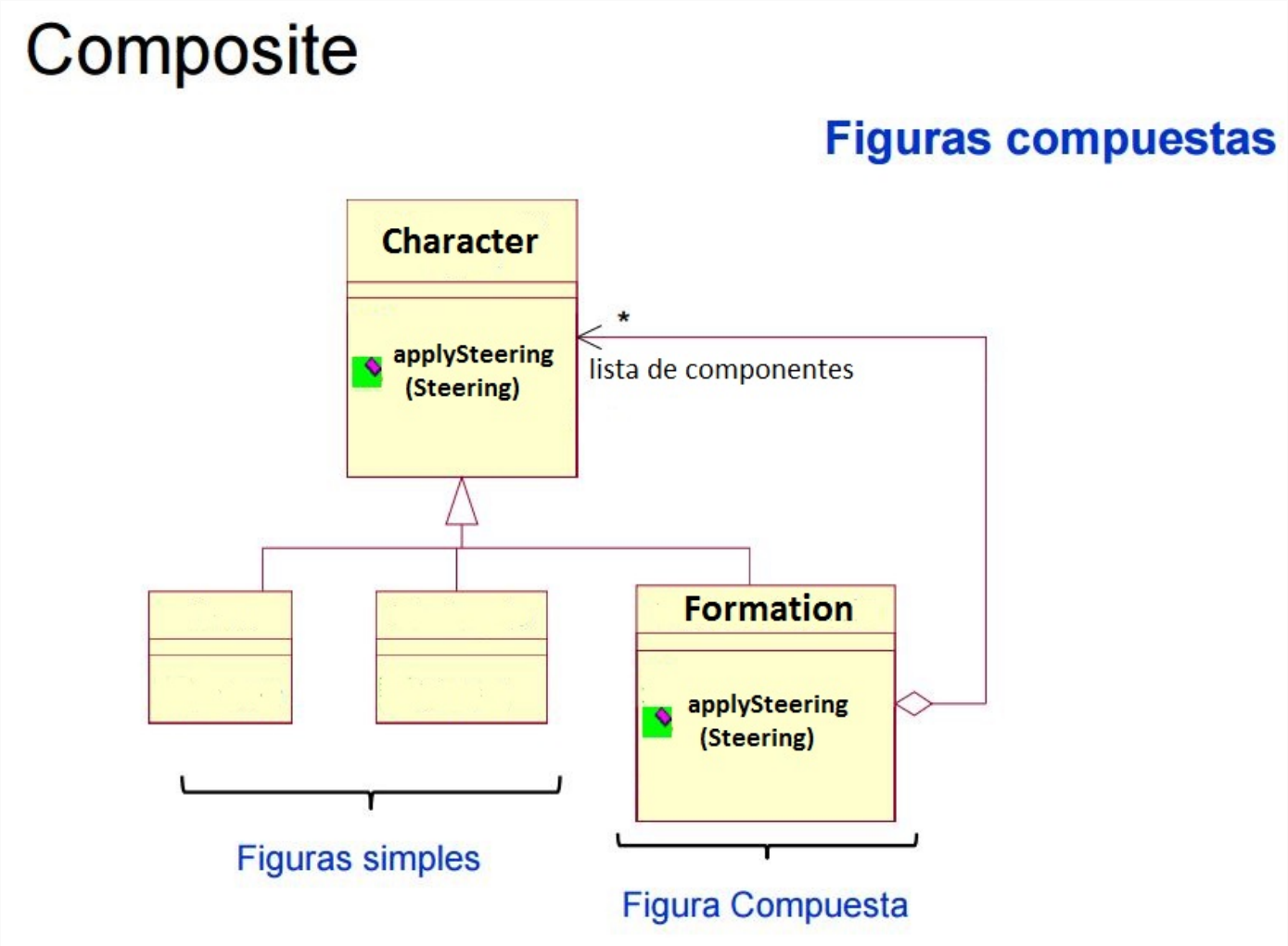
Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.
- Cierta funcionalidad se delega a las formaciones concretas:
  - Calcular la posición de los integrantes.
  - Calcular el Steering a aplicar (árbitro y lista de comportamientos "a pelo").
- La lista de comportamientos de los integrantes no se tiene en cuenta.

Modelo – Formaciones

- Patrón de diseño Composite.
- Patrón de diseño Método Plantilla.
- Cierta funcionalidad se delega a las formaciones concretas:
  - Calcular la posición de los integrantes.
  - Calcular el Steering a aplicar (árbitro y lista de comportamientos "a pelo").
- La lista de comportamientos de los integrantes no se tiene en cuenta.
- El movimientos de los integrantes a sus respectivas posiciones se hace con una lista de comportamientos "a pelo" (utilizando un árbitro por prioridad, se explica más adelante).

Modelo – Formaciones



Modelo – Ejemplo formación compleja

- Video

Steerings

Steerings

- Interfaz Steering.

Steerings

- Interfaz Steering.
  - Para tratar a todos los tipos de Steerings de manera homogénea.

Steerings

- Interfaz Steering.
  - Para tratar a todos los tipos de Steerings de manera homogénea.
- Steerings no acelerados.

Steerings

- Interfaz Steering.
  - Para tratar a todos los tipos de Steerings de manera homogénea.
- Steerings no acelerados.
  - Velocidad lineal y velocidad angular.

Steerings

- Interfaz Steering.
  - Para tratar a todos los tipos de Steerings de manera homogénea.
- Steerings no acelerados.
  - Velocidad lineal y velocidad angular.
- Steerings acelerados.

Steerings

- Interfaz Steering.
  - Para tratar a todos los tipos de Steerings de manera homogénea.
- Steerings no acelerados.
  - Velocidad lineal y velocidad angular.
- Steerings acelerados.
  - Aceleración lineal y aceleración angular.

Comportamientos

Comportamientos

- Interfaz Behaviour.

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align
  - Arrive con un radio Video

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align
  - Arrive con un radio Video
  - Arrive

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align
  - Arrive con un radio Video
  - Arrive
  - Flee

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align
  - Arrive con un radio Video
  - Arrive
  - Flee
  - Seek

Comportamientos

- Interfaz Behaviour.
  - Para tratar a todos los tipos de comportamientos de manera homogénea.
  - Método `getSteering()`.
- Behaviours no acelerados.
  - Seek
  - Flee
  - Arrive
  - Wander
- Behaviours acelerados.
  - Align
  - Anti-Align
  - Arrive con un radio Video
  - Arrive
  - Flee
  - Seek
  - Velocity Matching

Comportamientos

- Behaviours delegados.

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video
  - WallAvoidance

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video
  - WallAvoidance
  - Wander

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video
  - WallAvoidance
  - Wander
- Behaviours en grupo.

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video
  - WallAvoidance
  - Wander
- Behaviours en grupo.
  - Cohesion Video

Comportamientos

- Behaviours delegados.
  - CollisionAvoidance
  - Evade
  - Face
  - Looking Where You Going
  - PathFollowing con Arrive
  - PathFollowing con Seek Video
  - Persue Video
  - WallAvoidance
  - Wander
- Behaviours en grupo.
  - Cohesion Video
  - Separation Video

Árbitros

Árbitros

- Interfaz Arbitrator.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.
  - Ejecuta los comportamientos siguiendo su orden.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.
  - Ejecuta los comportamientos siguiendo su orden.
  - Devuelve el Steering de un comportamiento si supera cierto valor `epsilon`.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.
  - Ejecuta los comportamientos siguiendo su orden.
  - Devuelve el Steering de un comportamiento si supera cierto valor `epsilon`.
  - Si llega al final, devuelve el último de la lista.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.
  - Ejecuta los comportamientos siguiendo su orden.
  - Devuelve el Steering de un comportamiento si supera cierto valor `epsilon`.
  - Si llega al final, devuelve el último de la lista.
- Árbitro por mezcla. Comportamientos acelerados.

Árbitros

- Interfaz Arbitrator.
  - Para tratar a todos los tipos de árbitros de manera homogénea.
  - Método `getSteering(behaviours)`.
- Lista de comportamientos de los personajes.
  - Mapa con clave Float y valor Behaviour.
  - Ordenados de mayor a menor $\rightarrow$ Más importante primero.
- Árbitro por prioridad.
  - Para comportamientos acelerados y no acelerados.
  - Ejecuta los comportamientos siguiendo su orden.
  - Devuelve el Steering de un comportamiento si supera cierto valor `epsilon`.
  - Si llega al final, devuelve el último de la lista.
- Árbitro por mezcla. Comportamientos acelerados.
- Árbitro por mezcla. Comportamientos no acelerados.

Interacción parte Reactiva

Interacción parte Reactiva

- Método `applyBehaviour()`.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.
  - Se llama al método `applySteering(steering)` con el Steering devuelto.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.
  - Se llama al método `applySteering(steering)` con el Steering devuelto.
- Método `applySteering(steering)`.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.
  - Se llama al método `applySteering(steering)` con el Steering devuelto.
- Método `applySteering(steering)`.
  - Si el personaje no pertenece a una formación, se invoca al método `update(steering, time)`.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.
  - Se llama al método `applySteering(steering)` con el Steering devuelto.
- Método `applySteering(steering)`.
  - Si el personaje no pertenece a una formación, se invoca al método `update(steering, time)`.
- Método `update(steering, time)`.

Interacción parte Reactiva

- Método `applyBehaviour()`.
  - Si el personaje no pertenece a una formación, se invoca a su árbitro.
  - Se llama al método `applySteering(steering)` con el Steering devuelto.
- Método `applySteering(steering)`.
  - Si el personaje no pertenece a una formación, se invoca al método `update(steering, time)`.
- Método `update(steering, time)`.
  - Se actualizan los parámetros del personaje en función del Steering.

PathFinding

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.
  - Devuelve la **lista completa** de puntos desde el origen al destino.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.
  - Devuelve la **lista completa** de puntos desde el origen al destino.
  - Solo es necesario llamarlo una vez.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.
  - Devuelve la **lista completa** de puntos desde el origen al destino.
  - Solo es necesario llamarlo una vez.
- PathFinding Punto-A-Punto.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.
  - Devuelve la **lista completa** de puntos desde el origen al destino.
  - Solo es necesario llamarlo una vez.
- PathFinding Punto-A-Punto.
  - Va devolviendo **punto por punto**.

PathFinding

- PathFinding con algoritmo LRTA* y espacio de búsqueda minimal.
- Distancias Ecluidea, Chebyshev y Manhattan.
- PathFinding Continuo.
  - Devuelve la **lista completa** de puntos desde el origen al destino.
  - Solo es necesario llamarlo una vez.
- PathFinding Punto-A-Punto.
  - Va devolviendo **punto por punto**.
  - Se debe llamar continuamente para obtener al punto al que ir.

Modificaciones del modelo para la parte Táctica

Clase Character

- Nuevos atributos.

Clase Character

- Nuevos atributos.
  - Rol táctico.

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).
  - Equipo.

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).
  - Equipo.
- Nuevos métodos.

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).
  - Equipo.
- Nuevos métodos.
  - `initializeTacticalRole(role)`.

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).
  - Equipo.
- Nuevos métodos.
  - `initializeTacticalRole(role)`.
  - `updateTacticalRole()`.

Clase Character

- Nuevos atributos.
  - Rol táctico.
  - Vida (máxima y actual).
  - Equipo.
- Nuevos métodos.
  - `initializeTacticalRole(role)`.
  - `updateTacticalRole()`.
  - `getVelocityFactorOfThisCharacter()` (se explica más adelante).

Enumerado Team

- Equipos del juego.

Enumerado Team

- Equipos del juego.
  - `FJAVIER`.

Enumerado Team

- Equipos del juego.
  - `FJAVIER`.
  - `LDANIEL`.

Enumerado Team

- Equipos del juego.
  - `FJAVIER`.
  - `LDANIEL`.
  - `NEUTRAL`.

Enumerado Team

- Equipos del juego.
  - `FJAVIER`.
  - `LDANIEL`.
  - `NEUTRAL`.
- Método `getEnemyTeam()`.

Roles Tácticos

Roles Tácticos

- Interfaz TacticalRole.

Roles Tácticos

- Interfaz TacticalRole.
- Clases abstractas Archer y Soldier.

Roles Tácticos

- Interfaz TacticalRole.
- Clases abstractas Archer y Soldier.
- Roles defensivos.

Roles Tácticos

- Interfaz TacticalRole.
- Clases abstractas Archer y Soldier.
- Roles defensivos.
  - Soldado y arquero.

Roles Tácticos

- Interfaz TacticalRole.
- Clases abstractas Archer y Soldier.
- Roles defensivos.
  - Soldado y arquero.
- Roles ofensivos.

Roles Tácticos

- Interfaz TacticalRole.
- Clases abstractas Archer y Soldier.
- Roles defensivos.
  - Soldado y arquero.
- Roles ofensivos.
  - Soldado y arquero.

Interfaz TacticalRole

- Métodos.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.
  - `initialize(character)`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.
  - `initialize(character)`.
  - `update(character)`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.
  - `initialize(character)`.
  - `update(character)`.
  - Estos métodos son implementados en las clases abstractas Archer y Soldier.

Interfaz TacticalRole

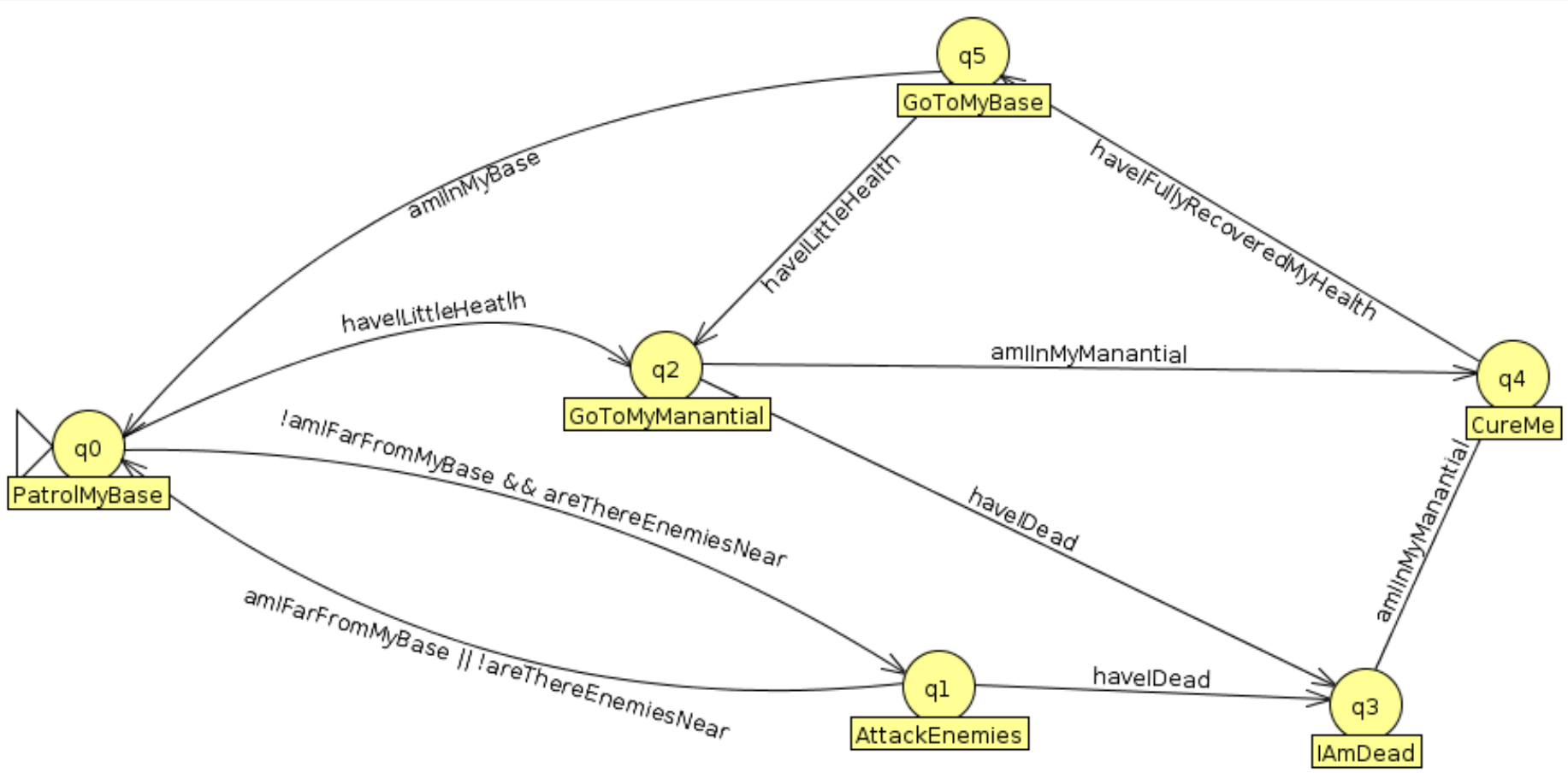
- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.
  - `initialize(character)`.
  - `update(character)`.
  - Estos métodos son implementados en las clases abstractas Archer y Soldier.
- Atributo `health_cure`.

Interfaz TacticalRole

- Métodos.
  - `getVelocityFactor(ground)` (es usado en la clase Character).
  - `getTacticalCost(ground)`.
  - `getMaxDistanceOfAttack()`.
  - `getDamageToDone()`.
  - `getMaxSpeed()`.
  - `initialize(character)`.
  - `update(character)`.
  - Estos métodos son implementados en las clases abstractas Archer y Soldier.
- Atributo `health_cure`.
  - Todos se curan al mismo ritmo, independientemente de su rol.

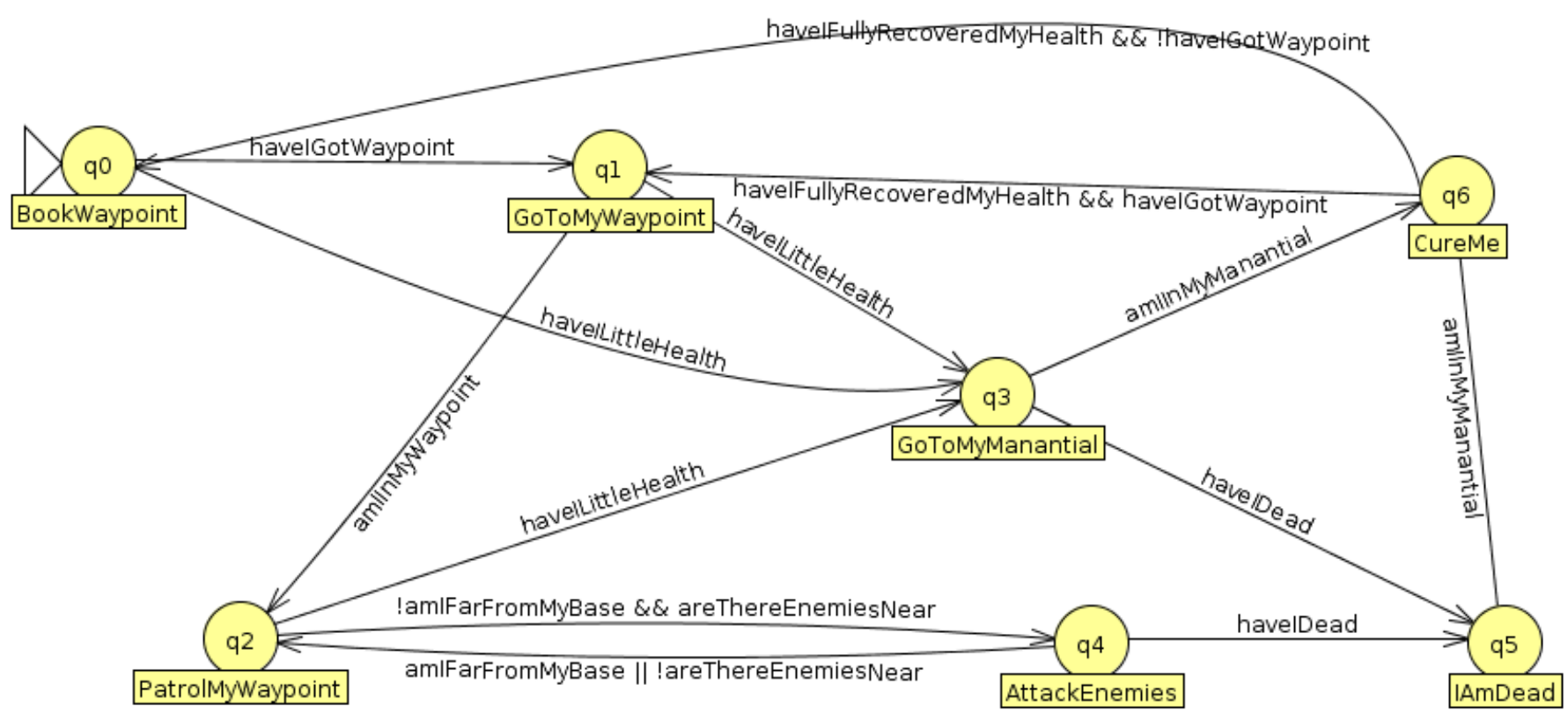
Roles Defensivos

- Soldado.



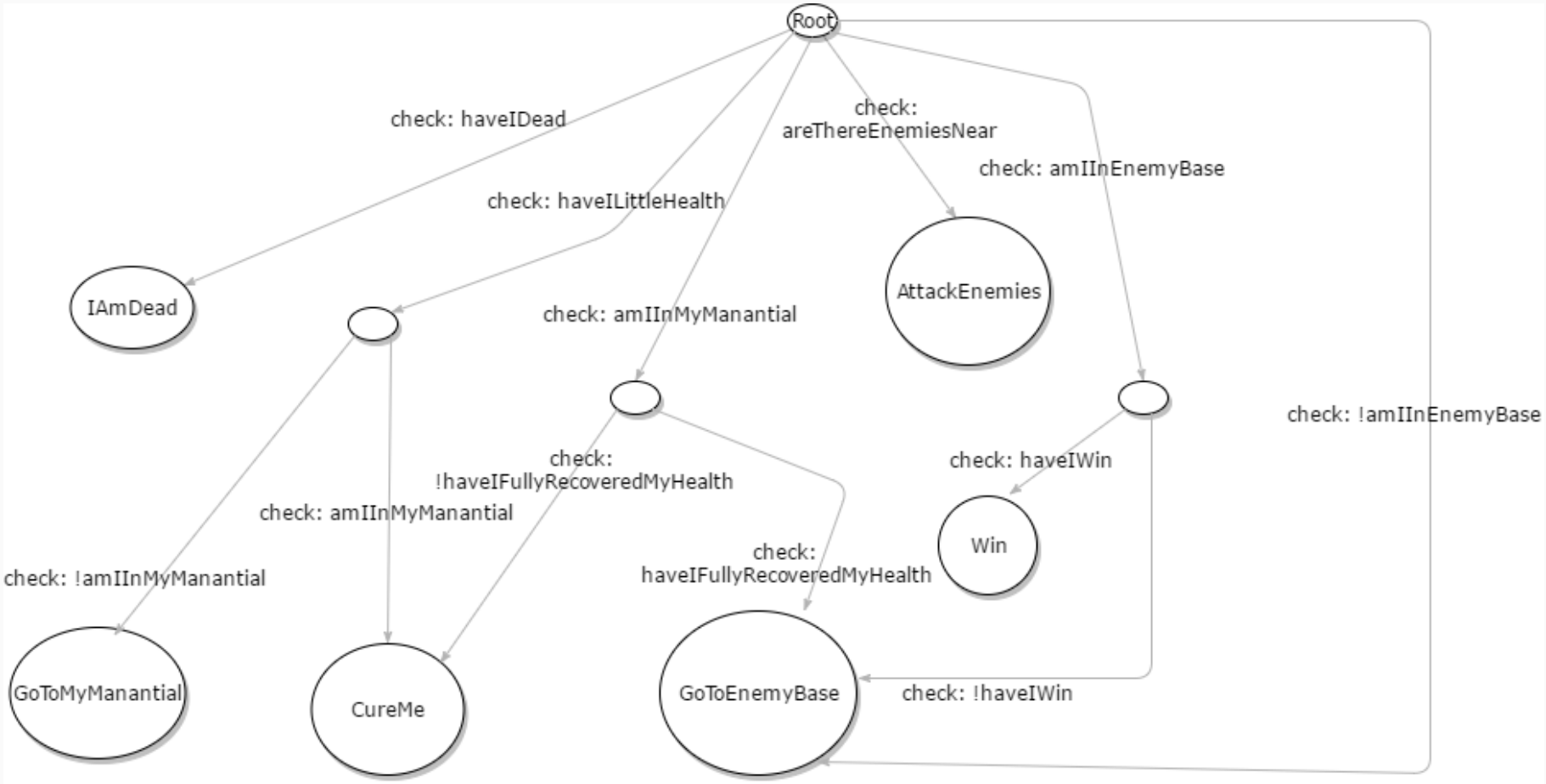
Roles Defensivos

- Arquero.



Roles Ofensivos

- Soldado y arquero.



Otros comportamientos

Otros comportamientos

- Ataque.

Otros comportamientos

- Ataque.
- Cura.

Otros comportamientos

- Ataque.
- Cura.
- Se han implementado como Behaviours para tratarlos de manera homogénea.

Otros comportamientos

- Ataque.
- Cura.
- Se han implementado como Behaviours para tratarlos de manera homogénea.
- Ataque y Cura en las formaciones.

Otros comportamientos

- Ataque.
- Cura.
- Se han implementado como Behaviours para tratarlos de manera homogénea.
- Ataque y Cura en las formaciones.
  - El ancla es el que da las órdenes de ataque y cura a los integrantes.

Acciones y comprobaciones

Acciones y comprobaciones

- Actions.

Acciones y comprobaciones

- Actions.
- Checks.

Acciones y comprobaciones

- Actions.
- Checks.
- Útiles para la abstracción de la parte reactiva.

Acciones y comprobaciones

- Actions.
- Checks.
- Útiles para la abstracción de la parte reactiva.
- Sirven de puente entre la parte reactiva y la parte táctica.

Waypoints

Waypoints

- Waypoints de las bases.

Waypoints

- Waypoints de las bases.
- Waypoints de los puentes.

Waypoints

- Waypoints de las bases.
- Waypoints de los puentes.
  - Cada equipo tiene 6 waypoints.

Waypoints

- Waypoints de las bases.
- Waypoints de los puentes.
  - Cada equipo tiene 6 waypoints.
  - En cada lado de los puentes, hay 2 waypoints.

Waypoints

- Waypoints de las bases.
- Waypoints de los puentes.
  - Cada equipo tiene 6 waypoints.
  - En cada lado de los puentes, hay 2 waypoints.
  - Sistema de reserva y liberación de waypoints.

Puntos de Moral

Puntos de Moral

- Cada base tiene una puntuación de moral.

Puntos de Moral

- Cada base tiene una puntuación de moral.
- La moral de una base sube cuando hay personajes de un equipo en su base y no hay personajes del equipo contrario.

Puntos de Moral

- Cada base tiene una puntuación de moral.
- La moral de una base sube cuando hay personajes de un equipo en su base y no hay personajes del equipo contrario.
- La moral de una base baja cuando la base no está protegida y hay personajes del equipo contrario.

Puntos de Moral

- Cada base tiene una puntuación de moral.
- La moral de una base sube cuando hay personajes de un equipo en su base y no hay personajes del equipo contrario.
- La moral de una base baja cuando la base no está protegida y hay personajes del equipo contrario.
- El equipo ganador es el que consigue reducir los puntos de moral de la base contraria a 0.

Mapas de Influencia

Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.

Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.
- Cada equipo tiene su propio mapa/matriz de influencia.

Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.
- Cada equipo tiene su propio mapa/matriz de influencia.
- El equipo con mayor influencia en una casilla, domina dicha casilla.

Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.
- Cada equipo tiene su propio mapa/matriz de influencia.
- El equipo con mayor influencia en una casilla, domina dicha casilla.
- La influencia se va reduciendo con la distancia.

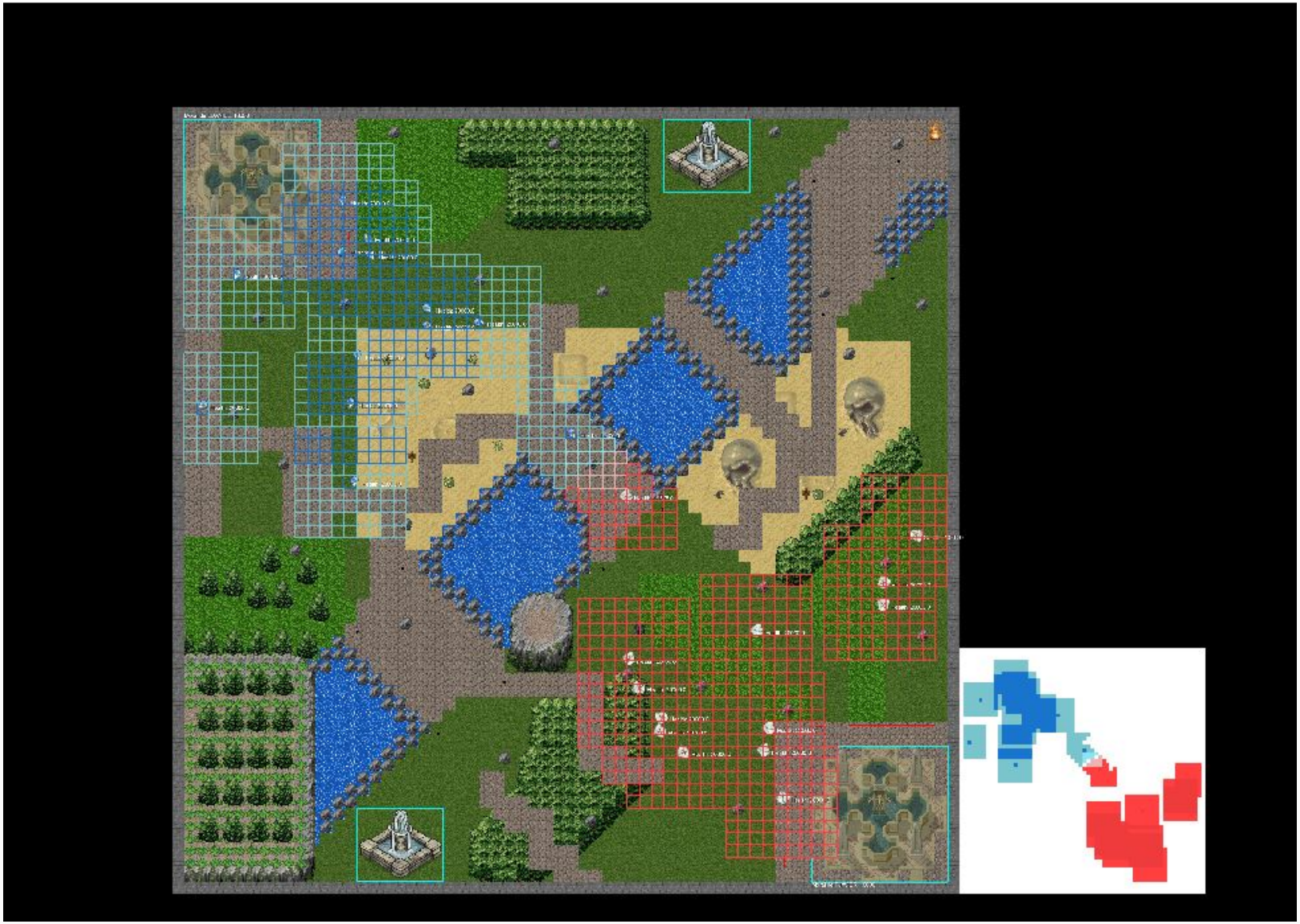
Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.
- Cada equipo tiene su propio mapa/matriz de influencia.
- El equipo con mayor influencia en una casilla, domina dicha casilla.
- La influencia se va reduciendo con la distancia.
- Para el cálculo de la influencia se ha utilizado la distancia de Chebyshev (con algunas modificaciones).

Mapas de Influencia

- El mapa de influencia refleja la presencia/poder de un equipo en una zona.
- Cada equipo tiene su propio mapa/matriz de influencia.
- El equipo con mayor influencia en una casilla, domina dicha casilla.
- La influencia se va reduciendo con la distancia.
- Para el cálculo de la influencia se ha utilizado la distancia de Chebyshev (con algunas modificaciones).
- Al dibujar el mapa se tienen en cuenta ambas matrices de influencia.

Mapas de Influencia



PathFinding Táctico

PathFinding Táctico

- Hace uso de los mapas de influencia.

PathFinding Táctico

- Hace uso de los mapas de influencia.
- También puede hacer uso de la información táctica (`getTactialCost(ground)`) de cada rol.

PathFinding Táctico

- Hace uso de los mapas de influencia.
- También puede hacer uso de la información táctica (`getTactialCost(ground)`) de cada rol.
- Flag para que se pueda ir activando y desactivando sobre la marcha.

PathFinding Táctico

- Hace uso de los mapas de influencia.
- También puede hacer uso de la información táctica (`getTactialCost(ground)`) de cada rol.
- Flag para que se pueda ir activando y desactivando sobre la marcha.
- Video (sin colisiones)

Flocking

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.
  - Cohesion.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.
  - Cohesion.
  - VelocityMatching.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.
  - Cohesion.
  - VelocityMatching.
  - LookingWhereYouGoing.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.
  - Cohesion.
  - VelocityMatching.
  - LookingWhereYouGoing.
  - Wander.

Flocking

- Se utiliza un árbitro por mezcla ponderada con los siguientes comportamientos:
  - CollisionAvoidance.
  - Separation.
  - Cohesion.
  - VelocityMatching.
  - LookingWhereYouGoing.
  - Wander.
- Video

Interacción con el usuario

Interacción con el usuario

- Selección de personajes del mismo equipo.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.
  - En el caso de las formaciones, la liberación implica que la formación se deshaga.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.
  - En el caso de las formaciones, la liberación implica que la formación se deshaga.
- Realizar ciertos comportamientos.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.
  - En el caso de las formaciones, la liberación implica que la formación se deshaga.
- Realizar ciertos comportamientos.
  - Todos los personajes seleccionados realizan el mismo comportamiento.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.
  - En el caso de las formaciones, la liberación implica que la formación se deshaga.
- Realizar ciertos comportamientos.
  - Todos los personajes seleccionados realizan el mismo comportamiento.
- Implementado siguiendo una máquina de estados.

Interacción con el usuario

- Selección de personajes del mismo equipo.
  - Cuando un personaje es seleccionado, se desactiva su rol.
  - En el caso de las formaciones, solamente se ha de seleccionar cualquiera de los integrantes.
- Liberación de los personajes seleccionados.
  - Cuando un personaje es liberado, se activa su rol.
  - En el caso de las formaciones, la liberación implica que la formación se deshaga.
- Realizar ciertos comportamientos.
  - Todos los personajes seleccionados realizan el mismo comportamiento.
- Implementado siguiendo una máquina de estados.
- La terminal muestra la información sobre lo que se puede hacer (y cómo).

Elementos opcionales

Elementos opcionales

- Todos los comportamientos implementados.

Elementos opcionales

- Todos los comportamientos implementados.
- Modo debug (comportamientos, personajes, información táctica...).

Elementos opcionales

- Todos los comportamientos implementados.
- Modo debug (comportamientos, personajes, información táctica...).
- Formación en línea y en estrella.

Elementos opcionales

- Todos los comportamientos implementados.
- Modo debug (comportamientos, personajes, información táctica...).
- Formación en línea y en estrella.
- Formación de formaciones hasta el infinito y más allá.

Elementos opcionales

- Todos los comportamientos implementados.
- Modo debug (comportamientos, personajes, información táctica...).
- Formación en línea y en estrella.
- Formación de formaciones hasta el infinito y más allá.
- PathFinding táctico: mapa de influencia y costes tácticos del terreno.

Demostración del juego completo

Demostración del juego completo

- Ejecución del juego.

Gracias por su atención.

Ruegos y preguntas.