ANÁLISIS Y DISEÑO DEL GRUPO
SISTEMAS MULTIAGENTES

Manuel Gallego Chinchilla
Julián Sánchez Orellana

Índice

1. INTRODUCCIÓN

1.1. Descripción del problema

Estamos realizando una práctica en la que intervienen varios agentes. Como son el agente central, el agente grupo, el agente tablero, y los diversos agentes jugadores que cada uno está especializado en jugar a un solo tipo de juego.
En este análisis y diseño, nos vamos a centrar en aquellos agentes que debemos posteriormente implementar el grupo de prácticas, en este caso dos agentes, el agente grupo de juegos, y el agente tablero.

1.2. Objetivos para la solución del problema

Los objetivos que tenemos es que nuestros agentes se comuniquen con todos los agentes que sea necesario, y esa comunicación se realice sin ningún tipo de error.

2. ANÁLISIS

2.1. Agente Grupo

Este agente será el encargado de recibir del agente central las partidas que tiene que organizar, para ello y una vez reciba una partida, debe crear al agente tablero correspondiente, para que este inicie y lleve la partida.
Posteriormente nuestro agente grupo, informará del resultado de la partida al agente central.

Tareas que realizará

- Decidir si acepta partidas por parte del agente central.
- Responder con la decisión tomada.
- Crear el tablero correspondiente una vez aceptada la partida.
- Informar al agente tablero con el número de partidas a echar por ronda.
- Informar al agente central de cómo va la partida.
- Informar al agente central del resultado de la partida.

2.2. Agente Tablero

Tendremos dos tipos de agentes tablero, uno para las partidas del Conecta 4 y otro para las partidas de barquitos.
Los dos realizarán las mismas funciones pero cada uno solo sabrá interpretar los movimientos del juego al que está preparado, y cada uno de los tablero crea una interfaz distinta. Por ello hemos optado por hacer dos agentes.
Nuestro agente tablero, ya sea de un juego u otro, será creado por parte del agente grupo, una vez acepte la realización de una partida.
Una vez acabadas las partidas correspondientes a un enfrentamiento, obtendrá un resultado, que será enviado al agente grupo.

Tareas que realizará

- Representar gráficamente el tablero de juego.
- Pedir movimiento a los jugadores.
- Actualizar en cada turno el tablero, para visualizar el efecto del movimiento.
- Enviar información acerca de la partida al agente grupo.

2.3. Ontología

La ontología nos ayuda a la hora de facilitarnos la vida con el envío de información entre agentes, y así generalizar el contenido de los mismos.
En nuestro caso debemos hablar de los conceptos que utilizamos en el agente grupo, y en el agente tablero, y viendo que es la fase de análisis vamos a enumerar las características que debe cumplir cada uno de estos agentes, que más tarde en la fase de diseño serán asignados como conceptos implementados en nuestro proyecto.

- **Conceptos referentes al agente Grupo**
  - Debe saber informar del transcurso de la partida.
  - Debe llevar una clasificación del juego.
  - Debe saber interpretar la recepción de partidas para lanzarlas en el agente tablero.
  - Debe conocer los motivos de rechazo de una solicitud de inicio de partida.

- **Conceptos referentes al agente Tablero**
  - Hay que interpretar los movimientos recibidos por el jugador.
  - Debe conocer las características del tablero de juego.
  - Tiene que saber representar la clasificación una vez acabada la partida.
  - Tiene que saber representar el resultado de cierto movimiento.
  - Hay que informar del turno de partida.

Todos estos puntos se convertirán en conceptos, en la fase de diseño.

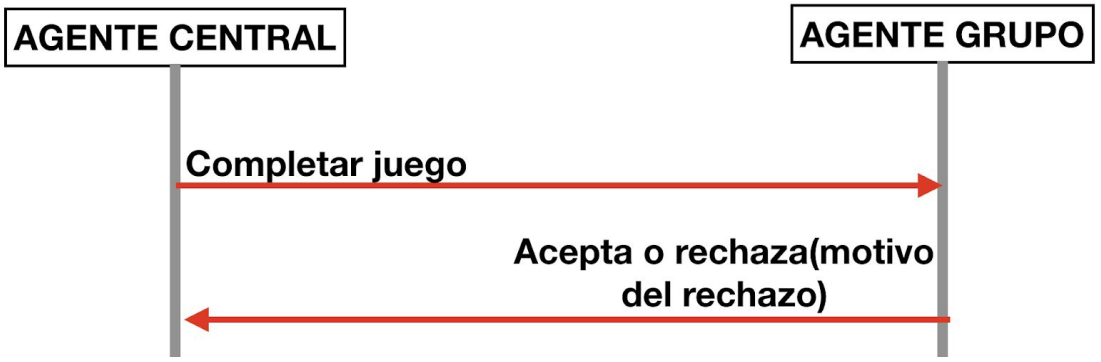
2.4. Comunicación entre agentes

En la parte de diseño entraremos en profundidad para ver qué protocolos utilizaremos para la realización de los mensajes que vamos a explicar ahora.
Vamos a empezar describiendo las comunicaciones que tiene que haber con el **agente grupo:**

2.4.1. Agente GrupoJuegos

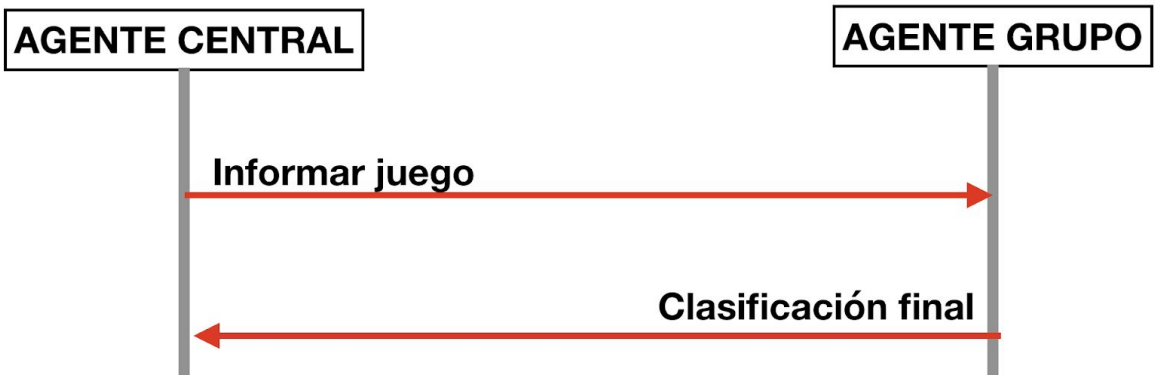
- **Completar Juego**

  El agente central de juego le propone al agente grupo que complete una partida, y nuestro agente debe responder si acepta o no.



- **Informar Juego**

  El agente central necesita conocer el resultado final de la partida que está realizando nuestro agente grupo, por lo que una vez terminada la partida, debemos devolverle la clasificación final.



Ahora vamos a ir con la comunicación que existe con el **agente tablero:**

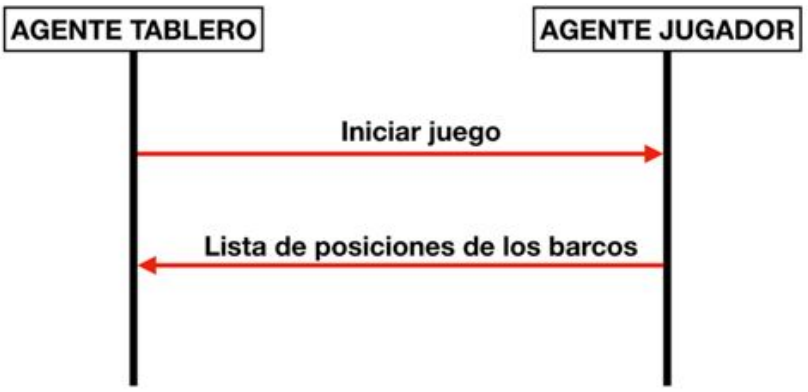
2.4.2. Agente Tablero

- **Jugar una partida:**

  Nuestro agente tablero se encargará de pedirle movimientos a los jugadores implicados en una partida. Recibirá los movimientos y será el encargado de procesarlos y enviarle el resultado del movimiento a los jugadores.

- **Preparación de una Partida (tablero de los barquitos).**

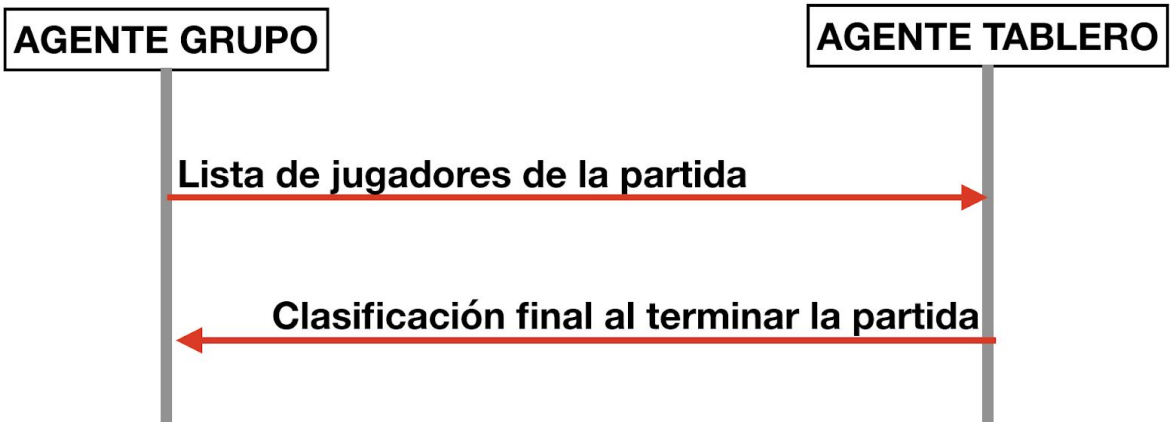
  El agente tablero le solicitará a los jugadores de los barquitos que coloquen los barcos inicialmente en las posiciones que ellos quieran, y una vez el tablero reciba la posición inicial, lo representará en el tablero de forma visual y podrá empezar a pedir turnos.



- **Actualizar Información**

  Por último tenemos la comunicación que existe entre el agente grupo y el agente tablero. En esta comunicación el agente grupo debe enviarle información al tablero de la partida que va a realizar, como al menos la lista de jugadores implicados en ella, para que el agente tablero pueda comunicarse con ellos.

  El otro punto de comunicación entre nuestro agente grupo y tablero se produce una vez acabada la partida, para devolverle la información de la clasificación final, para que más tarde el agente grupo, como ya hemos visto, se la envíe al agente central de juego.



- **Enviar Clasificación:**

  El Agente GrupoJuegos necesita saber la clasificación una vez terminada la partida, por lo que le enviaremos un mensaje simple de tipo **INFORM** donde le adjuntamos la clasificación de la partida

3. DISEÑO

3.1. Agente Grupo

- **Estructura de datos**

  - **ArrayList<Jugador> jugadores:** representará los jugadores que le envía central de juegos para redirigirlos a un grupo de 2 para una partida individual o organizar un torneo con los jugadores que haya. Esta lista vendrá ordenada por puntuación de manera que nos sirva de clasificación.
  - **ArrayList <Integer> tableros:** representará los id de los tableros que hemos creado para identificar qué partida se está jugando.

- **Tareas a realizar**

  - **Decidir si acepta partidas por parte del Agente Central**

```
if(aceptaCentralJuegos){
     enviar(ACCEPT-PROPOSAL(juegoAceptado));
}else{
     enviar(REJECT-PROPOSAL(Motivacion));
}
```

  - **Responder con la decisión tomada**
  - **Crear el tablero correspondiente una vez aceptada la partida**

```
if(partidaAceptada){
     try {
          miAgente.crearAgente("nombreAgente");
          iniciarNuevoAgente();
     } catch (StaleProxyException ex) {
            excepcion();
     }
}
```

- **Informar el Agente Tablero con el número de partidas a echar por ronda**

```
if(partidaAceptada){
     enviar(numRondas);
}
```

- **Informar al Agente Central de cómo va la partida**

```
//Mediante el protocolo Subscribe enviaremos los
mensajes
if(!suscrito){
     enviar(AGREE); //Para suscribirse
}else{
     enviar(INFORM(Clasificación))
}
```

- **Informar al Agente Central del resultado de la partida**

```
//Mediante el protocolo Subscribe enviaremos los
mensajes
if(!suscrito){
     enviar(AGREE); //Para suscribirse
}else{
     enviar(INFORM(Informe))
}
```

3.2. Agente Tablero

- **Estructura de datos**

  - **ArrayList<ArrayList<Casilla>> tablero:** representará las casillas del tablero que en función de las tipo de Casilla que represente, realizará la representación de un juego u otro.Las Casillas serán objetos que almacenarán los movimientos y estados de la partida
  - **boolean ganador:** vendrá inicializada a false y en el momento que haya un ganador se activará para enviar el mensaje Resultado

- **Tareas a realizar**

  - **Representar gráficamente el tablero de juego.**

```
if(partidaActiva){
     tablero.iniciar();
}
```

  - **Pedir movimiento a los jugadores.**

```
if(partidaActiva){
     enviar(CFP(PedirMovimiento)));
}
//Según el mensaje que reciba realizará una acción
if(mensajeRecibido == "REFUSE"){
     descalificar(mensajeRecibido.sender());
}else if(mensajeRecibido == "PROPOSE" &&
          mensajeRecibido.content() == null){
     movimientoRealizado = null;
}else if(mensajeRecibido == "PROPOSE" &&
          mensajeRecibido.content() != null){
     movimientoRealizado = mensajeRecibido.content
}
```

  - **Actualizar en cada turno el tablero, para visualizar el efecto del movimiento.**

```
if(partidaActiva){
     tablero.añadir(movimientoRealizado)
}
```

  - **Enviar información acerca de la partida al agente grupo.**

```
if(partidaActiva && ganador){
     enviar(Clasificacion));
}
```

3.3. Ontología

3.3.1. Para la de dominio y aplicación

- ***Conceptos***
  - **ClasificaciónJuego:** Este concepto representará la clasificación mediante una lista de jugadores y una lista de puntuación organizadas en la EEDD *List.*
  - **Gestor:** Este concepto representará al AgenteCentralJuego con su AID y su nombre en un String.
  - **Grupo:** Este concepto representará al AgenteGrupoJuegos con su AID y su nombre en un String.
  - **Informe:** Este concepto representará una **Incidencia** que denominaremos detalle que nos informará del problema o rechazo que surja en las comunicaciones entre agentes.
  - **Juego:** Este concepto vendrá representado por:
    - ❑ ***String idJuego:*** indicará el id de la partida que se está jugando
    - ❑ ***int minVictorias:*** indicará el mínimo de victorias que necesitará un jugador en una partida para poder ganar.
    - ❑ ***ModoJuego modoJuego:*** indicará si es Torneo o partida normal.
    - ❑ ***TipoJuego tipoJuego:*** indicará si es Barcos o Conecta4.
  - **Jugador:** Este concepto representa a un jugador por un String que será el nombre y su AID de agente.
  - **Posición:** Este concepto vendrá dado por las coordenadas x e y del movimiento que querrá realizar respecto a un juego
  - **Tablero:** Este concepto almacenará las dimensiones en x e y del tablero que se vaya a jugar.

- ***Acciones***
  - **CompletarJuego:** Esta acción representará:
    - ❑ ***Juego juego:*** indicará si el juego es Torneo o Partida individual.
    - ❑ ***Concept tipoJuego:*** indicará si el juego es Barcos o Conecta4 en nuestro caso.
    - ❑ ***List listaJugadores:*** indicará la lista de jugadores que participarán en el juego.
  - **InformarJuego:** Esta acción representará al **Gestor** explicado anteriormente en el apartado *Conceptos.*
  - **PedirMovimiento:** Esta acción vendrá representada por un **Juego** y un **Jugador** que esté activo en la partida para poder pedirle que le diga el movimiento que va a realizar.
  - **ProponerJuego:** Esta acción representará el **Juego** que quiere proponer y el tipo de Juego de este (Barcos o Conecta4).

- ***Predicados***
  - **DetalleInforme:** Este predicado representará el *Juego* que es y un tipo **Concept** llamado detalle que representará detalladamente la información del problema del informe.
  - **JuegoAceptado**: Este predicado representará cuando un agente acepte el juego propuesto por lo que vendrá representado por el **Juego** propuesto y por un **Concept** que será el agenteJuego.
  - **Motivación**: Este predicado representará un **Motivo** por el cual no se acepta o si un movimiento o partida.

- ***Conceptos y Predicados para Conecta4***
  - **EstadoJuego:** Este predicado indicará el estado de un Juego, este decir, si la partida sigue activa o ha finalizado por la falta de jugadores o por un ganador. Vendrá representada por un **Juego** y un **Estado** al que llamaremos estadoJuego.
  - **Ficha:** Este concepto lo único que representará el **Color** de la ficha para diferenciar a los jugadores. En este caso solo serán azul y rojo.
  - **JuegoConecta4:** Este concepto creará un tablero con las dimensiones en las que se jugará.
  - **Movimiento:** Este concepto vendrá dado por una **Ficha** para identificar al jugador y una **Posicion** con las coordenadas donde se quiera colocar la ficha.
  - **MovimientoEntregado:** Este predicado vendrá dado por el Juego que será Torneo o Partida normal y el Movimiento explicado anteriormente.

- ***Conceptos, Acciones y Predicados para Barcos***
  - **PosicionBarcos:** Este predicado representará el tipo de Juego que se estará jugando y una lista con las diferentes posiciones asociadas a cada barco que tenga un jugador.
  - **ResultadoMovimiento:** Este predicado representará el tipo de Juego que se estará jugando, la **Posicion** del movimiento donde se ha realizado, y el **Efecto** que en este caso puede ser ***{tocado, hundido, agua}.***
  - **ColocarBarcos**: Esta acción vendrá definida por el Juego.
  - **EstadoJuego:** Este predicado indicará el estado de un Juego, este decir, si la partida sigue activa o ha finalizado por la falta de jugadores o por un ganador. Vendrá representada por un **Juego** y un **Estado** al que llamaremos estadoJuego.

- **Localización:** Este concepto representará la posición y el barco. Sus atributos son:
  - ❑ **TipoBarco barco:** Indicará el tipo de barco que se va a colocar en función de su longitud.
  - ❑ **Posición posicion:** Indicará desde que posición empieza a colocarse el barquito.
  - ❑ **Orientación:** Indicará la orientación si es horizontal o vertical.
- **JuegoBarcos:** Este concepto definirá un tablero con las dimensiones correspondientes y el número de barcos según el tipo con el que se está jugando.
- **MovimientoEntregado:** Este predicado vendrá dado por el Juego que será Torneo o Partida normal y el Movimiento explicado anteriormente.

3.3.2. Validación mediante las preguntas de competencia

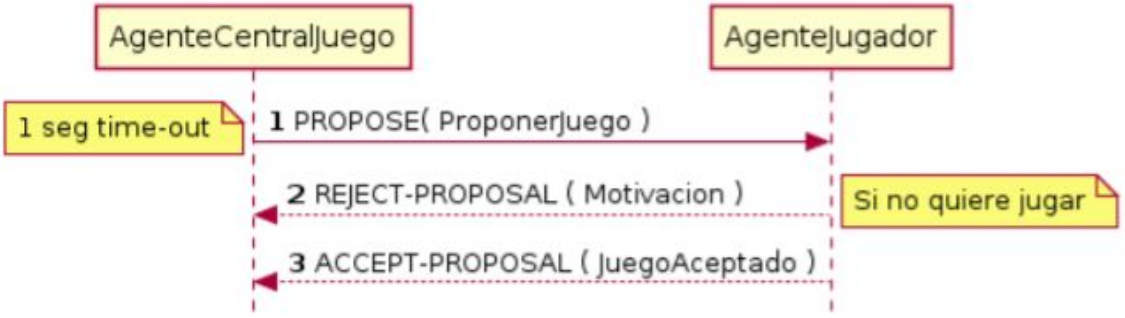
3.4. Comunicación entre agentes

A continuación expondremos los protocolos necesarios que necesitaremos en las comunicaciones establecidas en el análisis

3.4.1. Asignación de protocolos con los elementos de personalización necesarios

- **Proponer Juego**
  El AgenteCentralJuego tendrá que localizar a los jugadores que quieran jugar y se propondrá una partida individual o un torneo (vendrá implícito en la Ontología ***ProponerJuego***).

  El protocolo que utilizaremos será **Propose** y tendrá dos posibles respuestas por parte del AgenteJugador:
  - **REJECT-PROPOSAL:** Que enviará una ontología de tipo ***Motivación*** indicando lo que le lleva a rechazar el juego.
  - **ACCEPT-PROPOSAL:** Que enviará una ontología de tipo ***JuegoAceptado*** de manera que confirma al emisor del protocolo que si desea jugar.



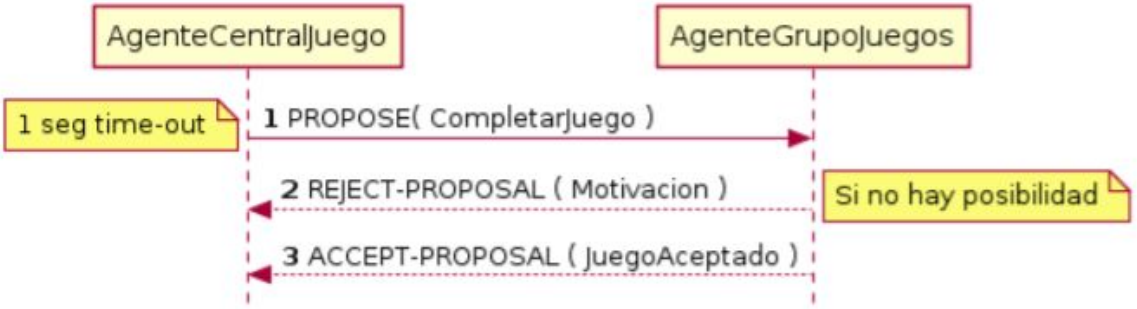
- ## Completar Juego

  El AgenteCentralJuego una vez tenga los jugadores necesarios para jugar, solicita al AgenteGrupoJuegos que complete el juego propuesto anteriormente. La ontología clave que en este caso enviaremos será ***CompletarJuego*** que vendrá dada por los siguientes atributos:

```
public class CompletarJuego implements AgentAction {
    private Juego juego;
    private Concept tipoJuego;
    private List listaJugadores;
```

  En ella enviaremos la lista de jugadores que participarán en el juego e indicará si es Torneo o partida Individual.
  El protocolo que utilizaremos será **Propose** y tendrá dos posibles respuestas por parte del AgenteGrupoJuegos:

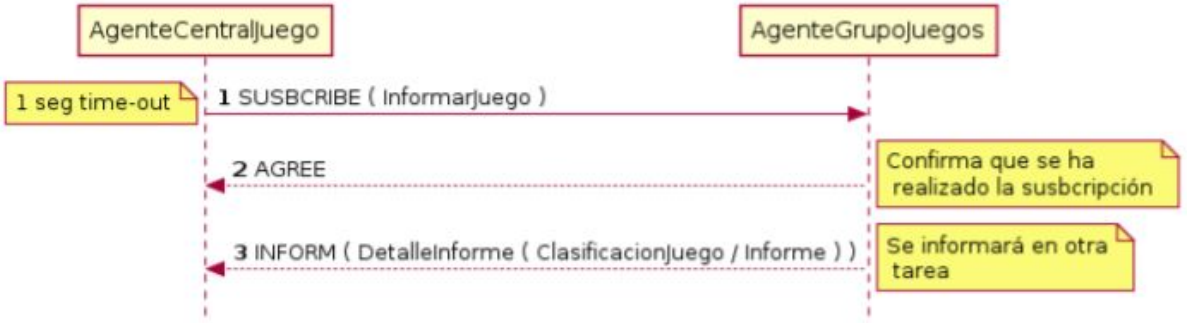
  - **REJECT-PROPOSAL:** Que enviará una ontología de tipo ***Motivación*** indicando si no hay posibilidad de jugar.
  - **ACCEPT-PROPOSAL:** Que enviará una ontología de tipo ***JuegoAceptado*** de manera que confirma al emisor del protocolo que si desea jugar.



- ## Informar Juego

  El AgenteCentralJuego una vez que el AgenteGrupoJuegos confirme un juego, realizará una suscripción para que se le informe de la clasificación final para los jugadores que han participado en el juego.
  El protocolo que usaremos en este caso será el **Subscribe** donde el AgenteGrupoJuegos tendrá dos posibles respuestas:

- **AGREE:** Indicará que confirma por primera vez la suscripción al AgenteCentralJuego.
- **INFORM:** En este caso se irá informando de la clasificación con la Ontología ***ClasificacionJuego*** donde vendrá una lista de los jugadores y otra lista con los puntos. También podrá informarse con una Ontología ***Informe*** donde se detalla el problema.



- ## Jugar una Partida (Barcos)

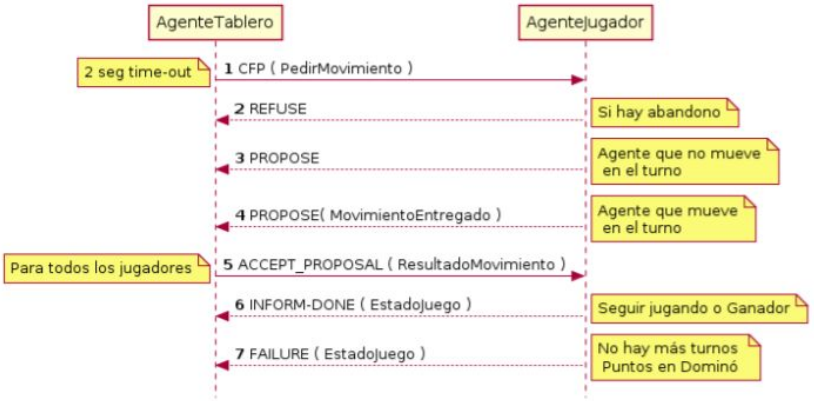
  Para jugar una partida comunicaremos nuestros Agentes Tablero con los Agentes Jugador implicados.
  En este caso usaremos el protocolo **ContractNet** para la comunicación.

  El primer mensaje que enviará el AgenteTablero será junto con la Ontología ***PedirMovimiento*** donde el AgenteJugador tendrá tres posibles respuestas:

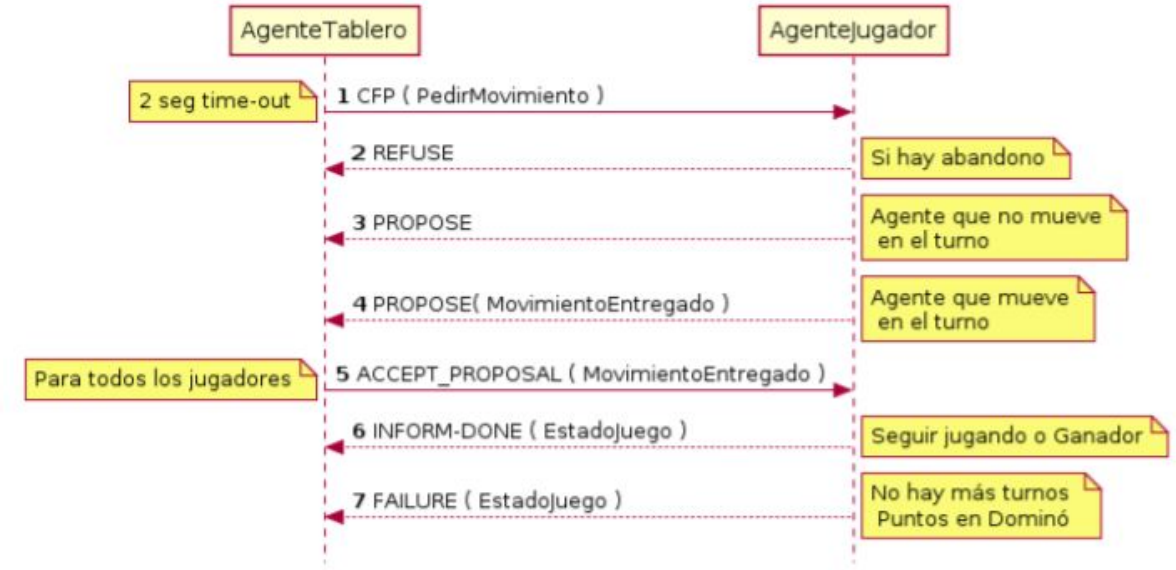
  - **REFUSE:** Si quiere abandonar la partida o surge algún problema con el agente.
  - **PROPOSE:** El Agente no mueve turno porque no puede o surge algún problema.
  - **PROPOSE(MovimientoEntregado):** Agente mueve y le pasa el movimiento al AgenteTablero.

  Ahora el AgenteTablero enviará un segundo mensaje para todos los jugadores informando del movimiento que se ha realizado mediante un **ACCEPT_PROPOSAL(ResultadoMovimiento)** donde el AgenteJugador tendrá dos respuestas posibles:

  - **INFORM-DONE(EstadoJuego):** indica si el jugador ya ha ganado la partida o sigue jugando después de saber el movimiento.
  - **FAILURE(EstadoJuego):** no hay mas turnos para jugar.

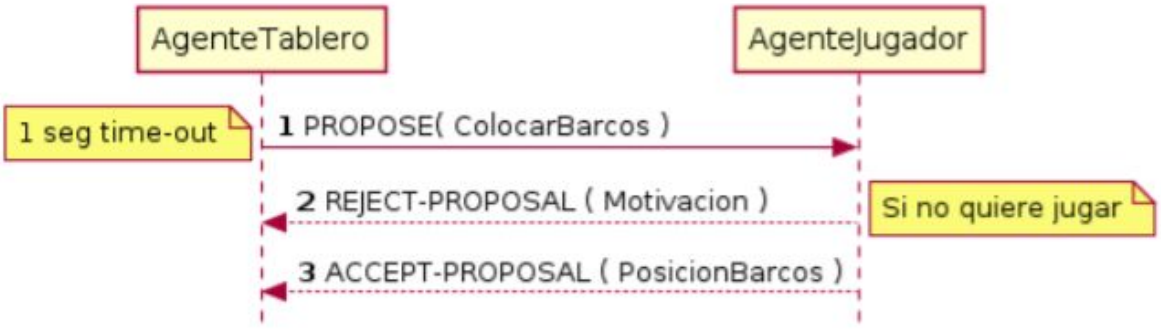


- **Jugar una Partida (Conecta-4)**
  La única diferencia respecto al otro es que el AgenteTablero envía la Ontología de ***MovimientoEntregado.***



- **Preparación Partida (Barcos)**
  El AgenteTablero antes de comenzar a jugar la partida de los Barcos debe saber donde coloca cada jugador cada pieza de los barcos, de esta forma mediante el protocolo **PROPOSE** envía un mensaje con la Ontología ***ColocarBarcos*** donde el AgenteJugador tiene dos posibles respuestas:

  - **REJECT-PROPOSAL(Motivacion):** Si no quiere jugar le envía los detalles del porqué.
  - **ACCEPT-PROPOSAL(PosicionBarcos):** Le envía la posición de los barcos con la Ontología creada.



3.4.2. Identificación con tareas de los agentes

- **AgenteGrupoJuegos**
  Este agente tiene asignados los protocolos **Propose** y **Subscribe** que vendrán representados por las siguientes tareas:
  - ***completarJuegoResponder:*** Esta tarea vendrá asociada al protocolo **Propose** que responde al mensaje del AgenteCentralJuegos el cual le propone participar en un juego y este mediante el protocolo le responde.
  - **informarJuegoASubs:** Esta tarea vendrá asociada al protocolo **Subscribe** donde el AgenteCentralJuegos tendrá que haberse suscrito previamente a nuestro protocolo para nosotros desde esta tarea ir informando sobre el transcurso de la partida.

- **AgenteTablero**
  Nosotros tendremos representados nuestros agentes tablero uno para cada juego de manera que la comunicación será muy similar. Aquí tenemos asignados los protocolos **Contract-Net** y **Accept-Propose.** Las tareas serán las siguientes:
  - **preparaciónPartidaIniciador:** Esta tarea será solo del AgenteTableroBarquitos, que será la encarga de colocar los barquitos al recibir el protocolo **Propose** y responder con las posiciones.
  - **jugarPartidaIniciador:** Esta tarea vendrá asociada al protocolo **Contract-Net** donde realizaremos la primera comunicación con nuestro AgenteJugador.
  - **comunicarMovimiento:** Esta tarea vendrá asociada al protocolo **Accept-Propose** donde informaremos a todos los jugadores de la partida el movimiento que se ha realizado.