



# TESINA DE LICENCIATURA

**Título:** Coordinación de Grupos en Juegos Móviles basados en Posicionamiento

**Autores:** Fernando Gabriel Inafuku – Pablo Fernando Galella

**Director:** Dra. Cecilia Chaliol

**Codirector:** Dra. Silvia Gordillo

**Carrera:** Licenciatura en Sistemas.

## Resumen

*Coordinar tareas para ser llevadas a cabo por un grupo de personas permite obtener una acción unificada y uniforme donde la suma de los esfuerzos individuales se potencian en pos de resolver actividades. En el caso de los Juegos Móviles basados en Posicionamiento, requiere que los participantes se coordinen en posiciones (o áreas) determinadas del ambiente físico para poder avanzar en el juego.*

*Se tomo como base la tesina de grado de Matías Apezteguía y Darío Rapetti denominada “Juego Educativo Móvil Colaborativo”, la cual considera la posición de los participantes del grupo para brindar consignas y además contempla aspectos de colaboración. En base a dicha tesis, se extendió tanto el modelo como el prototipo funcional, para incorporar aquellos aspectos relacionados a la coordinación de grupos en un determinado espacio físico. Esta extensión contempla distintas estrategias de coordinación, por ejemplo, que todos los integrantes del grupo o que un porcentaje de los mismos estén en una determinada área física en un momento del juego móvil. Cuando se cumplen estas condiciones el juego brinda, por ejemplo, una consigna al grupo para así continuar el mismo. Es decir, esta coordinación implica que los integrantes del grupo se movilicen físicamente dentro del espacio físico.*

## Palabras Claves

Coordinación de grupos en espacios físicos, Juegos Móviles basados en Posicionamiento; Estrategias de Coordinación; Puntos de Interés de Coordinación.

## Trabajos Realizados

Se analizaron distintos juegos móviles basados en posicionamiento que utilizan la coordinación de tareas como parte de la lógica de los mismos.

En base al análisis realizado, se propuso una solución de modelado, haciendo hincapié en la coordinación de grupo, en particular, en puntos de interés relevantes dentro del espacio físico del juego móvil.

Se creó un prototipo funcional en base al modelo propuesto, aprovechando el juego móvil previamente definido en la tesis usada como base. Usando el prototipo, se simularon escenarios de juego, para poder probar la coordinación de grupo en distintos puntos de interés relevantes dentro del espacio físico.

## Conclusiones

Se presentó una solución de modelado para la coordinación de grupos en juegos móviles basados en posicionamiento. El modelo propuesto usa aspectos del juego de la tesis tomada como base, pero podría ser extendido para otros dominios implementando la lógica de cada uno de ellos. Pudiendo reusar las estrategias de coordinación definidas en dicho modelo.

El prototipo funcional fue probado con simuladores, si bien sirvió como una primera prueba, se necesitan pruebas de campos para ver el real funcionamiento.

## Trabajos Futuros

- Definir “Estrategias de Coordinación” a nivel de punto de interés de coordinación, en lugar de que sea una para todo el *Juego*. Es decir, que se puedan combinar en un mismo juego diferentes estrategias de coordinación por ahora solo se tiene una por juego.
- Por ahora la dinámica de juego móvil (tanto a nivel de modelado como del prototipo) es lineal, se debería analizar otro tipo de dinámicas, por ejemplo, recorrido libre para determinar cómo se deberían comportar las estrategias de coordinación en esta situación.
- Realizar pruebas de campo para probar el real funcionamiento, por ejemplo, del GPS.

# Índice

|   |     |
|---|-----|
| 1. Introducción.....  | 2   |
| 1.1 Motivación.....   | 2   |
| 1.2 Objetivo .....  | 3   |
| 1.3 Estructura de la Tesis .....  | 4   |
| 2. Background.....  | 5   |
| 2.1 Who killed Hanne Holmgaard? [ Paay et al., 2007].....                 | 5   |
| 2.2 Life on the Edge [Benford et al., 2005] .....                         | 8   |
| 2.3 ContextsContacts [Oulasvirta et al., 2005] .....                      | 14  |
| 2.4 Coordinación de tareas en un Hospital [Ren et al., 2007] .....        | 17  |
| 2.5 Catch Bob! [Nova et al., 2009] .....                                  | 22  |
| 2.6 LaMOC (Location Aware Mobile Cooperation) [Gu et al., 2009].....      | 24  |
| 2.7 Framework para interacción colaborativa [Lundgren et al., 2015] ..... | 25  |
| 2.8. Resumen de aspectos de coordinación .....                            | 29  |
| 3. Modelo Propuesto.....  | 34  |
| 3.1 Descripción del Modelo usado como base .....                          | 34  |
| 3.2 Explicación de la problemática a resolver .....                       | 37  |
| 3.3 Descripción del Modelo Propuesto .....                                | 39  |
| 4. Prototipo Implementado.....  | 63  |
| 5. Simulación de un Juego.....  | 82  |
| 6. Conclusiones y Trabajos Futuros.....                                   | 100 |
| 6.1 Conclusiones .....  | 100 |
| 6.2 Trabajos Futuros .....  | 102 |
| Bibliografía .....  | 104 |
| Anexo A: Funcionalidad del Prototipo tomado como base.....                | 105 |
| Anexo B: Modificaciones de código realizadas .....                        | 110 |

# 1. Introducción

## 1.1 Motivación

En el mundo moderno, cualquier actividad con cierta complejidad; como puede ser la fabricación de un objeto, la construcción de una torre o el desarrollo de una pieza de software, puede encararse mediante un enfoque donde se divida el total del trabajo en varias tareas para su realización y la coordinación de éstas para poder completar la actividad.

Coordinar las tareas ejecutadas por un grupo de personas permite obtener una acción unificada y uniforme donde la suma de los esfuerzos individuales se potencian en pos de resolver actividades de forma exitosa. De esta manera, la coordinación de personas en un grupo no solo facilita el intercambio de la información, la generación de nuevas ideas y la simplificación de los problemas; sino también la resolución de problemas y ejecución de tareas de manera conjunta.

En el contexto de Juegos Colaborativos Móviles, encontramos varios casos donde la coordinación de grupos y/o personas está presente. Por empezar, podemos citar el juego presentado en [Paay et al., 2007]; donde dos participantes que simulan ser detectives trabajan en conjunto para resolver un caso. Ambos personajes, si bien desempeñan sus tareas de manera independiente, necesitan en ciertos puntos del juego coordinarse y reunirse físicamente para poder intercambiar información y determinar entre los dos a qué lugares físicos deben desplazarse para continuar con el juego. Otro juego donde también vemos aplicada la coordinación de participantes, es el expuesto en [Benford et al., 2005]; donde grupos de niños simulan ser leones que exploran una sabana virtual en busca de recursos para sobrevivir. El juego exige una cierta colaboración entre los participantes, dado que si se coordinan y ejecutan las tareas en conjunto, incrementan sus chances de obtener resultados exitosos. Por ejemplo, mientras más participantes se coordinen para lanzar un ataque (en la jerga del juego, esto representaría leones que cazan a sus presas) mayor serán las probabilidades de un ataque exitoso.

Los citados juegos, entre muchos otros de su especie, ya brindan una solución para el tema de la coordinación de grupos. Sin embargo, observamos que ninguno de ellos posee un modelo subyacente que represente a la solución. Es allí donde vemos la necesidad de proponer un modelo general para la coordinación de grupos en juegos móviles basados en posicionamiento, que represente los conceptos involucrados en la misma. Esta es la motivación principal de nuestro trabajo.

Para darle solución y forma al diseño de la solución propuesta, tomaremos de base el trabajo presentado en [Apezteguía and Rapetti, 2014] donde ya se cuenta con un modelo que incluye conceptos asociados a los *Juegos Colaborativos Móviles* basados en posicionamiento, como son la idea de grupo, posicionamiento, colaboración, etc. Asimismo, también se presenta en el mencionado trabajo un prototipo desarrollado, el cual también usaremos como base en esta tesis. De esta manera, el trabajo expuesto en [Apezteguía and Rapetti, 2014] nos facilitará aspectos de este tipo de aplicaciones, cabe destacar que estos autores no abarcan la temática de coordinación de grupo, algo que será el foco de nuestra tesis.

## **1.2 Objetivo**

El objetivo principal de este trabajo es brindar una solución de modelado para la coordinación de grupos en juegos móviles basados en posicionamiento. Para tal fin, tomaremos de base el modelo presentado en [Apezteguía and Rapetti, 2014] y propondremos una extensión de dicho modelo que incluya los conceptos asociados a la coordinación de grupos. El modelo propuesto es flexible y extensible permitiendo que el mismo pueda evolucionar en el tiempo agregándole otras características a parte de las presentadas en esta tesis.

El modelo propuesto hace hincapié en las coordinaciones de grupos en un espacio físico determinado. Por ejemplo, requerir que todos los integrantes del grupo o algunos integrantes del mismo, estén posicionados en una determinada área. Es decir, podría haber distintas condiciones que establezcan cómo se debe dar la coordinación del grupo. En el caso de los juegos móviles basados en posicionamiento, el mismo no podrá continuar hasta que se realice la coordinación del grupo, y una vez que se realiza dicha coordinación, el juego brinda información de cómo continuar (por ejemplo, brindar una pregunta relacionada al juego). Cabe aclarar que este tipo de juegos requiere que los usuarios se muevan físicamente.

En base al modelo propuesto se implementó un prototipo funcional para mostrar cómo se puede realizar la coordinación de grupos. Dicho prototipo se desarrollo como una extensión del prototipo expuesto en [Apezteguía and Rapetti, 2014].

## **1.3 Estructura de la Tesis**

*Capítulo 2.* En este capítulo se presentan diferentes enfoques que definen aspectos generales involucrados en la coordinación de personas para la resolución de ciertas tareas en conjunto, las cuales se realizan en algún lugar físico particular. Algunos de estos enfoques se dan en el contexto de aplicaciones móviles que mezclan el mundo real y se integran con el mismo, generando una experiencia de interacción entre las personas, las aplicaciones y el mundo físico. Otras, plantean soluciones teóricas para optimizar la coordinación y pueden ser aplicadas en contextos muy particulares como personas en una escuela o personal de un hospital. Para esta tesis se realizó un análisis de diferentes casos donde la coordinación de grupo es aplicada con el objetivo de resolver una tarea.

*Capítulo 3.* En este capítulo se describe el modelo usado como base, y se plantea en detalle la problemática que se quiere resolver. Luego, se presenta la extensión del modelo propuesta como solución para el problema planteado.

*Capítulo 4.* En este capítulo se describen las funcionalidades del prototipo implementado. Este prototipo usa el modelo definido en el Capítulo 3.

*Capítulo 5.* En este capítulo se realiza la simulación del prototipo propuesto donde veremos cómo es la interacción de los subgrupos y cómo se desarrolla un juego con las nuevas características de coordinación introducidas en nuestro trabajo.

*Capítulo 6.* En este capítulo se presentan las conclusiones relacionadas a la tesis y luego se mencionan algunos trabajos futuros que se desprenden de la misma.

## **2. Background**

En este capítulo presentamos diferentes enfoques que nos permiten determinar aspectos generales involucrados en la coordinación de personas para la resolución de ciertas tareas en conjunto, las cuales se deben realizar en algún lugar físico particular. Algunos de estos enfoques se dan en el contexto de aplicaciones móviles que mezclan el mundo real y se integran con el mismo, generando una experiencia de interacción entre las personas, las aplicaciones y el mundo físico. Otras, plantean soluciones teóricas para optimizar la coordinación y pueden ser aplicadas en contextos muy particulares como personas en una escuela o personal de un hospital.

### **2.1 Who killed Hanne Holmgaard? [ Paay et al., 2007]**

Este juego está enmarcado en una historia que trata de un misterioso asesinato en las calles de la ciudad de Aalborg, Dinamarca. Es un juego móvil colaborativo basado en posicionamiento, donde dos participantes trabajan en conjunto para resolver el misterio. Cada participante representa un personaje diferente en el juego, y deben trabajar juntos para resolver el crimen. El sistema se basa en la ubicación de los participantes y provee ciertos episodios basándose en la ubicación actual. La interacción de las personas con el juego se da mediante PDAs y auriculares, y también por el desplazamiento físico a través de la ciudad.

Los dos participantes además de contar con PDAs y auriculares, cuentan con un mapa en papel del centro de la ciudad (que ayuda en el desplazamiento y donde están marcados con símbolos los diferentes puntos a donde deben dirigirse los personajes) y con una mochila llena de sobres cerrados. Para la resolución del caso, ambos participantes caminan la ciudad visitando puntos claves de la historia en un orden particular de acuerdo a cómo se va desarrollando la historia. A excepción del punto inicial, el resto de los puntos a recorrer no son visibles para las personas desde un principio. La historia es narrada utilizando audio, imágenes y texto. La interacción se da tanto por la selección de opciones en la pantalla así como también por la interacción con lugares físicos y objetos tales como inscripciones o símbolos en edificios, mapas, etc.

Al comienzo del juego se les presenta a ambos detectives (participantes) la escena del crimen, las pistas iniciales y algunas evidencias. De ahí en más trabajan en conjunto para resolver el caso.

Al ser dos personajes diferentes, durante el transcurso del juego se le permite a los dos jugadores separarse y seguir un camino distinto, en cuyo caso interrogan a diferentes testigos o le siguen el rastro a diferentes pistas. En todo caso, ninguno de los dos recibe la información completa para resolver el misterio por sí solo, de manera que uno precisa de la información recolectada por la otra persona para obtener un entendimiento general de la trama y, discusión mediante, poder resolver ambos el caso planteado. Luego, en los puntos clave del juego, ambos deben reunirse físicamente para discutir sobre lo aprendido. Una vez que ambos participantes se reúnen y discuten sobre la información recolectada, deben dirigirse hacia la próxima escena en un lugar diferente. Para el desplazamiento por la ciudad, los participantes hacen uso del mapa en papel provisto, donde tienen marcados los lugares claves a donde deben dirigirse. Sin embargo, cada ubicación se anota con un símbolo en lugar de números o letras, de manera de que sea imposible para los detectives por sí mismos, saber a dónde ir a continuación. En cada punto de encuentro del juego (comenzando por el punto inicial, que es el único visible al comienzo del mismo), cada detective recibe en su dispositivo móvil medio símbolo, por lo cual para obtener el símbolo completo (y por consiguiente saber el próximo punto al cual dirigirse) deben reunirse ambos y compartir su mitad mutuamente. Con esta información obtienen el símbolo completo y simplemente observando el mapa en papel ya saben a donde tienen que ir a continuación. Esta situación se refleja en la Figura 1.

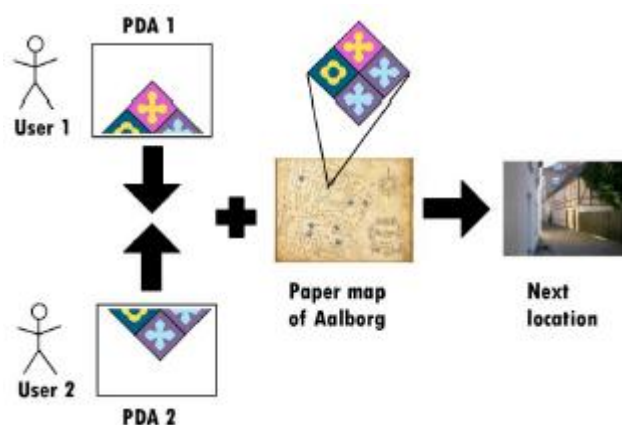


Figura 1: Ambos usuarios comparten su mitad para obtener el símbolo completo y con la ayuda del mapa saber hacia dónde dirigirse [Paay et al., 2007]

En la Figura 2 vemos cómo se lleva a cabo la colaboración para inferir el símbolo completo en los PDAs.



Figura 2: Alineando los dos PDA se obtiene la figura completa [Paay et al., 2007]

Cabe destacar que los símbolos son diseñados de manera que el símbolo completo no pueda ser inferido conociendo solo la mitad.

### ***Coordinación Involucrada***

Con respecto a la coordinación implementada en el juego, existe una única estrategia que consiste en reunir a ambos participantes en un mismo lugar, comenzando por el punto inicial que es el único visible para ambos al comienzo del juego. Una vez que ambos se encuentran en un punto, además de intercambiar la información obtenida y opiniones sobre los interrogatorios realizados, cada participante visualiza en su PDA la mitad de un símbolo que representa el siguiente punto donde ambos jugadores deben encontrarse. Luego, al alinear ambos sus dispositivos móviles obtienen el símbolo completo y por consiguiente el próximo lugar a donde deben dirigirse (tal como se visualizó en la Figura 2). Luego, hasta el encuentro en el próximo lugar, cada participante puede tomar un camino independiente o pueden ir juntos. Si uno de los participantes termina sus tareas antes que el otro y arriba al siguiente punto sin que el otro haya llegado, no puede hacer nada más que esperar a su compañero, dado que ambos deben finalizar sus interacciones y estar en el lugar para que se les provea la mitad del símbolo correspondiente, alinear los dispositivos y conocer el siguiente punto al cual dirigirse.

Según el análisis de la coordinación se deduce que el camino a seguir por los participantes es un camino lineal, que se va descubriendo a medida que se van recorriendo los puntos (salvo el punto inicial que es visible en el comienzo del juego). Si bien los participantes visualizan todos los puntos de encuentro en el mapa en papel (están marcados con símbolos) no saben a priori el orden en el que deben recorrerse. Eso lo van conociendo punto por punto, en la medida que van encontrándose en cada uno de ellos.



## **2.2 Life on the Edge [Benford et al., 2005]**

Es un juego colaborativo basado en posicionamiento donde un grupo de niños aprende sobre ecología en una sabana Africana (la cual esta simulada virtualmente). Un grupo de seis niños juegan representando ser leones explorando una sabana virtual que se delimita en un campo abierto de juego de un colegio. Los niños se equipan con PDAs y habiendo disponibilidad de WiFi y GPS, se mueven en el campo de juego explorando los variados terrenos de la sabana y descubriendo los recursos que los leones precisan para sobrevivir.

El juego se divide en tres niveles. En el primero, los leones exploran y demarcan su territorio. En el segundo y tercer nivel, los leones cazan por comida en la temporada húmeda y seca correspondientemente. Cada león tiene que maximizar su salud, con puntos que se obtienen de comer y beber, y se pierden por estar hambrientos, deshidratados o por haber sido atacado por otros animales. Asimismo, los leones pueden morir y resucitar luego de cumplir una cierta penalidad de tiempo.

El juego requiere colaboración, de la misma manera que en una manada los leones tienen que trabajar juntos para sobrevivir; por lo que se espera que los jugadores trabajen juntos y decidan que animales atacar y en qué número para que los ataques sean exitosos.

Técnicamente hablando, los jugadores se mueven en el campo con su GPS reportando sus posiciones, que se mapean con sus posiciones en la sabana virtual. Cada vez que ingresan a una nueva ubicación, se le provee al usuario un sonido y una imagen asociada y posiblemente una acción. Es muy importante, que los jugadores estén todos en una misma ubicación para compartir información y actuar en conjunto. A cada segundo, los dispositivos transmiten su ubicación a un servidor central vía WiFi y éste responde enviando un sonido, una imagen y la información de estado al dispositivo móvil, revelando la sabana virtual y sus habitantes.

Cada jugador invoca una acción como “atacar” a partir de un botón que aparece en la pantalla y depende de la ubicación actual del usuario y del nivel en que se encuentre. Una notificación de esta acción se envía por WiFi al servidor el cuál responde con una notificación de éxito u fracaso.

El servidor por su parte implementa las reglas del juego. También contiene la información que especifica las características de los animales que pueden aparecer en el mapa, incluyendo cuántos leones se necesitan para atacarlos satisfactoriamente, cuánta salud se gana comiéndoselo, etc.

Para la implementación de este juego hay que lidiar con los problemas inherentes de las tecnologías de posicionamiento. Los más comunes tienen que ver con la disponibilidad, resolución, latencia y precisión. Como ejemplo, el GPS cuenta con muchos black-spots que son lugares donde el receptor no puede leer suficientes satélites como para obtener la posición circunstancial, además ofrece una resolución de un metro o menos y una precisión de varios metros.

Estos problemas afectan la experiencia del usuario dado que las posiciones reportadas no se corresponden con las actuales provocando comportamientos indeseados.

Teniendo en cuenta los aspectos de coordinación, durante el desarrollo del juego puede haber un sólo ataque en curso para una ubicación en el mismo espacio de tiempo. Por eso cuando un jugador ataca, o inicia un nuevo ataque, o se une a uno en curso. Cuando sucede lo primero, el resto de los jugadores tiene 10 segundos para unirse al ataque (y de esta manera hacer un ataque en conjunto) antes de que el ataque se de por resuelto. Durante el ataque se muestra en la pantalla de los dispositivos una imagen especial de ataque y se congela allí durante los 10 segundos. Esta ventana de tiempo y el congelamiento de la imagen tienen como objetivo dar la posibilidad de que todos los jugadores puedan coordinar correctamente sus acciones. Una vez resuelto el ataque, todos los jugadores involucrados reciben un mensaje de texto con el resultado del mismo. Luego, se les actualiza su salud, hambre, sed, etc.

En líneas generales, los autores describen en [Benford et al., 2005] que el juego resultó ser una experiencia satisfactoria para los jugadores, dado que se divirtieron, pudieron formar subgrupos, se juntaron y separaron frecuentemente (durante los 10 segundos que dura un ataque) para ejecutar ataques en conjunto, y la formación de grupos resultó ser totalmente fluida.

Sin embargo, también encontraron grandes dificultades de coordinación, las cuales se mencionan a continuación:

Caso 1: Problemas de coordinación que surgen entre algunos participantes que si bien están físicamente uno al lado del otro, algunos de ellos encontrándose en el borde de una región/zona pueden ir “alternando su ubicación” (entrando/saliendo de la zona), no pudiendo sumarse a un ataque. Las principales causas de esto, según los autores, son la imprecisión y la latencia del sistema. Se ve en la Figura 3 como *Nala* y *Dandelion* tienen problemas para sumarse al ataque en conjunto, con resultados dispares. Al inicio de los 10

segundos, *Nala* se encuentra fuera de la zona de ataque pero al finalizar dicho tiempo logra ser detectada y es tenida en cuenta para el ataque en conjunto. *Dandelion*, por su parte, si bien estaba dentro de la zona al comenzar la ventana, al expirar el tiempo, se encuentra fuera de ella no pudiendo sumarse al ataque.

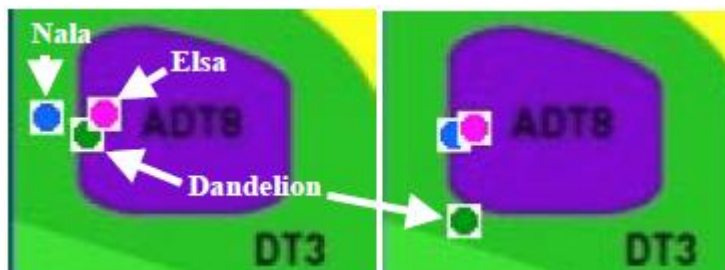


Figura 3: *Dandelion* queda finalmente fuera del ataque en conjunto [Benford et al., 2005]

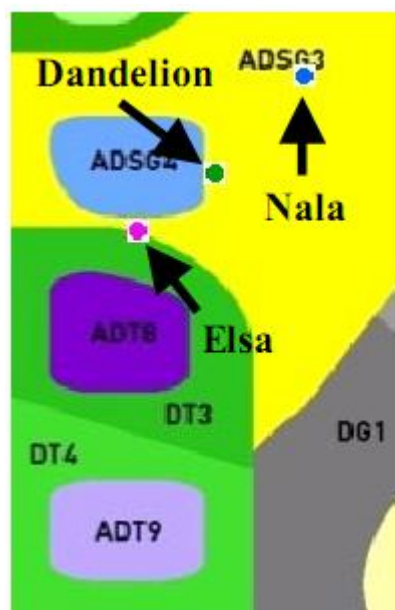
Caso 2: Los jugadores utilizan marcadores físicos para localizar con facilidad aquellos puntos donde “tienen comida”. Sin embargo, los problemas de coordinación persisten. Un mismo jugador, estando siempre parado en la misma ubicación, puede ver a su presa y repentinamente dejar de verla. A criterio de los autores, esto es debido a la imprecisión del GPS.

Caso 3: Un grupo de jugadores se predispone a lanzar un ataque, cuando uno de ellos tiene problemas para visualizar a la presa. En un primer momento, los jugadores que sí podían verla, lanzan el ataque obteniendo un resultado negativo. Luego, se alejan. En ese momento, el jugador que en un principio no podía ver a la presa, obtiene la visión de ésta en la pantalla. Lanza un ataque y falla por haber iniciado el ataque solo. Nuevamente, los autores señalan que estos problemas son debidos a la latencia e imprecisión. En la Figura 4, se ve como los niños se encuentran todos reunidos uno al lado del otro, y sin embargo el sistema ve a uno de ellos (*Dandelion*) un poco más alejado y fuera de la zona de ataque.



Figura 4: La posición física de los niños no se corresponde con lo que ven en las pantallas [Benford et al., 2005]

Caso 4: En este caso un jugador, recorriendo velozmente el campo de juego (*Elsa*), lanza un ataque rápido al pasar tras obtener una visión “fugaz” de una presa. En el momento de la resolución del ataque, *Elsa* ya se encuentra lejos de la posición de su presa y los demás jugadores cercanos a su posición (*Nala* y *Dandelion*) que intentaron sumarse al ataque no lo consiguieron porque nunca llegaron a verla (ya que al momento de reunirse los tres estaban lejos de la presa). Sin embargo, a pesar del ataque solitario, el mismo fue exitoso. Esta situación la vemos reflejada en la Figura 5.



**Figura 5: *Nala* y *Dandelion*, lejos de la presa (presente en la zona ADT8) no pueden sumarse al ataque iniciado por *Elsa* al pasar [Benford et al., 2005]**

El ataque coordinado en este juego es iniciado por uno de los participantes y pueden unirse al mismo hasta un máximo de cinco participantes más. No existe una estrategia en particular para llevar a cabo dicho proceso. Las únicas condiciones para realizarlo es que todos se unan a la coordinación dentro de la ventana de diez segundos que dura y estén todos los involucrados dentro de la misma zona de ataque (esto implica que no solo se encuentren físicamente en la misma zona, sino también que sean detectados por el sistema en dichas posiciones). Si en el lapso de los diez segundos de haberse iniciado el ataque, ningún compañero logra unirse, el ataque se ejecuta en solitario teniendo menos probabilidades de ser exitoso. Es decir, a mayor cantidad de participantes en un ataque, mayor es la probabilidad de éxito. Así también, hay que tener en cuenta que dependiendo del tamaño de la víctima, a mayor tamaño de ésta, mayor es la cantidad de

participantes para lograr un ataque satisfactorio.

En cuanto a los inconvenientes encontrados por los autores en [Benford et al., 2005] al querer coordinar un ataque, el problema más obvio fue el de no poder establecer un contexto compartido. Ciertos jugadores que estaban parados uno al lado de otro, con el fin de poder coordinar sus acciones, no recibían la misma información en sus dispositivos. Mientras algunos se hallaban dentro de la zona de ataque, otros estaban fuera, y otros tanto se encontraban dentro por momentos y fuera en un tiempo posterior.

Dentro de las causas de las dificultades de coordinación los autores destacan que la tecnología juega un factor importante. De esta manera, afirman que el GPS tiene muchas limitaciones como tecnología de geolocalización con una variedad de tipos de errores como drift, jitter, lag y no disponibilidad. Asimismo, destacan también que la latencia del sistema es un factor clave para las dificultades de los jugadores, dado que hay retrasos de segundos entre que el dispositivo envía la posición actualizada del jugador, y éste recibe la respuesta del gameserver.

Algunas propuestas de los autores en [Benford et al., 2005] para la resolución de estos problemas son:

\* *Especificar el tamaño de la zona*: Una zona más grande puede alojar más jugadores facilitándoles su encuentro y reduciendo la imprecisión del GPS (ya que hay menos posibilidades que un jugador entre y salga de la zona). En contraste, una zona pequeña, es más difícil de encontrar y caben pocos jugadores que pueden tener problemas de imprecisión del GPS lo cual puede provocar la falsa percepción de que una presa se encuentre en movimiento.

\* *El despliegue de los sensores*: La posición de los sensores GPS parece afectar la calidad de los datos GPS. Ubicar estos dispositivos en la espalda o en el frente del jugador puede causar problemas cuando los participantes adoptan formaciones circulares. Como solución, en [Benford et al., 2005] se plantea ubicar los sensores a la altura de la cabeza de los jugadores o integrarlos en los dispositivos móviles aunque esto provoque una baja en la performance. Cabe aclarar que este problema hoy en día se encuentra prácticamente solucionado (el paper tiene diez años a la fecha) dado que los sensores se encuentran integrados a los dispositivos móviles y la performance no se ve casi afectada.

\* *La interpretación de las posiciones sensadas*: Los autores proponen rediseñar la

manera en la cuál el software interpreta los datos obtenidos por los sensores GPS para decidir cuando un grupo de jugadores se encuentran juntos para realizar un ataque en un mismo espacio. Las dos opciones propuestas son:

- Zonas de dos niveles: Una zona se divide en dos niveles. La parte interna se llama '*zona de disparo*' y la externa '*zona de captura*'. Para iniciar un ataque un jugador tiene que estar en la primer zona y una vez iniciado éste, todos los jugadores que se encuentren en la zona de captura pueden unirse.
- Auras personales: Cada jugador tiene un aura personal que proyecta su presencia en el espacio circundante a él mismo. Cuando un jugador inicia un ataque en una ubicación, todos los jugadores presentes en su aura pueden unirse a este ataque.

Dentro de las conclusiones elaboradas por los autores, determinan que el tema más significativo de este estudio es la formación de grupos. Más específicamente, encuentran una considerable divergencia entre lo que los jugadores y el sistema subyacente interpretan de formar un grupo. Para el jugador, la formación de grupos es un proceso altamente dinámico e intercalado, ambos temporal y espacialmente. Por ejemplo, algunos jugadores pueden estar en plena acción, mientras otros se congregan, y otros pueden estar ubicando recursos por su cuenta, y sin embargo, estar todos ellos en un mismo espacio común. Por el contrario, la visión del sistema del concepto de agrupamiento, es mucho más rígida. El sistema interpreta múltiples fuentes de datos, impregnados de incertidumbre y latencia, y esto provoca decisiones concretas y discretas sobre cuándo un grupo se ha formado.

Para achicar esta brecha, los autores proponen en [Benford et al., 2005] un enfoque más flexible para determinar la pertenencia a un grupo en una aplicación colaborativa basada en el posicionamiento. La primera propuesta es la utilización de diferentes sistemas de sensado. Por ejemplo, utilizar GPS para actividades en áreas más “amplias” (como la recolección de recursos en el sistema de los leones) y descartarlo para otras tareas como la coordinación de grupos, donde se opera a una escala mucho menor, con jugadores agrupándose en unos pocos metros y por consiguiente tornándose la precisión algo indispensable para el buen funcionamiento.

Luego, también plantean si es más apropiado trabajar con posicionamiento

absoluto o relativo. La respuesta que dan a ello es que dependiendo la tarea, es conveniente uno u otro. Por ejemplo, determinan que estaría bien utilizar posicionamiento absoluto en la ubicación de recursos; y posicionamiento relativo para el agrupamiento y trabajo como un grupo dado que con éste los jugadores pueden comparar sus posiciones entre sí. En resumen, los autores están a favor de utilizar alguna tecnología para detectar proximidad, como Bluetooth y RFID para la congregación y actuación en conjunto; y GPS para tareas en áreas más amplias como la recolección de recursos.

### ***Coordinación Involucrada***

Los jugadores exploran el área de juego de manera libre sin seguir ninguna ruta lineal. En la medida que van recorriendo los diferentes espacios, van encontrándose con estas “zonas de ataque” en donde están ubicadas las presas objetivo. Luego, en cada una de estas zonas, un participante puede iniciar un ataque, unirse a uno ya comenzado por otro participante o simplemente ignorarlo y no hacer nada. Por esta razón, se considera a la conformación de grupos un proceso altamente dinámico, temporal y espacialmente. No hay restricciones en cuanto a los lugares que deben recorrerse previamente para la conformación de los grupos ni tiempos establecidos para hacerlo. Sin embargo, hay que tener en cuenta que la coordinación en este juego se da exclusivamente en estas zonas o áreas de ataque. Todos los participantes que desean atacar en conjunto a una presa deben estar dentro de la misma zona para poder llevar a cabo un ataque coordinado. Un grupo se considera coordinado para un ataque, teniendo en cuenta a todos los participantes que se encuentran dentro de la zona del ataque (y son detectados por el sistema) y pudieron unirse al mismo dentro de los diez segundos que tiene de duración. La mínima cantidad de personas para un ataque coordinado es de dos (una sola persona puede ejecutar un ataque pero no sería coordinado) y la máxima es de seis.

## **2.3 ContextsContacts [Oulasvirta et al., 2005]**

*ContextsContacts* es una aplicación móvil que reemplaza a la libreta de contactos de un teléfono móvil y cuyo comportamiento y look-and-field es similar.

La principal diferencia con respecto a la libreta de contactos tradicional radica en que *ContextsContacts* provee de información adicional sobre los contactos (Ej: información sobre la ubicación de un contacto o el tiempo que estuvo una persona en dicho lugar). En las Figura 6 y Figura 7 vemos una libreta de contactos tradicional y la libreta de

*ContextContacts*, que provee cierta información de un contacto (*Vilmar*).



Figura 6: Libreta de contacto tradicional [Oulasvirta et al., 2005]

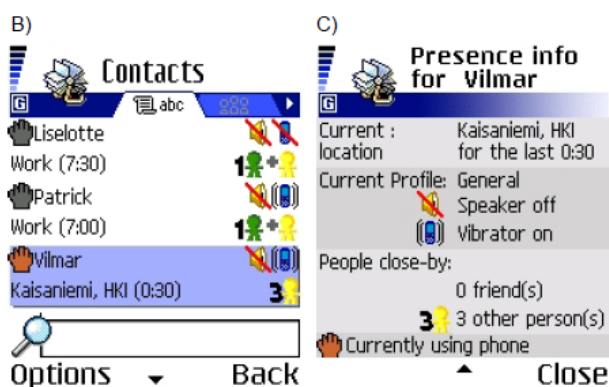


Figura 7: Libreta de ContextContacts con información ampliada de los contactos [Oulasvirta et al., 2005]

La reciprocidad de los datos compartidos está soportada por la aplicación de manera que si un contacto decide apagar la aplicación, dicha persona no recibirá información adicional alguna sobre sus contactos. En otras palabras, alguien no puede monitorear a alguna otra persona si no permite que sea monitoreado.

La utilidad de esta aplicación se evalúa en [Oulasvirta et al., 2005] para actividades grupales. Se toma un grupo de cinco estudiantes que concurren al mismo establecimiento y llevan a cabo un pequeño emprendimiento en conjunto. Utilizan *ContextsContacts* por más de un mes y toman conciencia de ciertas ventajas que provee la aplicación tales como:

- Inferir la posibilidad de interrupción: Se monitorea el “estado de interrupción” de los contactos, especialmente antes de llamarlo.
- Selección del canal de contacto: Coordinación para iniciar reuniones, por ejemplo en la elección del mejor canal de contacto.



- Monitoreo del progreso: Coordinación del progreso en tareas grupales, por ejemplo decidir no llamar para notificar un retraso percibiendo que los otros también están demorados.
- Monitoreo de la ubicación: Inferir la ubicación de un amigo, por ejemplo si está presente en el colegio para llevar a cabo alguna tarea en conjunto.

Siguiendo con el análisis de la aplicación, los autores proponen una extensión donde se divide a los contactos en grupos (Ej: familia, amigos, trabajo, etc.) y se configura qué información propia se comparte con cada grupo. También remarcan como una posible mejora la posibilidad de ver quiénes observaron la información de uno y qué información fueron capaces de consumir. Ambas opciones se visualizan en la Figura 8.

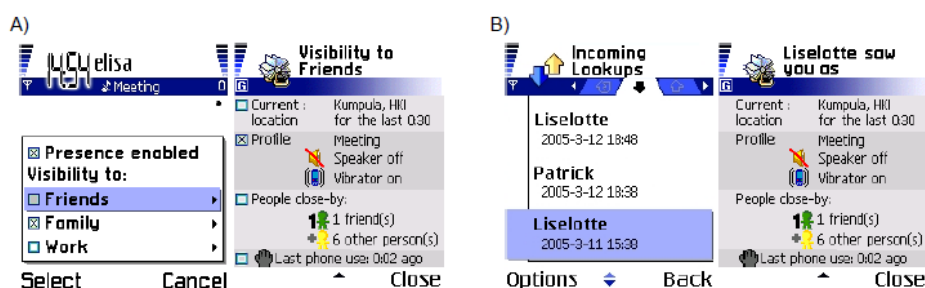


Figura 8: A) Separación de contactos por grupos B) Log sobre las visualizaciones al perfil de uno [Oulasvirta et al., 2005]

### ***Coordinación Involucrada***

De esta herramienta se pueden tomar ciertas ideas para la coordinación de un grupo y una posterior ejecución de tareas. Precisamente, el tener conocimiento sobre el estado en que se encuentra otra persona, su disponibilidad y su ubicación, entre otros datos, facilitan la tarea de coordinación. Además, el hecho de que uno no tenga que solicitar activamente la información (sino que la misma esté disponible para consumir) hace a la agilidad y simplificación de los mecanismos.

Algunos escenarios específicos de uso de *ContextsContacts* fueron en el marco de “dispositivos de conciencia inter-personal” que indicaban las identidades de las personas cercanas (en un rango menor a 100m) y se utilizaban para ciertos propósitos puntuales, apuntando a un cierto grupo y contexto. En un ambiente laboral, se utilizó la aplicación para saber qué empleados se encontraban presentes en un determinado lugar y tiempo; y en el contexto de un concierto musical, se lo usó para que las personas pudieran entablar un primer contacto entre sí. Además, en el contexto de conferencias se utilizó para

coordinar encuentros y para notificarse cuando ocurren ciertos eventos en determinadas personas (como podría ser notificarse cuando una persona está lista para dar su charla).

*ContextContacts* por sí mismo no es una aplicación que resuelva la coordinación de grupos, obligando a las personas a reunirse o a entrar en contacto entre sí. Por el contrario, la aplicación es una simple herramienta que provee cierta información de interés (sobre los contactos) que puede ser utilizada para la conformación de grupos y eventualmente la generación de alguna actividad en conjunto.

A manera de ejemplo, podría darse el caso de un grupo de profesores (todos con *ContextsContacts* instalado en su dispositivo móvil y teniéndose como contacto entre sí) donde cada uno se encuentra dictando una clase distinta y necesitan reunirse al finalizar dichas clases para discutir ciertos temas. Suponiendo que las diferentes clases pueden terminar en horarios distintos, se podría aprovechar la aplicación para conocer en qué momento todos se encuentran liberados y coordinar mediante llamadas telefónicas o mensajes de texto la hora y el lugar de reunión.

Si bien las ventajas de esta aplicación pueden no ser del todo claras hay que tener en cuenta que este paper tiene diez años de antigüedad por lo cual probablemente hoy en día existan otras herramientas más potentes y completas con características similares a *ContextsContacts*.

## **2.4 Coordinación de tareas en un Hospital [Ren et al., 2007]**

En [Ren et al., 2007] se hace un estudio de la coordinación necesaria para la ejecución exitosa de tareas en una sala de operaciones de un establecimiento sanitario. Dicha coordinación involucra múltiples grupos, con diferentes incentivos, culturas, y rutinas que pueden ser una dificultad a la hora de trabajar en conjunto.

Los autores definen como trayectoria a la secuencia de actividades y caminos a través de las cuales se movilizan las personas, los recursos y los grupos. Pacientes, doctores, enfermeras, y personal auxiliar, cada uno se mueve a través de una trayectoria entre tareas y tiempo. Los recursos como la sala de operaciones y el equipamiento también “se mueven” en trayectorias al ser utilizados caso tras caso para servir a diferentes pacientes.

La investigación se centra en la utilización de las salas de operaciones, qué grupos intervienen y cómo es la comunicación entre ellos. Una sala de operaciones (OR por sus siglas en inglés) se organiza principalmente por un programa (schedule) y para su

optimización se hacen reservas para su utilización.

Los grupos que intervienen en una operación normalmente son: ambulatorio, pre-operatorio, enfermeros, anestesistas, cirujanos, y el grupo que vela por los cuidados post anestesia. También son importantes para el trabajo, el personal de limpieza y aquellos que trasladan a los pacientes.

En el grupo ambulatorio ubican a aquellas personas que toman el primer contacto con el paciente; realizan las entrevistas pertinentes y realizan la admisión.

El grupo pre-operatorio está conformado por enfermeros que chequean los signos vitales de los pacientes y que estén todas las condiciones dadas para comenzar con la cirugía (que el paciente haya seguido las instrucciones pre-operatorias, que los estudios estén correctos, etc). Son responsables de dar aviso en la recepción de la sala de operaciones, al equipo de anestesistas y cirujanos que el paciente ya se encuentra en el hospital.

Los enfermeros reciben al paciente cuando arriba a la sala de operaciones, registra cierta información como tiempo, cargos, y asiste al equipo de cirujanos. Los técnicos proveen y mantienen estériles todo el equipamiento a utilizar durante la operación.

El equipo de anestesistas, visita al paciente en el área pre-operatoria, traslada al paciente hasta la sala de operaciones, aplica la anestesia, monitorea al paciente durante la operación, despierta al paciente y lo traslada hasta la sala post-operatoria.

El grupo post-anestesia monitorea el estado del paciente hasta que pueda trasladarse a una sala o volverse a su casa.

Cada grupo sigue diferentes trayectorias que se intersectan y se unen en diferentes momentos. Las principales trayectorias expuestas en este trabajo son: la del paciente, la de los empleados del hospital y la de los recursos. Estas trayectorias necesitan combinarse eficientemente en ciertos puntos para que la atención al paciente sea en término, segura y de calidad. La Figura 9 muestra una trayectoria tradicional para un paciente que necesita ser operado.

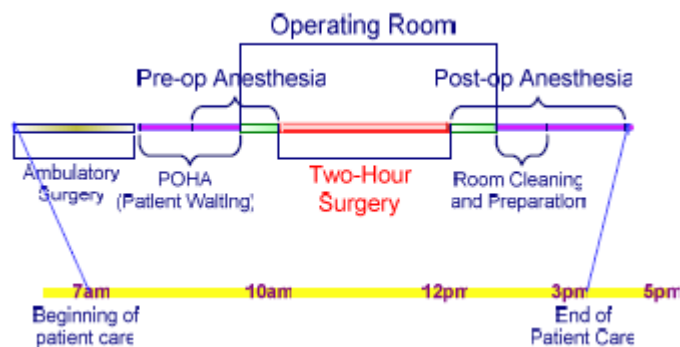


Figura 9: Trayectoria de un paciente [Ren et al., 2007]

El procedimiento común indica que el paciente ingresa al establecimiento, realiza las tareas pre-operatorias, se lo traslada a la sala donde se anestesia al paciente, los cirujanos operan, se detiene la anestesia y se vuelve a despertar al paciente para luego ser trasladado a la zona post-anestesia. Una vez que la sala de operaciones se libera, se higieniza y se reemplazan todos los materiales para dejarla lista para la próxima operación. En todo este proceso los autores identifican tres tipos de interdependencia: secuencial (no se puede realizar una tarea hasta que no estén dadas las condiciones), recíproca (grupos deben intercambiar información o completar acciones para que otros grupos hagan sus tareas) y dependencia en recursos compartidos o comportamientos. Cuando alguna de estas interdependencias falla, se produce una descoordinación.

Las descoordinaciones, a criterio de los autores, afectan el funcionamiento normal y las personas y los recursos se deben ajustar. La mayoría de ellos ocurre en los bordes (cuando un grupo “le encomienda” el paciente a otro). Entre las causas que podrían afectar una coordinación óptima se encuentran: casos de emergencia, cambios repentinos en el estado de un paciente, licencias o demoras del personal o de los pacientes, falta de información, mala comunicación, cambios de turnos en el personal, falta de experiencia en el personal, interrupciones constantes entre los equipos, etc.

Las consecuencias de esto pueden ser “positivas” si se toma en cuenta que los grupos tienen que ir adaptándose constantemente en tiempo real (a modo de adaptabilidad), pero por otro lado se generan demoras, se reducen los márgenes de ganancias, crecen los conflictos personales y las cargas de trabajo y stress se ven incrementadas.

Según [Ren et al., 2007] para mitigar estos riesgos, cada uno de los grupos debe conocer cuáles son los eventos críticos que pueden ocurrir y afectar a las diferentes

trayectorias. Los grupos tienen que tener una visión holística del proceso, para optimizar no solo el trabajo propio sino la coordinación en los bordes y poder evitar los cuellos de botella.

La coordinación de los diferentes equipos en una sala de operaciones es clave para el resultado exitoso de las operaciones y la optimización de los recursos. Los grupos que intervienen y las interacciones entre ellos son de lo más variadas y suelen sufrir imprevistos que atentan contra el funcionamiento normal de una sala de operaciones. Dentro de las situaciones no esperadas, hay responsabilidad en mayor o menor medida de todos los grupos intervinientes en el proceso. A manera de ejemplo, los *pacientes* suelen demorarse en la preparación de los papeles (formularios de consentimiento, historia clínica, etc.), no siguen la dieta pre-operatoria u olvidan tomar los medicamentos recetados oportunamente. Luego, también inciden cuando hay cambios repentinos en las condiciones del paciente mismo. Por el lado de los cirujanos, las principales demoras se dan por operaciones que se extienden más allá de lo previsto y por la práctica común de operar en varias salas en simultáneo, que si bien dan la posibilidad de llevar a cabo varias operaciones al mismo tiempo, agregan complejidad e introducen demoras indeseadas. Finalmente, los anestesiólogos y los enfermeros dificultan la coordinación por su falta de experiencia y, sobre todo, en los cambios de turno.

Para poder llevar una correcta coordinación del trabajo de todos los grupos, los autores consideran fundamental tener conciencia de las trayectorias de los grupos y recursos intervinientes. Por ejemplo, si se sabe que un paciente está demorado y va a tardar en entrar a la sala de operaciones, la enfermera a cargo de la sala puede programar en ella algún caso de emergencia.

Finalmente, los autores proponen una solución a través de la tecnología para dar soporte a este problema de la coordinación de grupos. La idea consiste en utilizar unos pizarrones interactivos llamados *eWhiteBoards* que optimizan la comunicación de las trayectorias. Los mismos se colocan en sectores claves, como en la sala pre-operatoria, en la recepción de la sala de operaciones, y la sala post-operatoria. Estos pizarrones se configuran para que muestren la información necesaria para cada grupo y se van actualizando automáticamente tomando la información de una base de datos. Por ejemplo, cuando una enfermera cambia el horario de una tarea, los demás grupos son notificados inmediatamente por medio de las pizarras. Luego, para actualizar esta base de datos, se utilizan sistemas sensibles al contexto como camas sensibles al contexto, brazaletes, dispositivos móviles que brinden información sobre la ubicación del personal, sensores de

audio y video para detectar actividades en las salas de operaciones.

Con toda esta información, se puede analizar, rediseñar y optimizar la coordinación para un óptimo funcionamiento de los grupos. Así también, gracias a la tecnología se logra optimizar la comunicación, haciéndola más fluida y mitigando el efecto negativo que tienen los imprevistos (de todas las trayectorias intervinientes) en los procesos. Además, se destaca la importancia de una cultura de colaboración para que los resultados sean los esperados. No solo la tecnología hará más eficiente este proceso, sino también las comunicaciones informales, las relaciones sociales, las actividades grupales ayudan a optimizar el proceso y la coordinación entre diferentes grupos.

### ***Coordinación Involucrada***

La coordinación, en este caso, se deja librada a la tecnología y la utilización de la herramienta *eWhiteBoards*. Los diferentes equipos no intervienen activamente (no cargan datos ellos mismos en el sistema) sino que son notificados por el sistema cuando cambios o modificaciones en las diferentes trayectorias surjan. La estrategia de coordinación para ejecutar acciones en conjunto, está íntimamente ligada a la información dinámica que provee el pizarrón en tiempo real. Esto es así, por el gran dinamismo que hay en las actividades que involucran a una sala de operaciones.

A manera de ejemplo, a medida que van surgiendo imprevistos y cambios de horarios repentinos, cada uno de los diferentes equipos se van enterando de ello en tiempo real pudiendo así optimizar el trabajo y tiempo propio. Si el cirujano encargado de operar se encuentra retrasado y producto de ello se pospone la operación por un par de horas, los anestesistas e instrumentistas al enterarse de este cambio al instante, pueden ser asignados en otras operaciones y/o tareas. De la misma manera, si la sala de operaciones que en un principio estaba programada para ser utilizada en dicho horario para esta operación en particular; se reprograma y puede ser utilizada para otra operación o atención de emergencia médica.

La coordinación de las trayectorias está dada por la disponibilidad de información en tiempo real, gracias a esta herramienta *eWhiteBoards*; haciendo posible la optimización de los recursos y logrando una optimización de los procesos en general.

## 2.5 Catch Bob! [Nova et al., 2009]

El juego *Catch Bob* permite realizar "actividades de colaboración con posicionamiento", estas pueden ser actividades en campo (por ejemplo: ubicación de minas terrestres o de exploración arqueológica), como así también rescates de emergencia o ciertas tareas militares.

Cada uno de estas actividades de colaboración tiene un componente espacial (de posicionamiento de algo o alguien) y una parte social (coordinación o colaboración).

Estas actividades se caracterizan también por el hecho de que pueden ocurrir en situaciones en las cuales el audio no es posible, por ejemplo, las operaciones militares a veces tratan de no producir ruidos.

Hay que destacar que los autores en este juego trataron de investigar la influencia de *Mutual Location Awareness* (MLA, capacidad de conocer el paradero o posición de otros participantes) sobre la coordinación en entornos móviles, concluyendo en que conocer la ubicación de otros participantes podría influir en los procesos socio-cognitivos implicados en una coordinación.

En la prueba del juego a la mitad de los grupos se les proveyó una herramienta que proporciona MLA, de este modo la otra mitad de los grupos no contó con la posibilidad de conocer la ubicación de los participantes.

Participaron sesenta alumnos de una escuela con un rango de edad entre los 19-28 años, promedio 23,1. Estos alumnos hablaban el mismo idioma dentro del grupo (había participantes que hablaban distintos idiomas). Además, dentro de cada grupo, el reparto fue en general dos hombres y una mujer, o dos mujeres y un hombre.

Como los diferentes niveles de conocimiento entre los jugadores pueden afectar la representación que cada uno de ellos tiene sobre sus compañeros de equipo, se seleccionaron grupos que pertenecían a la misma clase dentro de la escuela, es decir que anteriormente se conocían. También se verificó que los jugadores estén familiarizados con el Campus (espacio físico real donde se desarrollaría el juego), estudiando allí durante al menos un año.

En cuanto a la consigna del juego, los grupos tienen que atrapar a *Bob* que es un personaje estático virtual. Para completar el juego se requiere que los jugadores rodeen a *Bob* con un triángulo virtual formado por la posición de cada participante. Este triángulo debe ser equilátero con una longitud de unos 20-35 metros. Cuando un jugador está cerca de *Bob* el triángulo aparece en su pantalla, y en ese momento los tres participantes deben

adaptarse a la dimensión esperada. Cabe destacar que *Bob* sólo aparece en la pantalla cuando forman la configuración del triángulo adecuada a su alrededor. Un detalle a mencionar es que el juego se lleva a cabo de un grupo a la vez.

En cuanto a la interface, se puede mencionar que el juego se ejecuta en una Tablet, empleándose una aplicación, que en el caso de los grupos que contaban con la herramienta MLA, muestra un mapa con los jugadores en diferentes colores dispersos en el mapa (tal como se puede apreciar en la Figura 10), y un botón para actualizar las posiciones en el mapa.

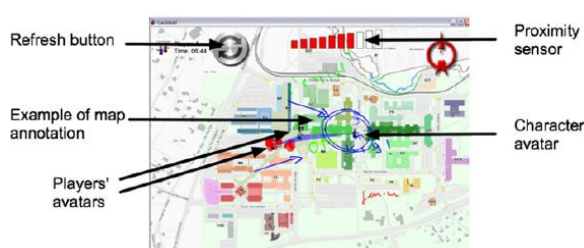


Figura 10: Interface del juego visto por un jugador [Nova et al., 2010]

Asimismo, mediante una comunicación sincrónica los jugadores pueden realizar anotaciones en el mapa (por medio de un lápiz) que se reproducen en las otras pantallas y se desvanecen hasta que se vuelven invisibles luego de cuatro minutos.

Para todos los grupos, tengan o no MLA, en la parte superior de la interfaz un sensor de proximidad individual indica si el jugador está lejos o cerca de *Bob* (objetivo).

Para el objetivo de la tesis, no se especifican detalles de las comparaciones obtenidas entre los grupos que usaron MLA y los que no. En general, se puede decir que la asistencia por sistema (MLA) facilita la conformación del triángulo ya que se puede ver donde deben ubicarse para encontrar a *Bob*.

### ***Coordinación Involucrada***

Los jugadores se distribuyen en el espacio físico, y usando el sensor de proximidad tratan de encontrar el área aproximada donde se encuentra *Bob*. En esta etapa el perímetro del triángulo aumenta.

Luego, cuando uno de los jugadores se encuentra en la proximidad de *Bob* invita a los otros indicándoles la ubicación a la que deben dirigirse (en el caso de ser necesario y posible utilizando annoting (anotaciones) en el mapa) para que se unan a construir el triángulo. En esta segunda fase, los jugadores se acercan unos a otros y el perímetro del



triángulo disminuye.

Por último, se requiere que los jugadores formen el triángulo alrededor de *Bob* en función de sus posiciones.

## **2.6 LaMOC (Location Aware Mobile Cooperation) [Gu et al., 2009]**

El sistema LaMOC, es un sistema cooperativo basado en el posicionamiento espacial móvil con decisiones apoyadas en grupos cooperativos y espaciales. Minimiza esfuerzos de usuario que está en movimiento para acceder a información, siendo conscientes de las condiciones ambientales (contexto) y las actividades de los otros usuarios. Los principales retos son la informática móvil y sensibilidad al contexto.

Veamos más detalles de este sistema, LaMOC es un sistema de computación móvil, en realidad un entorno híbrido: PC de escritorio, ordenadores portátiles, teléfonos inteligentes, PDAs, servidores, bases de datos activas en servidores. Los dispositivos móviles en LaMOC son principalmente teléfonos inteligentes. Un grupo de usuarios trabajan juntos, pero se distribuyen en diferentes lugares, para lograr un objetivo común, los usuarios en el mismo grupo comparten información con el navegador. Cada usuario navega en un mapa personalizado y pueden hacer clic en un icono en el mapa para marcar (hacer una llamada telefónica), hacer clic en un icono para SMS (enviar un mensaje corto), así como hacer clic en un icono para IM (mensaje instantáneo).

En cuanto a la sensibilidad al contexto, la ubicación hace que sea posible responder a las preguntas del usuario basadas en su ubicación real, como por ejemplo:

1. ¿Dónde estoy ahora?
2. ¿Quiénes están cerca de mí ahora?
3. ¿Hay algún amigo cerca de mí?

De esta manera, el usuario puede consultar en la base su ubicación, como encontrar un restaurante favorito cerca de sí mismo, la estación de metro más cercana, o la mejor ruta para llegar al lugar de destino. Y estos servicios se pueden extender a un grupo, como por ejemplo, encontrar un restaurante favorito por cinco miembros distribuidos de un grupo. Es decir, los usuarios de un mismo grupo cooperan entre sí para alcanzar un objetivo común.

En cuanto a la colaboración móvil se consideran desafíos tales como: el cambio dinámico de la estructura de la organización del grupo, el modelo de comunicación, así como el mecanismo de colaboración.

Uno de los mayores problemas de LaMOC es la obtención de un verdadero mapa en tiempo y forma, así como también afrontar el ancho de banda de comunicación móvil limitada y el tamaño limitado de la pantalla de la terminal.

### ***Coordinación Involucrada***

Debido a que LaMOC funciona en un entorno móvil, la movilidad del usuario es un gran reto. Por ejemplo, cinco miembros ubicados en un área restringida participan en una actividad de colaboración. Mientras están en movimiento, diez minutos más tarde uno de los miembros estará fuera de la zona, por lo que saldrán de la actividad de colaboración de forma automática. Al mismo tiempo, hay otros dos miembros que pertenecían al mismo grupo que tal vez quieran entrar en el área, entonces estas dos personas se unirán a la actividad de colaboración de forma automática. Los usuarios de un mismo grupo cooperan entre sí para alcanzar un objetivo común.

Otro ejemplo, podría ser, durante la visita a una exposición, uno de los miembros del grupo recomienda un pabellón de exposiciones de interés común a los integrantes de un mismo grupo.

En el dominio turístico, una actividad colaborativa podría ser que un grupo de turistas distribuidos en la ciudad tratan de encontrarse en un lugar de encuentro común.

## **2.7 Framework para interacción colaborativa [Lundgren et al., 2015]**

El análisis de este paper difiere al realizado en las secciones anteriores ya que los otros describían aplicaciones puntuales donde se implementaba algún tipo de coordinación de tareas posicionadas, o alguna herramienta que ayudaba a implementar la coordinación. Por su parte, en esta sección se presenta un framework conceptual que sirve como guía para el diseño de aplicaciones móviles para interacción colaborativa. El framework propone cuatro perspectivas diferentes: la situación **social** en la que tiene lugar, la **tecnología** utilizada, el **espacio físico** y los **elementos temporales** del diseño. Para dar soporte a la explicación del framework se introducen tres ejemplos diferentes de diseño, pertenecientes a dominios distintos.

- **Ruffor audioguide** es una aplicación móvil basada en audio diseñada específicamente para parejas que visitan un salón de esculturas. Ambas

personas exploran el lugar juntas, aunque en algún momento pueden elegir su propio camino dentro del predio. El diseño contempla momentos en los que una persona realiza la visita en solitario y otros, en donde ambos se juntan alrededor de ciertas esculturas para conversar sobre ellas.

- **Atomic Orchid** es un juego móvil de respuesta ante desastres. Los participantes toman diferentes roles como pueden ser: médico, soldado, bombero, transportista, etc. Cada rol distinto cuenta con habilidades diferentes y todos tienen como objetivo el rescate de objetivos virtuales (pueden ser recursos o personas). En ciertas ocasiones, se necesita de la cooperación entre personas con diferentes roles.
- **iBrainstorm** es una aplicación comercial que soporta trabajo creativo en grupo. Consta de un iPad y hasta cuatro iPhones que se conectan mediante Bluetooth y en donde los teléfonos escriben Post-Its virtuales y se los envían a la tablet. La aplicación de la tablet permite editar, colorear, agregar, trabajar sobre estos Poist-Its.

En la Figura 11 vemos las cuatro perspectivas del Framework con sus principales características.

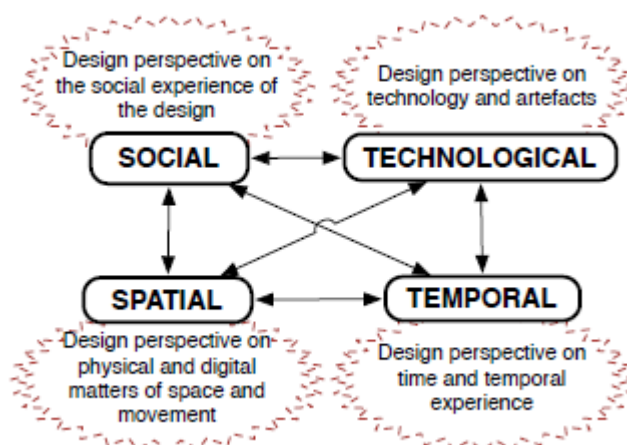


Figura 11: Las cuatro perspectivas propuestas por el Framework [Lundgren et al., 2015]

La **perspectiva social** describe el diseño “social” de la experiencia. Tiene en cuenta las implicancias sociales que surgen de la interacción, como por ejemplo los diferentes roles que pueden tomar los usuarios durante el desarrollo. Esta perspectiva cuenta con tres propiedades:

- **Focus:** Describe el foco de las interacciones entre los usuarios (Ej: colaborativo, comunicacional, competitivo).
- **Coordination of Action:** Esta propiedad está relacionada a cómo los usuarios ejecutan las acciones en conjunto. Pueden realizarlas de forma simultánea, o combinar los esfuerzos (complementando las características particulares de cada uno), o ambas cosas al mismo tiempo. Se relaciona con la perspectiva de espacio dado que afecta o es afectada por propiedades como proximidad y ubicación.
- **Framing:** Esta relacionado a la perspectiva de espacio dado que afecta la ubicación y el movimiento.

La **perspectiva tecnológica** es la más controlable por diseño. Se relaciona con la elección del hardware basándose en capacidades y limitaciones. En un sentido más amplio, la proporción de dispositivos por usuario son de importancia, así como también si los usuarios comparten dispositivos, etc. Esta perspectiva cuenta con cuatro propiedades:

- **Simetría de la información:** Se relaciona con la distribución de la información. Regula si todos los usuarios tienen acceso al mismo tipo de información o no. Es simétrico para el caso afirmativo y asimétrico cuando diferentes usuarios tienen acceso a diferente tipo de información.
- **Interaction Abilities:** Establece cómo los usuarios pueden actuar e interactuar en la aplicación; teniendo en cuenta capacidades especiales que el sistema provee a ciertos tipos de usuario. Es simétrico, si todos los usuarios tienen las mismas capacidades y asimétrico, si los usuarios tienen capacidades diferentes de acuerdo al rol que toman.
- **Distribución de la información:** Determina la manera en la cual la información se distribuye a los usuarios y espectadores.
- **Event Triggers:** Se relaciona a las acciones que los usuarios o el sistema deben realizar para que se dispare un evento que puede cambiar o generar un progreso en el sistema.

La **perspectiva espacial** tiene en cuenta la consideración de los espacios físico y virtual en la cual la experiencia se lleva a cabo. Tiene tres propiedades:

- **Proximidad:** Se relaciona con las diferentes maneras en la cual la proximidad se usa como un mecanismo, incluyendo las entidades y relaciones para las cuales la proximidad es un factor de importancia.
- **Ubicación:** Una o más ubicaciones, o lugares específicos que importan para la experiencia.
- **Movimiento:** Tienen en cuenta si los usuarios se mueven en el espacio como parte de la experiencia.

La **perspectiva temporal** tiene en cuenta el curso y el orden posible de las acciones y eventos, así como también el posible ritmo de la experiencia.

Las propiedades para esta perspectiva son:

- **Sincronización:** La manera en que las acciones, dentro de un mismo marco de tiempo, se sincronizan. Puede ser, que sea manejado por los usuarios mismos, por el sistema o una combinación de ambos.
- **Engagement:** Se relaciona a los patrones temporales de acción entre usuarios y la aplicación. Tiene en cuenta la intensidad y la frecuencia con la cual interactúan entre sí y con el sistema.
- **Pacing:** Describe la manera en la cual las acciones se distribuyen a lo largo del tiempo.

Las propiedades presentadas se pueden combinar siendo algunas más configurables que otras. En lo concerniente a la coordinación de entidades para realizar una tarea, encontramos muy relacionadas las propiedades de '*coordination of action*' de la perspectiva social, '*event triggers*' de la perspectiva tecnológica y 'proximidad' y 'ubicación' de la perspectiva espacial. Por ejemplo, la proximidad entre las personas puede ser controlable a través de la tecnología forzando a un grupo de usuarios a encontrarse en una ubicación específica para realizar una acción de coordinación cuyo resultado disparará un evento.

Este framework introduce muchos conceptos que ayudan a entender cuáles son las variables que se manejan a la hora de crear/configurar una aplicación móvil colaborativa y posicionada y de qué manera se relacionan.

### ***Análisis de los aspectos de coordinación propuestos por el framework***

La perspectiva que trata el tema de la coordinación de manera directa es la *Social*

mediante la propiedad *Coordination of Action*. Como se mencionó anteriormente, esta propiedad tiene en cuenta cómo los usuarios ejecutan las acciones en conjunto. Las mismas pueden ser simultáneas (en tiempo), pueden combinar los esfuerzos (complementándose) o ambas cosas al mismo tiempo.

Para el juego **Atomic Orchid**, se tiene una combinación de las tareas en tiempo y espacio, dado que el rescate de ciertos recursos, requiere la participación de diferentes personas con distintos roles en un mismo lugar. En contraste, **iBrainstorm** sólo espera que los usuarios combinen sus acciones (escribir post-its, moverlos, ordenarlos, etc.) sin necesidad de que estas tareas se hagan en conjunto al mismo tiempo.

La coordinación, a su vez, se encuentra relacionada con los conceptos de *proximidad* y *posicionamiento* de la perspectiva espacial, dado que éstos pueden ser determinantes de la coordinación de un grupo. La proximidad es un concepto que interesa al momento de coordinar un grupo (ej: que ciertos participantes estén en un radio no mayor a 5 metros) y el posicionamiento también lo es (ej: que los participantes estén en algún punto en particular). Así también, hay una estrecha relación de la coordinación con la *sincronización* de la perspectiva temporal, ya que ésta tiene en cuenta si los usuarios deben sincronizarse entre sí por sus propios medios, o si el sistema fuerza a la sincronización de alguna manera. Por ejemplo, en **Atomic Orchid**, la sincronización es controlada por el sistema dado que es éste quien exige a los participantes estar cerca entre sí y con el objetivo, para poder llevar a cabo el rescate.

En resumen, se podría decir que las propiedades de *Focus*, *Coordination of Action*, *Interaction Abilities*, *Event Triggers*, *Proximidad*, *Posicionamiento* y *Sincronización*; todas están relacionadas con aunar esfuerzos para ejecutar tareas en conjunto, en un mismo espacio de tiempo y/o lugar. La coordinación, a su vez, dispara otras acciones y tienen como objetivo resolver un tema o problema.

## 2.8. Resumen de aspectos de coordinación

Entre los papers analizados encontramos variadas formas de coordinación y diferentes formas de aplicar cada una de ellas. En esta sección realizaremos una comparación de las diferentes formas introducidas en las secciones anteriores, destacando similitudes y diferencias. Para ello, encararemos el análisis desde dos perspectivas distintas. En la primera, la comparación se lleva a cabo teniendo en cuenta los siguientes ítems.

- Cómo se conforman los grupos
- Qué tipo de recorrido realizan
- Cómo se recibe el próximo lugar al cual deben dirigirse
- Tareas que se realizan de manera coordinada

Y luego se analizarán acorde a considerar algunos aspectos presentados en el framework de la Sección 2.7.

En la Tabla 1 se puede apreciar el primer análisis realizado, se puede observar como cada ítem descripto es explicado para cada aplicación analizada en las Secciones 2.1 a 2.6.

**Tabla 1: Descripción de cada ítem mencionado anteriormente para cada una de las aplicaciones analizadas en las Secciones 2.1 a 2.6**

|  | <b>GRUPOS</b>  | <b>RECORRIDO DE CADA GRUPO</b>  | <b>PROXIMA POSICION</b>  | <b>TAREA COORDINADA</b>  |
|--|--|---|--|--|
| Who killed Hanne Holmgaard?<br>(Sección 2.1)           | Dos participantes trabajan en conjunto para resolver el misterio.                          | Desplazamiento físico a través de la ciudad. El recorrido es lineal y secuencial, y es el mismo para ambos participantes  | Mapa donde están marcados los puntos a donde deben dirigirse los participantes.  | La coordinación se da cuando ambos participantes consiguen encontrarse en los puntos de encuentro.   |
| Life on the Edge<br>(Sección 2.2)                      | Grupo de seis participantes juegan.  | Desplazamiento a través del espacio físico, no tiene recorrido lineal ni secuencial.  | Cada participante recorre libremente en busca de su presa  | La coordinación se da cuando dos o más jugadores se reúnen dentro de una misma zona de ataque, rodeando a una presa, y dentro de una ventana de tiempo de diez segundos. |
| ContextsContacts<br>(Sección 2.3)                      | Se divide a los contactos de un teléfono en grupos.  | ---   | Coordinación para iniciar reuniones, por ejemplo en la elección del mejor canal de contacto  | Coordinación de un grupo y una posterior ejecución de tareas según ciertas condiciones disponibilidad y su ubicación.  |
| Coordinación de tareas en un Hospital<br>(Sección 2.4) | Pacientes, doctores, enfermeras, y personal auxiliar.                                      | Desplazamiento a través del espacio físico, no tiene recorrido lineal ni secuencial pero el concepto de <i>trayectorias</i> es similar al de recorrido lineal.  | Según la trayectoria cambia el próximo punto a recorrer.<br>Se usa <i>eWhiteBoards</i> para analizar, rediseñar y optimizar la coordinación.                                       | Las trayectorias necesitan combinarse eficientemente en ciertos puntos para que la atención al paciente sea en término, segura y de calidad                              |
| Catch Bob<br>(Sección 2.5)                             | El juego se lleva a cabo de un grupo a la vez. Cada grupo está compuesto de tres personas. | Los jugadores se distribuyen en el espacio físico, y tratan de encontrar el área aproximada donde se encuentra <i>Bob</i> , por lo tanto no se considera un juego lineal o secuencial en cuanto a los grupos. | Uno de los jugadores se encuentra en la proximidad del objetivo e invita a los otros indicándoles la ubicación a la que deben dirigirse para que se unan a construir el triángulo. | Encontrar en el espacio físico un objetivo estático y formar un triángulo de un determinado perímetro para encerrarlo.   |
| LAMOC<br>(Sección 2.6)                                 | Compuestos por participantes sin ninguna restricción.                                      | Los grupos de usuarios trabajan juntos, pero se distribuyen en diferentes lugares. No se considera un recorrido lineal o secuencial.  | Pueden reunirse en un momento determinado, para seleccionar un sitio óptimo de alistamiento, para seleccionar una ruta óptima etc.   | Los usuarios de un mismo grupo cooperan entre sí para alcanzar un objetivo común.  |

Ahora se analizarán las aplicaciones presentadas teniendo en cuenta algunas de las propiedades presentadas por el framework de la Sección 2.7, relacionadas a las



aplicaciones móviles interactivas y posicionadas.

En la Tabla 2 se detallan las propiedades que consideramos relevantes del framework presentado en la Sección 2.7, en dicha tabla se introducen los valores que puede tomar cada una de estas propiedades.

**Tabla 2: Selección de Propiedades a analizar**

|                               |   |
|-------------------------------|---|
| <i>Focus</i>                  | Collaboration / Communication / Competition / Combined      |
| <i>Coordination of Action</i> | Timing Actions / Combining Actions / Combined               |
| <i>Interaction Abilities</i>  | Symmetrical / Asymmetrical                                  |
| <i>Event Triggers</i>         | Information-based / Time-based / Proximity-based / Combined |
| <i>Proximidad</i>             | People / Devices / Objects / Locations / Combined           |
| <i>Posicionamiento</i>        | One or more / None  |
| <i>Sincronización</i>         | User-driven / System-driven / Combined                      |

A continuación, en la Tabla 3, se detallan valores toman cada una de las aplicaciones analizadas en las Secciones 2.1 a 2.6, teniendo en cuenta las propiedades presentadas en la Tabla 2.

En particular, la propiedad *Proximidad* (de la Tabla 2) puede tomar diferentes valores: People, Devices, Objects, Locations y Combined. Dado que estos valores se pueden interpretar de diferente manera, a continuación se detalla que se está considerando como proximidad en cada aplicación, para así comprender mejor los valores descriptos en la Tabla 3.

- La proximidad en **WHO KILLED HANNED HOLMGAARD?** la conforman PDAs (Esto se relaciona con los valor *Devices*). En los puntos clave del juego, ambos deben reunirse físicamente para discutir sobre lo aprendido. Una vez que ambos participantes se reúnen y discuten sobre la información recolectada, deben dirigirse hacia la próxima escena en un lugar diferente (Esto se relaciona con los valores *People* y *Locations*).
- La proximidad en **Life on the Edge** la conforman PDAs (Esto se relaciona con el valor *Devices*). Se espera que los jugadores trabajen juntos (Esto se relaciona con el valor *People*) y decidan que animales atacar (Esto se relaciona con el valor *Locations*).
- Respecto a **ContextsContact** no posee ningún tipo de proximidad.
- La proximidad en **Hospital** la conforman pacientes, doctores, enfermeras, y personal auxiliar (Esto se relaciona con el valor *People*), las distintas salas (Esto se relaciona con el valor *Locations*), los recursos de equipamiento a utilizar durante la

operación (Esto se relaciona con el valor *Objects*). Grupos deben intercambiar información o completar acciones para que otros grupos hagan sus tareas (Esto se relaciona con el valor *Combined*).

- La proximidad en **CATCH BOB** la conforman los usuarios (Esto se relaciona con el valor *People*), el posicionamiento para formar el triángulo (Esto se relaciona con el valor *Locations*), y LAPTOPS (Esto se relaciona con el valor *Devices*).
- La proximidad en **LAMOC** la conforman los usuarios (Esto se relaciona con el valor *People*) y los dispositivos móviles (Esto se relaciona con el valor *Devices*). Los grupos de usuarios trabajan juntos, pero se distribuyen en diferentes lugares, para lograr un objetivo común, como para también reunirse en un momento determinado (Esto se relaciona con los valores *Locations* y *Combined*)

**Tabla 3: Análisis de las propiedades presentadas en la Tabla 2 respecto a las aplicaciones presentadas en las Secciones 2.1 a 2.6**

|  | Focus                         | Coordination of Action | Interaction Abilities | Event Triggers    | Proximity                               | Positioning | Synchronization |
|--|-------------------------------|------------------------|-----------------------|-------------------|---|-------------|-----------------|
| WHO KILLED HANNED HOLMGAARD?<br>(Sección 2.1)          | Collaboration / Communication | Combined               | Symmetrical           | Proximity-based   | People / Devices / Locations            | One or more | Combined        |
| Life on the Edge<br>(Sección 2.2)                      | Combined                      | Combined               | Symmetrical           | Proximity-based   | People / Devices / Locations            | One or more | Combined        |
| ContextsContacts<br>(Sección 2.3)                      | Communication                 | --                     | Symmetrical           | Information-based | --                                      | None        | --              |
| Coordinación de tareas en un Hospital<br>(Sección 2.4) | Collaboration / Communication | Combined               | Asymmetrical          | Combined          | People / Objects / Locations / Combined | One or more | System-driven   |
| CATCH BOB<br>(Sección 2.5)                             | Combined                      | Combined               | Symmetrical           | Proximity-based   | People / Devices / Locations            | One or more | User-driven     |
| LAMOC<br>(Sección 2.6)                                 | Collaboration / Communication | Combined               | Symmetrical           | Combined          | People / Objects / Locations / Combined | One or more | Combined        |

## 3. Modelo Propuesto

En este capítulo se describe el modelo usado como base, se plantea la problemática que se quiere resolver para luego proponer una extensión/modificación del modelo base para resolver la misma.

### 3.1 Descripción del Modelo usado como base

En el modelo propuesto en [Apezteguía and Rapetti, 2014], el objetivo específico consistía en diseñar un juego colaborativo móvil. Es decir, la característica principal que se buscaba para el juego, era que pudiera ser resuelto en grupos que colaboran entre sí para llegar a un resultado final. Este modelo tiene sus cimientos sobre el aprendizaje colaborativo, como los autores describen en [Apezteguía and Rapetti, 2014], en donde los participantes colaboran mutuamente intercambiando información para obtener conocimiento y juntos trabajan para resolver una tarea, generando ideas y simplificando los problemas. A continuación se da una descripción del modelo que se usará como base en nuestro trabajo.

Como se mencionó anteriormente, la propuesta de la cual partimos para nuestro trabajo, propone un modelo general para juegos colaborativos móviles, que toma como relevantes aquellos aspectos relacionados por un lado, al posicionamiento de los usuarios y por otro, a la colaboración entre los mismos. La idea principal es proponer un modelo general para juegos colaborativos móviles [Apezteguía and Rapetti, 2014].

En este modelo, un **juego** se define como un conjunto de consignas que posteriormente serán asignadas a los grupos que participan del mismo. No necesariamente todas las consignas serán asignadas en el transcurso del juego, ya que esto depende de cómo se vaya desarrollando el mismo. El objetivo del juego, es que los grupos recorran ciertos puntos que se les son asignados y al mismo tiempo, dependiendo de la modalidad del juego, que resuelvan las consignas en base a los elementos que van descubriendo en los diferentes puntos, mediante la colaboración mutua de los subgrupos de un mismo grupo. Cuando finaliza el juego, se calcula el puntaje para cada grupo, de acuerdo a los puntos que visitaron y a las decisiones tomadas por cada subgrupo ante la presentación de un elemento (suman puntos si el elemento cumplía con la consigna y la

respuesta dada por el subgrupo fue afirmativa y no suman en caso contrario). El grupo que suma mayor cantidad de puntos resulta el ganador del juego.

Todo lugar físico destacable en el juego se representa como un **punto de interés** (en adelante POI), del cual se tiene (entre otros posibles datos) la posición real.

Cada **consigna** tiene un conjunto de piezas de interés, o piezas a recolectar asociado. Cada pieza posee cierta información (una descripción y un nombre distintivo), y tiene asociado un POI, el cual determina la posición donde se ubica dicha pieza. Estas piezas asociadas a las consignas se comparan con las decisiones de los subgrupos y la consigna que tienen que cumplir, de manera de poder determinar si las respuestas dadas fueron correctas o no.

En cuanto a la organización de las personas que intervienen en el juego, podemos decir que el mismo consta de varios **grupos**, que compiten entre sí, donde cada uno de ellos está conformado por uno o más subgrupos. Cada **subgrupo** tiene al menos un participante y tiene en cuenta la consigna asignada al grupo al que pertenece. Cada subgrupo, cuenta (físicamente) con un dispositivo móvil desde el cual interactúa, con el juego en cuestión. La consigna es única para cada grupo, y los subgrupos de un mismo grupo pueden colaborar entre sí para resolverla. Por esta razón, para poder apreciar las características de un juego colaborativo es necesario tener más de un subgrupo para un grupo.

Un juego tiene un **punto de encuentro inicial**, que tiene asociado el lugar físico (POI) en donde los grupos se encuentran para comenzar a jugar y un **punto de encuentro final**, en donde los grupos se reúnen para que sean calculados los resultados finales del juego y se da por concluido el mismo. La posición del punto de inicio y punto de fin pertenecen al mismo espacio de juego, y está determinada por el POI que tienen asociado.

El juego posee una **estrategia**, que puede consistir en mostrarle a cada subgrupo solamente los POIs que debe visitar (estrategia de Buscar Elementos) o puede consistir en mostrarle a cada subgrupo los POIs del juego que deben visitar y que al visitarlos determinen si cumplen o no con la consigna que tienen asignada (estrategia de Descubrir Elementos).

En la Figura 12, se muestra el modelo presentado en [Apezteguía and Rapetti, 2014], al cual de ahora en más denominaremos modelo base.



Resumiendo, en el momento en que un subgrupo encuentra una pieza, cuenta con dos opciones:

- a) Tomar una decisión final para dicha pieza en base a su propio criterio.
- b) Iniciar un proceso de consulta al resto de los subgrupos que componen su propio grupo, para así poder elaborar la respuesta final. Para ello, el subgrupo genera una decisión parcial, emitiendo una justificación y especificando, según el propio parecer, si la pieza cumple o no la consigna propuesta. Esta decisión parcial se envía al resto de los subgrupos (del propio grupo) para luego recibir una respuesta por parte de ellos, que ratifique o rectifique la decisión parcial propuesta. Con la opinión de los demás subgrupos se genera finalmente la decisión final para la pieza encontrada.

Cuando un subgrupo termina de visitar todas las piezas asignadas, se le presenta el punto de encuentro final en el mapa. Una vez que todos los grupos se reúnen en el lugar, se da por finalizado el juego y se presentan los resultados.

### **3.2 Explicación de la problemática a resolver**

A continuación procedemos a explicar las características del tipo de juego móvil basado en posicionamiento que se quiere modelar. Esta caracterización está basada en el análisis realizado en el Capítulo 2. Mientras se describe la problemática se irá comparando con el tipo de juego descripto en la Sección 3.1.

Una vez que se inicia el juego, aparecerá en la pantalla de los dispositivos móviles de cada subgrupo, el primer POI al que deben dirigirse y en el cual como en el problema original se tomará una decisión respecto al elemento presente en dicho POI para luego brindarle el siguiente POI al cuál deberán dirigirse. Es decir, la presentación de los POI a recorrer para cada grupo se hará de a uno en uno, de manera gradual; en contraste con el juego propuesto en [Apezteguía and Rapetti, 2014], donde todos los elementos (POI) a recorrer se visualizaban de una vez al arribar al punto inicial. Este cambio está dado, porque en nuestra adaptación buscamos contar con un recorrido lineal, secuencial y predefinido para todos los subgrupos. De esta manera, los subgrupos ya no podrán ir en busca de sus elementos de manera aleatoria (sin orden) sino que van a ir descubriendo la posición del próximo elemento en la medida que vayan recorriendo cada uno de los POI.

Una de las grandes diferencias con el juego que resuelve el modelo base, es la introducción en este trabajo de ciertos puntos, en donde los subgrupos deberán

coordinarse para poder realizar una tarea o dar una respuesta.

De esta manera, tendremos POI comunes los cuales estaban en el modelo usado como base (denominándolos a estos de ahora en más como un POI generales), y se agregarán en este trabajo otros que requerirán algún tipo de coordinación, denominándolos a estos últimos, POI de coordinación. Estos POI de coordinación, van a requerir alguna clase de coordinación para poder recibir en conjunto la consigna asociada al POI, y así poder continuar el juego. Ambos tipos de POI, podrán estar intercalados en el recorrido definido para los grupos. Es decir, el recorrido será secuencial, y podrá contener POI generales o de coordinación.

Para hacer posible la coordinación en los POI de coordinación, contaremos con una “*Estrategia de Coordinación*” que va a estar asociada al juego (y cuyo criterio de validación determinará cuando un grupo estará coordinado o no). Dicha estrategia, no podrá cambiar durante todo el desarrollo del juego y será aplicada por el mismo en cada POI de coordinación.

Para nuestro trabajo, propondremos tres clases diferentes de “*Estrategia de Coordinación*”, aunque por supuesto, la cantidad y variedad de estrategias no se limitan solo a éstas. En la primera, un grupo se considerará coordinado en un POI de coordinación, cuando todos los subgrupos del mismo grupo se encuentren presentes en dicho POI. En la segunda estrategia, la coordinación se considerará efectiva, cuando todos los subgrupos de un mismo grupo se encuentren dentro de un área circundante definida para el POI de coordinación; y finalmente, en la tercera estrategia se considerará coordinado a un grupo cuando un porcentaje de todos los subgrupos (de dicho grupo) estén presentes en el POI.

Cuando un subgrupo se encuentre en un POI general o de coordinación, dará una respuesta en base a la consigna asignada al grupo del cual forma parte. Una vez dada la respuesta, el subgrupo visualizará en su dispositivo la posición del próximo elemento.

Por otro lado, cuando un subgrupo llega a un POI de coordinación, deberá esperar a que el grupo del cual forma parte él mismo, esté coordinado para poder continuar. Es decir, recién una vez que el grupo cumpla con los criterios definidos para la estrategia de coordinación (definida para el juego en curso), el subgrupo podrá dar una respuesta para la consigna asignada y recibirá la información de cuál es el próximo POI en el recorrido (por parte del juego).

El resto de los aspectos a considerar para el desenvolvimiento del juego, se mantendrán de la misma manera que en el juego descrito en [Apezteguía and Rapetti, 2014]. Es decir, la organización de los grupos y subgrupos, el sensado de los mismos, el

cálculo de los puntajes para cada grupo en base a las respuestas dadas, la interacción entre subgrupos a la hora de brindar respuestas para las consignas.

Nuestro trabajo, agregará coordinación de grupos en ciertos POI definidos para el juego conservando todas las demás características del juego descrito en la Sección 3.1.

### 3.3 Descripción del Modelo Propuesto

Habiendo introducido en las secciones anteriores, el modelo base y explicado la problemática con la que nos encontramos, a continuación procederemos a explicar cada una de las modificaciones que sufre el modelo tomado como base hasta llegar a nuestro *Modelo Extendido* propuesto.

- *Clase Juego*

En el *Modelo Base*, la clase *Juego* tiene conocimiento de un *PuntoDeEncuentro* inicial, un *PuntoDeEncuentro* final, el conjunto de *PiezaAREcolectar* durante el desarrollo del juego, las *Consigna* de juego que luego les eran asignadas a los grupos y los *Grupo* participantes del juego. Todas estas relaciones se ven reflejadas en la Figura 13.

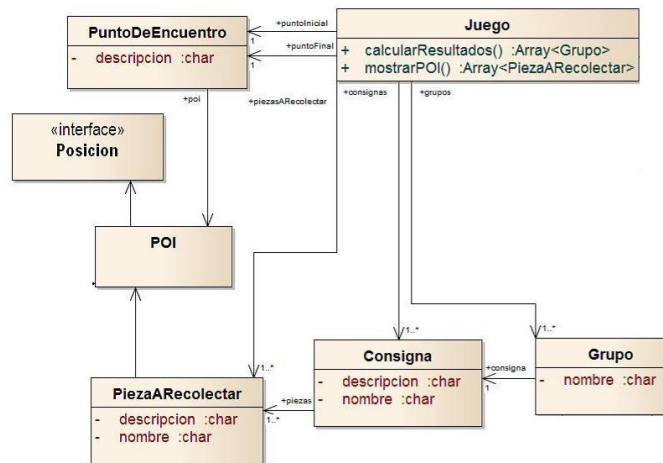


Figura 13 – Clase Juego en el Modelo base

En nuestro *Modelo Extendido*, la clase *Juego* pasará a conocer a todos los POI que participan del mismo, con un orden preestablecido y será la encargada de dar a cada subgrupo en cada POI, el siguiente POI en el recorrido. De esta manera, se preservará el camino lineal predefinido, como se mencionó en la Sección 3.2, que



tienen que seguir los subgrupos. Luego, de la misma manera que en el *Modelo Base*, y teniendo el conocimiento de todos los POI, será la clase *Juego* quien realice el cálculo final de los puntajes. Asimismo, el conocimiento general de *Juego* a los POI permitirá hacer consultas generales, como por ejemplo saber la posición de los POI del Juego.

Finalmente, se agregó un nuevo método `dameSiguientePOIPara(POI)` que devolverá el siguiente POI en el recorrido para el POI pasado por parámetro.

En la Figura 14, vemos cómo queda la clase *Juego* con nuestros cambios introducidos en el *Modelo Extendido*.

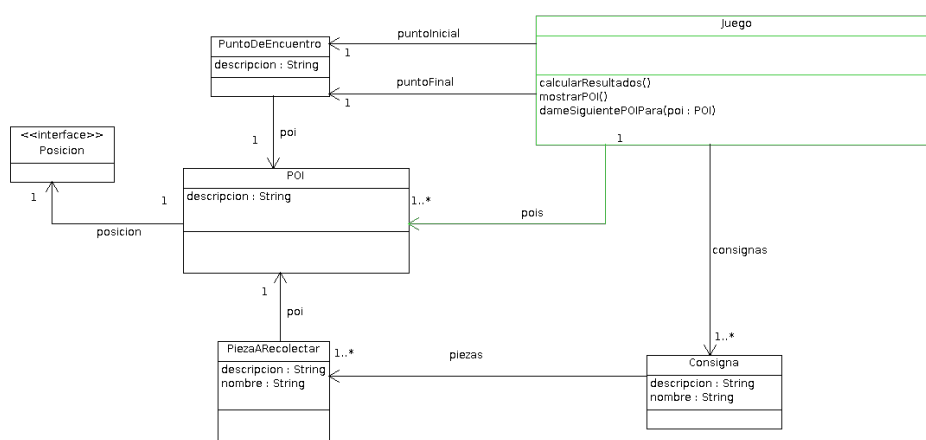


Figura 14 – Clase Juego y su relación con POI

- *Clase Consigna*

En el *Modelo Base*, un juego se define como un conjunto de consignas que posteriormente serán asignadas a los grupos que participan del mismo. La *Consigna* tiene un conjunto de *PiezaARecolectar* asociado. Los grupos recorren las piezas (ubicadas en ciertos POI) que se les asigna, y al mismo tiempo dependiendo de la modalidad del juego, resuelven las consignas en base a los elementos que van descubriendo en los diferentes puntos. Estas piezas asociadas a las consignas se comparan con las decisiones de los subgrupos y la consigna que tienen que cumplir, de manera de poder determinar si las respuestas dadas fueron correctas o no.

En la Figura 15 observamos las relaciones de la clase *Consigna* en el *Modelo base*.

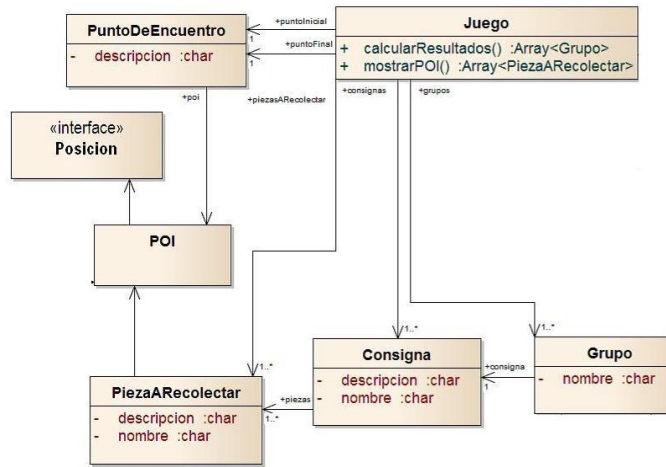


Figura 15 – Clase Consigna en el Modelo Base

En nuestro *Modelo Extendido*, tanto la clase *Juego* (tal cual se mencionó en el punto anterior) como la clase *Consigna* pasarán a conocer directamente a la clase *POI*, y es por medio de ésta que ambas clases conocerán a *PiezaARecolectar*. Esto se puede apreciar en la Figura 16.

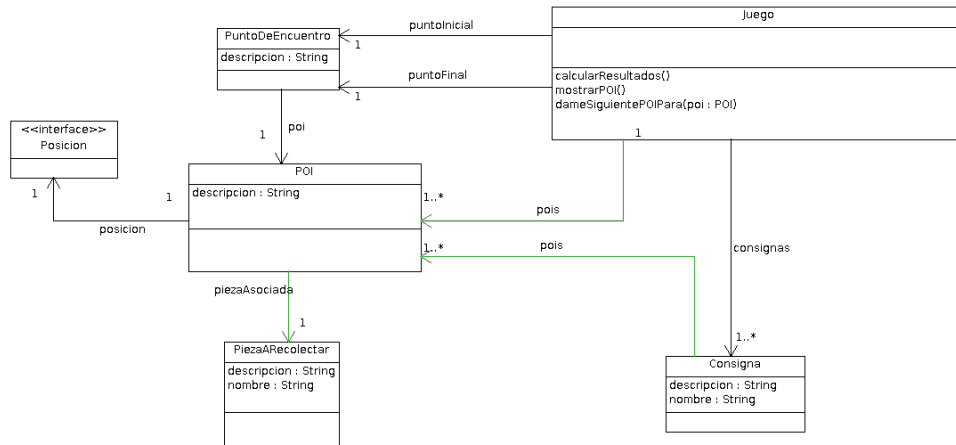


Figura 16 – clase Consigna también pasa a conocer a los POI

Como se ve en la Figura 16, para nuestra extensión del modelo, tanto la clase *Juego* como *Consigna* pasarán a conocer directamente a la clase *POI* por la importancia que tiene el recorrido lineal en nuestro problema y la relevancia que toma la clase *POI*

en este nuevo escenario. De esta manera, consideramos más apropiado que tanto la consigna como el juego conozcan directamente los puntos por donde tienen que ir pasando los jugadores, al mismo tiempo que se simplifica la solución.

Podríamos decir que el *Modelo Base* "se centraba" en las *PiezaARecolectar* (porque cada consigna tenía un conjunto de piezas de interés, o piezas a recolectar asociado), esto era así porque se recolectaban los elementos en un orden aleatorio, entonces una vez que encontraba una pieza, se tomaba la información de su ubicación. El desarrollo del juego se centraba en las piezas. En cambio, nuestra extensión cambia la perspectiva orientando el desarrollo del juego hacia los *POI*, dándole más relevancia a éstos últimos por sobre las piezas. Los subgrupos ya no "buscan" o "descubren" piezas, sino que recorren los *POI* y en cada uno de ellos responden una consigna en base a la pieza allí presente. Cambia la perspectiva del desarrollo del juego.

Por este motivo, *Consigna* pasa a conocer a los *POI*. Es decir, conoce los *POI* involucrados en la misma y, por ende, cuál es la pieza asociada a cada uno de ellos.

- *Clase PuntoDeEncuentro y POI*

En la Sección 3.1 para el *Modelo Base*, se definió un *POI* como la representación de todo lugar físico destacable en el juego, del cual se tiene (entre otros posibles datos) la posición real. Además, cada *Consigna* tiene un conjunto de piezas de interés, o piezas a recolectar asociado, que a su vez cada pieza posee cierta información (una descripción y un nombre distintivo), y tiene asociado un *POI*, el cual determina la posición donde se ubica dicha pieza. Estas piezas asociadas a las consignas se comparan con las decisiones de los subgrupos y la consigna que tienen que cumplir, de manera de poder determinar si las respuestas dadas fueron correctas o no.

Asimismo, el *Modelo Base* también define para la clase *Juego*, un *PuntoDeEncuentro* inicial, que tiene asociado el lugar físico (*POI*) en donde los grupos se encuentran para comenzar a jugar y un *PuntoDeEncuentro* final, en donde los grupos se reúnen para que sean calculados los resultados finales del juego y se dé por concluido el mismo. La posición de ambos *PuntoDeEncuentro* (inicial y final) pertenecen al mismo espacio de juego, y está determinada por el *POI* que tienen asociado.

Una vez iniciado el juego, cada subgrupo visualiza el punto de encuentro inicial (en la pantalla del dispositivo móvil asociado); lugar donde todos los subgrupos se reúnen para dar comienzo al juego. Cuando el juego detecta que todos los subgrupos

están en la ubicación previamente presentada, a cada subgrupo se le despliega en el mapa las piezas a visitar.

En la Figura 17, visualizamos lo anteriormente descrito, se puede apreciar los *PuntoDeEncuentro* inicial y final para *Juego*, en el *Modelo Base*.

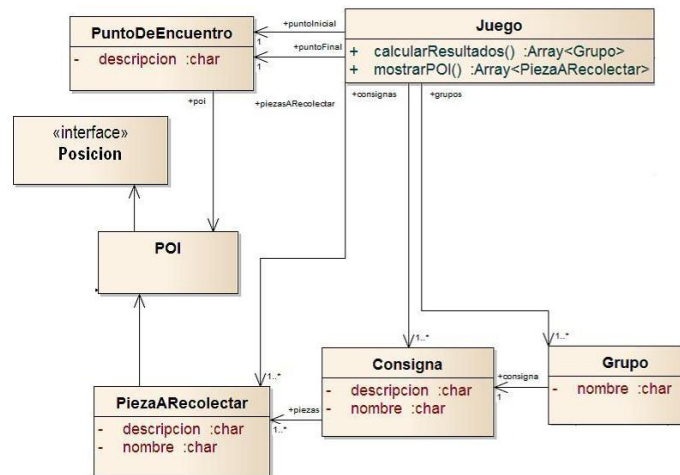


Figura 17 – Clase PuntoDeEncuentro en el Modelo Base

Por su parte, nuestro *Modelo Extendido* debe soportar la coordinación de subgrupos en ciertos puntos definidos del juego. Para hacerlo posible especializaremos la clase *POI*, de manera de tener *POI* generales y una nueva clase de *POI* que llamaremos *POIDeCoordinacion*, que exigirá precisamente alguna coordinación de subgrupos para poder darle continuidad al juego. La clase *POI* original, a su vez, también se verá modificada dado que pasará a conocer al siguiente *POI* en el recorrido lineal de los subgrupos y agregará un método `hacerJugar(Juego, Subgrupo)` que llevará a cabo las acciones necesarias para hacer jugar a un subgrupo en un *POI*. Cabe destacar que la clase *POIDeCoordinacion* sobrecargará este método con el fin de que además se pueda realizar la coordinación del grupo en dicho *POI*.

Por su parte, la clase *Juego* agregará los siguientes métodos que amplían su comportamiento:

- `procederEnPOIGeneral(POI, Subgrupo)`: implementa las acciones para hacer jugar a un subgrupo en un *POI*.
- `procederEnPOIDeCoordinacion(POIDeCoordinacion, Subgrupo)`: implementa las acciones para hacer jugar a un subgrupo en un *POIDeCoordinacion*, contemplando la coordinación del grupo previa a la

resolución de la consigna.

- `verificarDestino (Subgrupo, POI)`: verifica si el subgrupo se encuentra en la ubicación del *POI* al que debía dirigirse.
- `solicitarRespuesta (POI, Subgrupo)`: encapsula la solicitud de la respuesta a la consigna.
- `recibirRespuestaAConsigna (Subgrupo, POI, respuesta)`: encapsula la recepción de la respuesta a la consigna y la asignación del siguiente *POI*.
- `procesarRespuesta (Grupo, POI, respuesta)`: procesa la respuesta del grupo en *POI* para los cálculos finales.

De esta manera, tendremos que una vez iniciado el juego aparecerá en la pantalla de los dispositivos móviles de cada subgrupo, el primer *POI* al que deben dirigirse. Ya estarán definidas las consignas (y los *POI* que deben recorrer con un orden preestablecido) que deben cumplir cada uno de los grupos, y ya no será necesario, como en el *Modelo Base*, reunir a todos los grupos en un punto inicial. De la misma manera, un grupo finaliza su participación en el juego al responder la última consigna que se le presente y para ello no precisa reunir a todos los subgrupos integrantes en un punto final.

Por este motivo, y teniendo en cuenta las características de las clases *POI* y *POIDeCoordinacion*, ya no será necesaria la clase *PuntoDeEncuentro* ya que la misma se utilizaba tanto para representar ciertos puntos específicos de la aplicación (como eran el punto inicial y el final) y para coordinar a los subgrupos en dichos puntos. Con nuestra extensión del modelo, podremos representar mediante las clases *POI* y *POIDeCoordinacion*, los lugares físicos y aquellos puntos donde se realiza algún tipo de coordinación. Por esta razón, quitaremos la clase *PuntoDeEncuentro* de nuestro modelo y sus relaciones asociadas, quedándonos el modelo como se visualiza en la Figura 18.

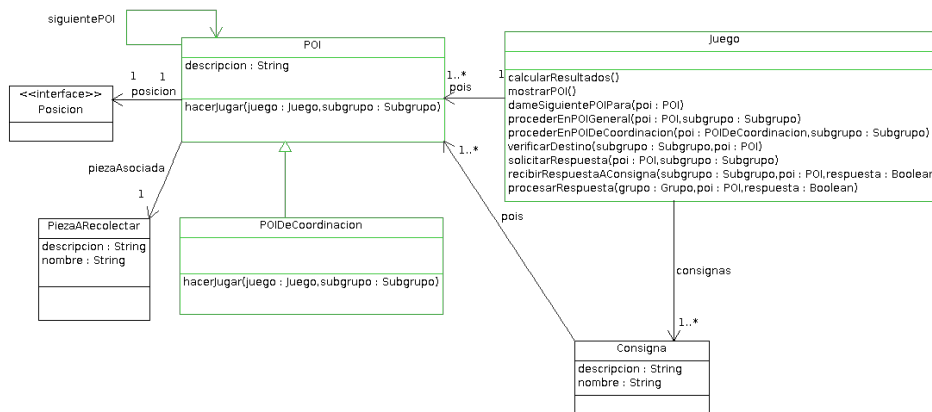


Figura18 – Clase POI se especializa en POIDeCoordinacion

- *Estrategias de Coordinación*

La clase *EstrategiaDeCoordinacion* no existe en el *Modelo Base* dado que no contempla la posibilidad de coordinar grupos en ciertos puntos del juego. Esta característica es introducida precisamente en nuestra extensión del modelo y para ello asociada a la clase *Juego* contaremos con una nueva clase llamada *EstrategiaDeCoordinacion*.

Esta estrategia de coordinación será aplicada por el juego en cada *POI* de coordinación y para ello agregará un método `aplicarEstrategiaDeCoordinacion(POIDeCoordinacion, Grupo)`. Podrán existir múltiples estrategias de coordinación y particularmente en este trabajo propondremos tres ejemplos concretos. Estas estrategias respetan el patrón de diseño *Strategy* [Gamma et al, 1994] y por tal motivo tendremos una clase padre abstracta *EstrategiaDeCoordinacion* y una subclase de ésta, por cada estrategia concreta que tengamos (donde cada una definirá su criterio de validación propio).

En la Figura 19, observamos cómo es la relación de *Juego* con *EstrategiaDeCoordinacion* y las subclases concretas que representan a cada una de las posibles estrategias a aplicar.

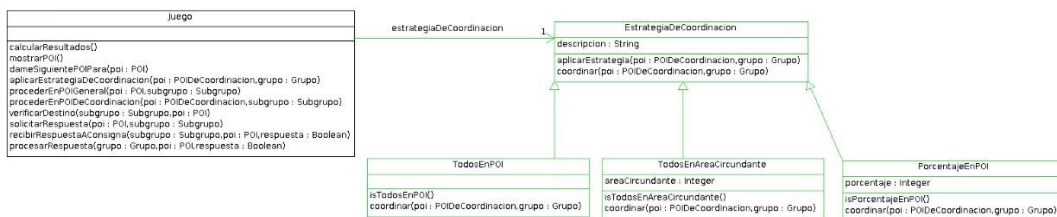


Figura 19 – La clase *Juego* tiene conocimiento de su *EstrategiaDeCoordinacion*

Veamos a continuación una descripción de cada una de las estrategias especificadas en la Figura 19:

- La estrategia *TodosEnPOI* considerará coordinado un grupo cuando la totalidad de los subgrupos de dicho grupo, se encuentre presente en el *POIDeCoordinacion* (es decir, en la posición del POI)
- La estrategia *TodosEnAreaCircundante* definirá un área circundante al *POIDeCoordinacion* (en donde se aplica la coordinación) y considerará coordinado un grupo, si la totalidad de los subgrupos de dicho grupo, se encuentra presente dentro del área definida para dicho *POIDeCoordinacion*.
- La estrategia *PorcentajeEnPOI* definirá un porcentaje y considerará coordinado un grupo, si la cantidad de subgrupos de dicho grupo representa (como mínimo) el porcentaje definido para dicho *POIDeCoordinacion*.

La *EstrategiaDeCoordinacion* establece el criterio de validación que determinará cuando un grupo estará coordinado o no. Un grupo deberá cumplir con los criterios definidos para la estrategia de coordinación para considerarse validado y poder continuar el juego.

Para nuestro trabajo consideramos que el *Juego* tendrá durante todo el desarrollo del mismo, la misma *EstrategiaDeCoordinacion* y no podrá variar. Por dicho motivo, consideramos lo más apropiado que sea *Juego* quien tenga conocimiento de la *EstrategiaDeCoordinacion*, por sobre otras opciones.

La clase *EstrategiaDeCoordinacion* contará con un método `aplicarEstrategia(PoiDeCoordinacion poi, Grupo grupo)` que permitirá la coordinación de los subgrupos pertenecientes al grupo pasado como parámetro en el *POIDeCoordinacion* dado. Además, contará con un método `coordinar(PoiDeCoordinacion poi, Grupo grupo)`, cuya implementación

dependerá de las particularidades de cada estrategia de coordinación; y desarrollará la implementación concreta de cada estrategia; por lo tanto, el método se sobrecargará en las subclases (en cada estrategia de coordinación concreta) agregando cada una la particularidad de su estrategia.

La Figura 20 visualiza cómo queda integrada la clase *EstrategiaDeCoordinacion* en nuestro Modelo Extendido propuesto.



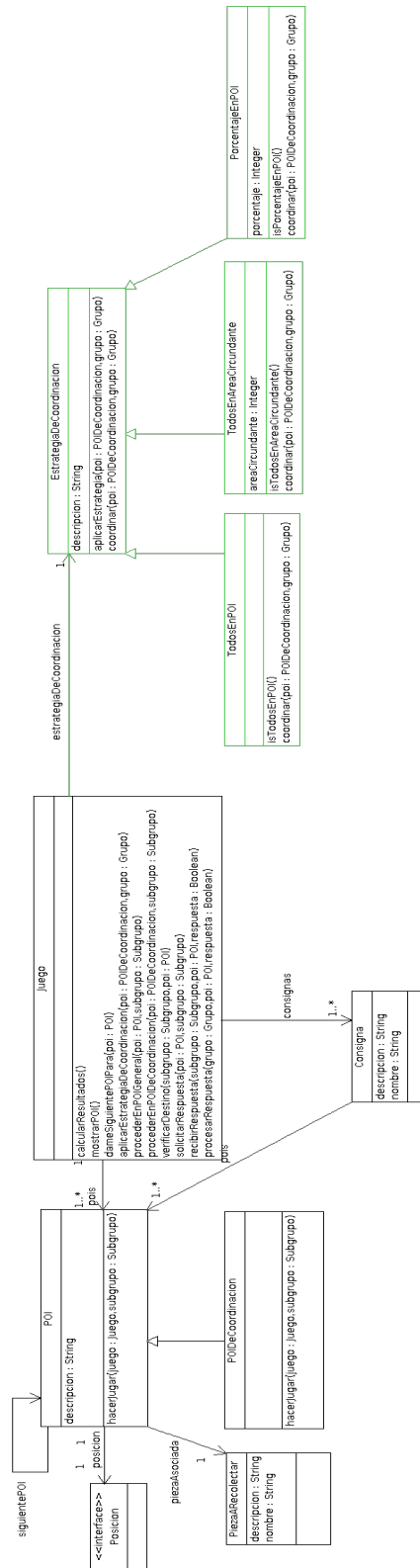


Figura 20 – Clase Juego conoce a su EstrategiaDeCoordinacion

- *Grupo*

A la clase *Grupo*, en nuestra extensión, se le agregarán los siguientes métodos:

- `asignarSiguientePunto()`, mediante el cual el *Grupo* hace la asignación a cada *Subgrupo* del siguiente *POI* en el recorrido y los notifica para que continúen su recorrido hacia el siguiente punto.
- `damePosicionSubgrupos()`, que informa la posición de cada uno de los *Subgrupo* pertenecientes a él.

En la Figura 21 vemos a la clase *Grupo* en el marco de nuestro *Modelo Extendido*.

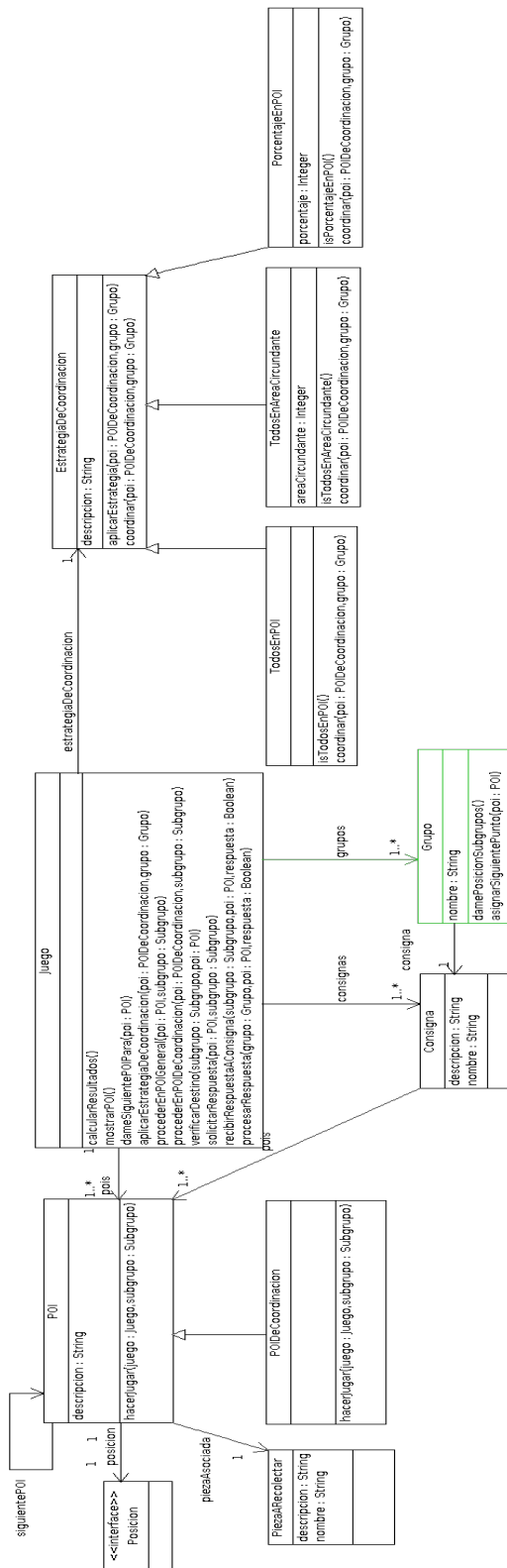


Figura 21 - Clase Grupo agrega ciertos métodos que colaboran con la coordinación

- *Subgrupo*

Por su parte, la clase *Subgrupo* se verá modificada ya que conocerá a su *destinoProximo*, que es el *POI* al cuál debe dirigirse en el momento actual para cumplir con el recorrido lineal predefinido. Asimismo, esta clase contará con un método para informar su posición actual, llamado `damePosicion()` y un método mediante el cual un *Subgrupo* es notificado para poder continuar su recorrido cuando se encuentra esperando al resto de los subgrupos de su mismo grupo en un *POIDeCoordinacion*, llamado `continuarRecorrido()`.

Finalmente, tendrá dos métodos para el manejo del siguiente *POI* a visitar en su recorrido: `getDestinoProximo()` y `setDestinoProximo(POI)`.

Y por último, tendrá un método `responderConsigna(POI)` mediante el cuál se le da aviso al subgrupo que está habilitado para responder la consigna para dicho *POI*.

La Figura 22 muestra a la clase *Subgrupo* en el marco de nuestro *Modelo Extendido*.

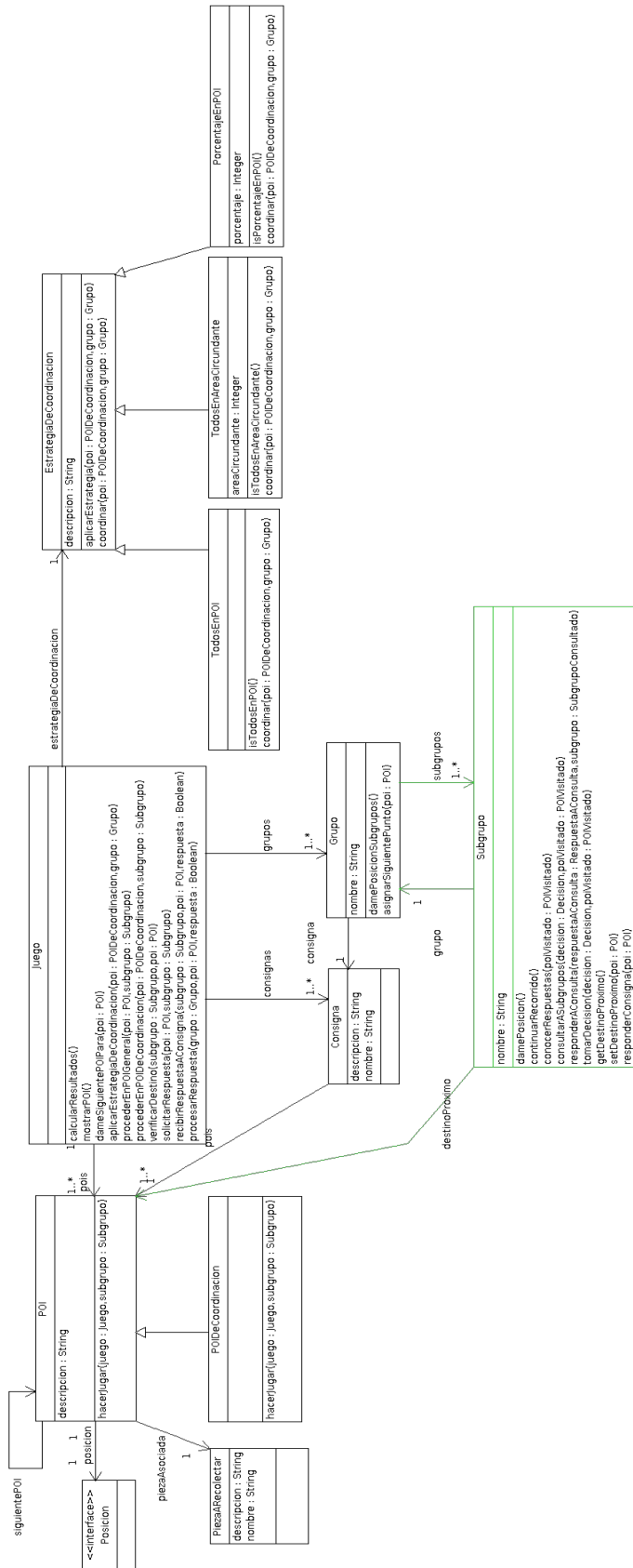


Figura 22 - Clase Subgrupo agrega ciertos métodos que colaboran con la coordinación

En la Figura 23 se presenta el *Modelo Extendido* propuesto que proponemos para nuestro trabajo. Se destacan en el mismo, aquellas clases que son agregadas o sufren algún tipo de modificación acorde a las características del problema a resolver.



### 3. 4 Diagramas de secuencia que explican la interacción

En esta sección presentaremos dos escenarios distintos con sus respectivos diagramas de secuencia. Cada uno de ellos representa una situación de juego particular, siendo:

- a) Primer punto es un *POI* general.
- b) Primer punto es un *POIDeCoordinacion* y se está utilizando la estrategia *TodosEnPOI*.

A continuación se presenta el flujo de mensajes generados para cada una de las situaciones descritas anteriormente.

- a) Primer punto es un *POI* general.

Este caso plantea el escenario donde el primer *POI* del recorrido lineal es un *POI* general. La instancia de *Juego* (de ahora en más *juego*) es quien detecta la presencia de una instancia de *Subgrupo* (llamémosla *subgrupo*) en la posición del primer *POI* del recorrido lineal (denominemos a este *POI* como *poi*).

La interacción comienza cuando *juego* invoca al método polimórfico `hacerJugar()` sobre el *poi*, enviándose por parámetro a sí mismo y al *subgrupo*. Por su parte el *poi*, continua el proceso invocando al método `procederEnPOIGeneral()` y se envía también a sí mismo y al *subgrupo*. A partir de allí, el *juego* comienza con la ejecución de una serie de métodos con el fin de que el *subgrupo* pueda desarrollar su juego en el *poi*. Para ello, como primera medida, verifica si el *poi* es el punto al que debe dirigirse el *subgrupo* en ese momento. Si la validación es positiva, el *juego* auto invoca al método `solicitarRespuesta()`; que simplemente solicita al *subgrupo* la respuesta a la consigna, invocando al método `responderConsigna()` en el *subgrupo*. Este proceso se refleja en la Figura 24. Por una cuestión de legibilidad en el diagrama se vuelve a repetir la instancia del *juego* en dicho diagrama. Al ser un *POI* general, cuando se detecta un *subgrupo* en el punto de interés se valida si es el destino donde debería llegar, y si es así, se le da para responder consigna.



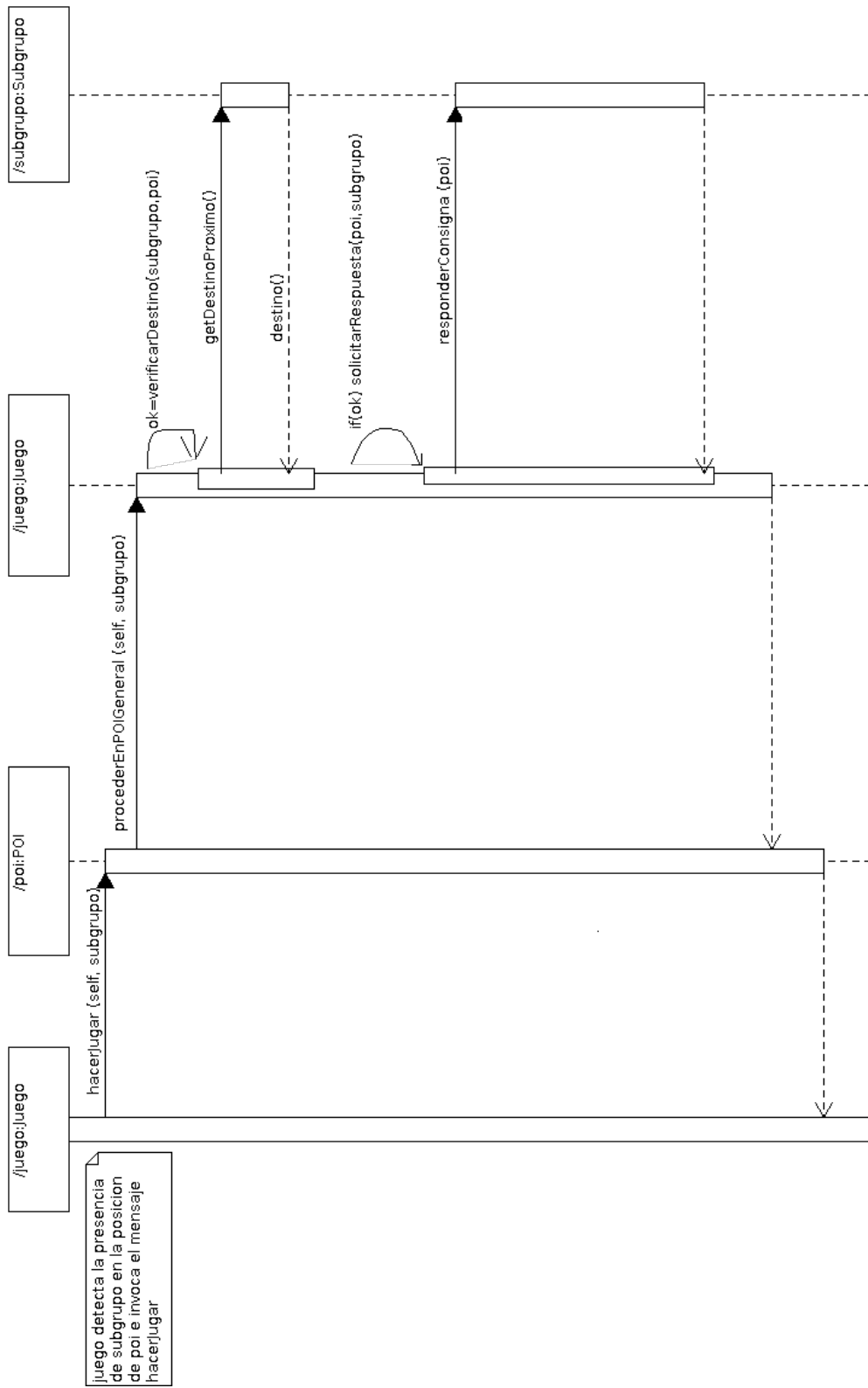


Figura 24 – Diagrama de Secuencia para el caso de un POI general como primer POI a visitar

A continuación, una vez que el *subgrupo* tiene la respuesta a la consigna, entrega dicha respuesta al *juego* invocando al método `recibirRespuestaAConsigna()`. Luego, el *juego* procesa la respuesta mediante el método `procesarRespuesta()` y una vez procesada ésta, le asigna a los subgrupos compañeros del *subgrupo* el siguiente POI al que deben dirigirse (por medio de la invocación al método `asignarSiguientePunto()` de la clase *Grupo*). Además, mediante la ejecución de este mismo método, los subgrupos reciben un mapa que sirve para indicar la posición del próximo punto y cómo llegar a él.

Este comportamiento se ve reflejado en la siguiente Figura 25.

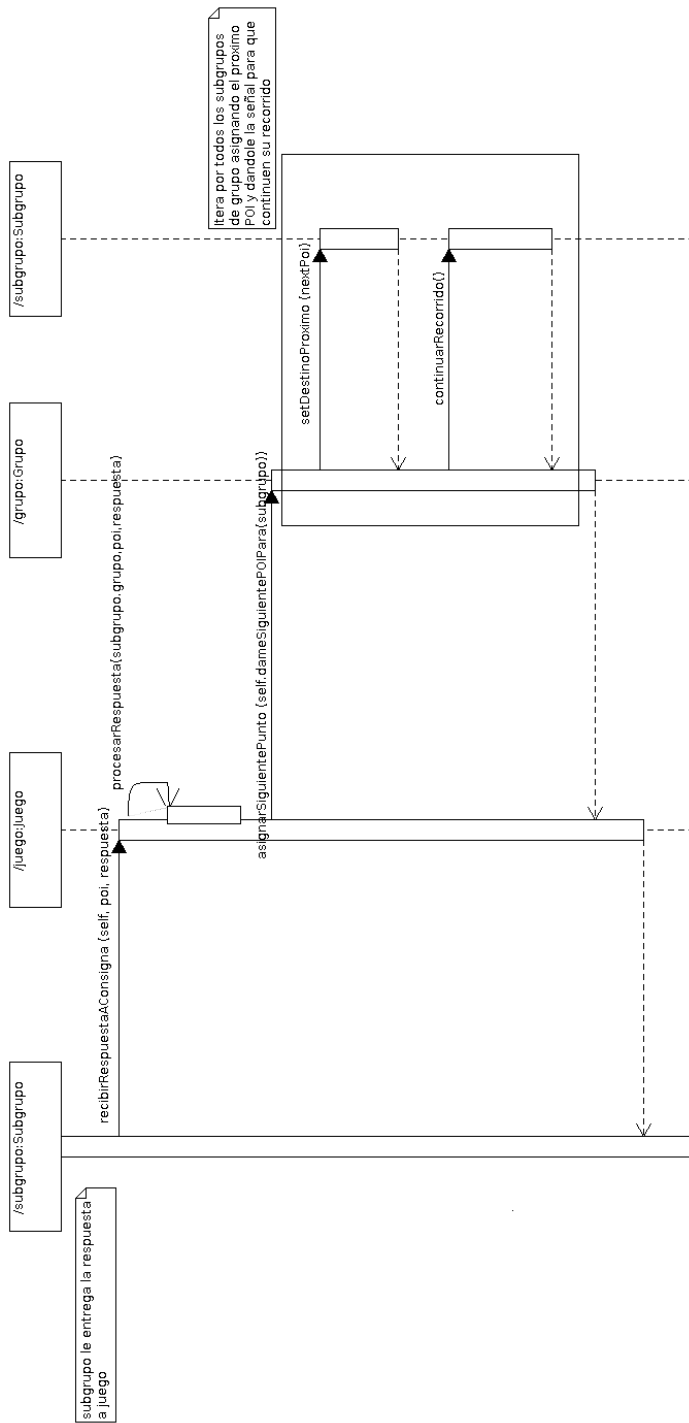


Figura 25 - Diagrama de secuencia que refleja la entrega de la respuesta del subgrupo y la asignación del próximo POI en el recorrido

Esta última secuencia es idéntica tanto para los casos a) y b) planteados al principio del capítulo. No importa desde que tipo de POI se reciba la respuesta a la consigna, la manera de procesarla y de asignar el siguiente POI es la misma.

b) Primer punto es un *POIDeCoordinacion* y se está utilizando la estrategia *TodosEnPOI*.

Este caso plantea el escenario donde el primer POI del recorrido lineal es un *POIDeCoordinacion* (llamémoslo *poiDeCoordinacion*) y la instancia de *EstrategiaDeCoordinacion* asociada a la instancia de *Juego* (de ahora en más *juego*) es una instancia de *TodosEnPoi* (la llamaremos *estrategia*).

Teniendo en cuenta lo antes descripto, el proceso comienza cuando el *juego* detecta la presencia de un *subgrupo* en la posición de *poiDeCoordinacion*. De esta manera, el *juego* invoca al método polimórfico `hacerJugar()` sobre *poiDeCoordinacion*, enviándose por parámetro a sí mismo y al *subgrupo*. Por su parte el *poiDeCoordinacion*, continua el proceso invocando al método `procederEnPOIDeCoordinacion()` y se envía también a sí mismo y al *subgrupo*. A partir de allí, el *juego* comienza con la ejecución de una serie de métodos con el fin de poder coordinar al grupo del que forma parte el *subgrupo* y para que una vez coordinado, el *subgrupo* pueda desarrollar su juego en el *poiDeCoordinacion*. Para tal fin, al igual que en el caso de un *POI* general, lo primero que hace el *juego* es verificar si el *poiDeCoordinacion* es el punto al que debe dirigirse el *subgrupo* en ese preciso momento. De ser así, el *juego* auto invoca al método `aplicarEstrategiaDeCoordinacion()` que recibe como parámetro al grupo del que forma parte el *subgrupo* (llamémoslo *grupo*) y al *poiDeCoordinacion*. Este método encapsula la coordinación del grupo. Para comenzar, el *juego* invoca al método `aplicarEstrategia()` de *TodosEnPoi*. Éste a su vez, no retornará el control al *juego* hasta tanto el grupo no esté coordinado para el *poiDeCoordinacion*. Luego, para llevar a cabo la coordinación propiamente dicha la clase *EstrategiaDeCoordinacion* cuenta con un método polimórfico `coordinar()` cuya implementación concreta en *estrategia* retorna si *poiDeCoordinacion* cumple con los criterios de coordinación para la estrategia en cuestión. Para este caso, `coordinar()` dirá si todos los subgrupos de *grupo* se encuentran en la misma posición que el *poiDeCoordinacion*. Esto se debe a que se está usando la estrategia *TodosEnPoi*.

Una vez que `aplicarEstrategia()` retorna el control al *juego*, se tiene la certeza que el *grupo* se encuentra coordinado y se auto invoca al método `solicitarRespuesta()`, que al igual que para el caso del *POI* general, solicita al

*subgrupo* la respuesta a la consigna. Esta secuencia se puede apreciar en la Figura 26. Por una cuestión de legibilidad en el diagrama se vuelve a repetir la instancia del *juego* en dicho diagrama.

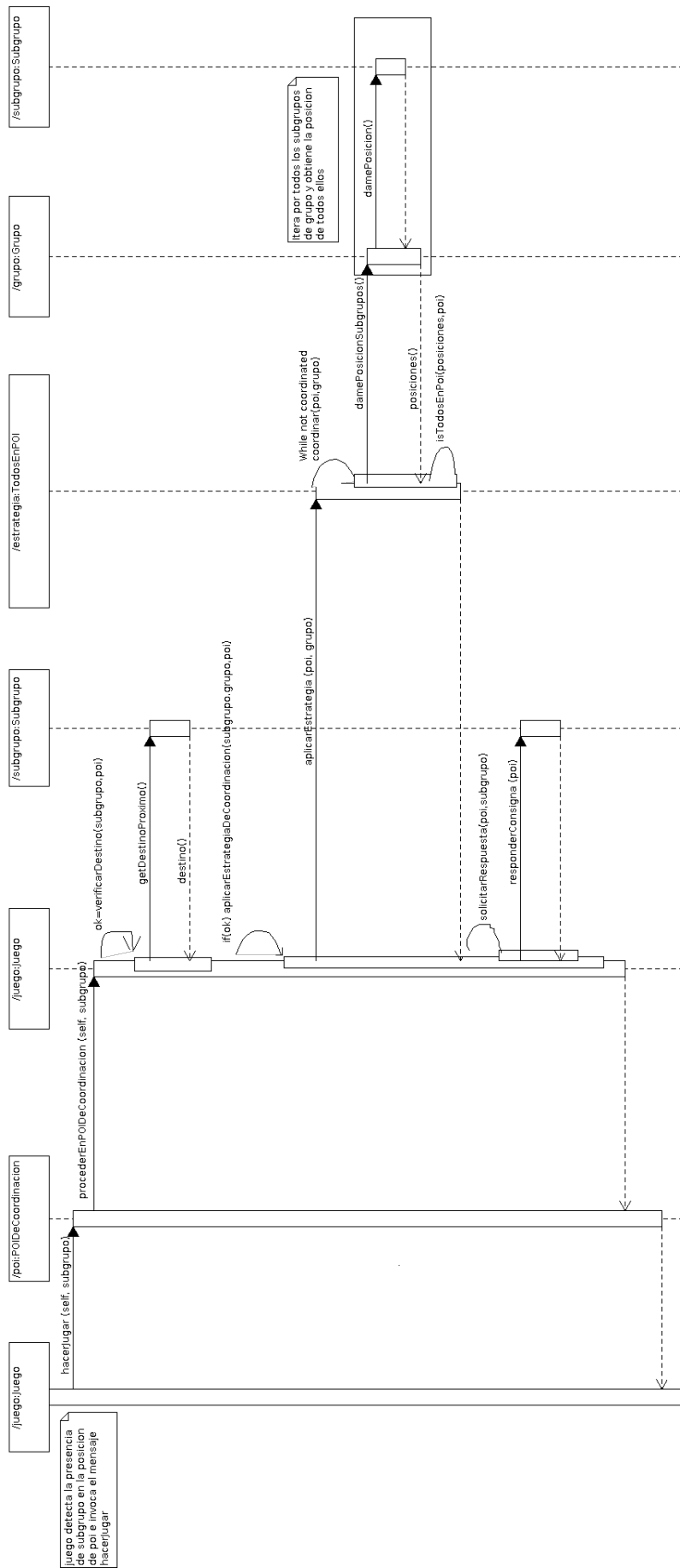


Figura 26 – Diagrama de Secuencia para el caso de un POIDeCoordinacion como primer POI

Finalmente, la secuencia para entregar la respuesta por parte del subgrupo y la asignación del siguiente POI para los subgrupos se lleva a cabo de la misma manera que para el caso a), siendo reflejada dicha secuencia en la Figura 25.

En el caso de tener otro tipo de estrategia de coordinación, la misma debe implementar el método `coordinar()` de acuerdo a los propios criterios de coordinación que dicha estrategia maneje. Luego, la interacción de los objetos no va a variar para el caso b), hasta el momento de la invocación al método `coordinar()`. En dicho punto, la estrategia de coordinación interactúa con las instancias de las otras clases de manera de poder determinar si el grupo está coordinado o no. Una vez determinado esto, la secuencia continua de igual manera con el *juego* solicitándole al *subgrupo* la respuesta a la consigna.

## 4. Prototipo Implementado

En este capítulo haremos una descripción de las funcionalidades del prototipo implementado. En este prototipo se usará el modelo definido en el Capítulo 3, como ya se menciono anteriormente, dicho modelo toma de base los conceptos definidos en [Apezteguía and Rapetti, 2014]. Acorde a esto, también se usan algunos conceptos del prototipo definido en [Apezteguía and Rapetti, 2014] como base de nuestro prototipo.

Cabe aclarar que nuestro prototipo conserva la misma arquitectura del prototipo usado como base [Apezteguía and Rapetti, 2014]. Es decir, se conserva la arquitectura Cliente/Servidor, que consiste en una aplicación nativa que se instala en los dispositivos móviles que lleva cada subgrupo y una aplicación web que se monta sobre un *Servidor Web* y donde también se encuentra alojada una base de datos relacional que persiste el modelo de datos a utilizar en la solución.

La aplicación web desarrollada por [Apezteguía and Rapetti, 2014] para administrar y configurar el juego, está desarrollada con el framework *Symfony*<sup>1</sup> en su versión 1.4. La experiencia previa acumulada con este framework, nos ha facilitado la extensión de la solución para resolver el problema propuesto en esta tesis.

Por el lado de la aplicación nativa desarrollada por [Apezteguía and Rapetti, 2014], está implementada con Android<sup>2</sup>. La modificación de la misma representó una dificultad al no tener demasiados conocimientos en dicha tecnología. Ello representó un desafío importante para nosotros que nos introdujo en el desarrollo de Android y en el de aplicaciones móviles en general. La curva de aprendizaje, si bien fue alta, fue suavizada parcialmente por los conocimientos previos en Java y otras soluciones similares como GWT (*Google Web Toolkit*).

El método de comunicación entre cliente y servidor tampoco difiere con respecto a lo presentado en [Apezteguía and Rapetti, 2014]. La misma se lleva a cabo mediante *Web Services* bajo el protocolo SOAP (*Simple Object Access Protocol*).

Los conceptos que en nuestro prototipo se tomaron como estaban definidos en [Apezteguía and Rapetti, 2014], es decir, no se modificaron, se describen en el Anexo A. En este capítulo nos centraremos en aquellos aspectos del prototipo que variaron respecto al tomado de base. Estas modificaciones guardan relación con el modelo presentado en el

---

<sup>1</sup> Página de Symfony: <https://symfony.com> (Ultimo acceso: 1/8/2016)

<sup>2</sup> Página de Android: <https://www.android.com> (Ultimo acceso: 1/8/2016)



Capítulo 3. En el Anexo B se detallan aspectos de código relacionados a las modificaciones realizadas y que guardan relación con las pantallas presentadas en este capítulo.

Cabe destacar que como parte del servidor (en el prototipo base) se encuentra el *Panel de Administración* el cual fue modificado para los fines de esta tesis. Pasemos a ahora a describir el módulo de *Administración Web* que fue implementado sobre la base del *Panel de Administración* del prototipo usado de base, y en el cual se agregan las funcionalidades de coordinación que modifican el manejo de los POI y las estrategias de coordinación.

Veamos las modificaciones realizadas al prototipo base, una vez logueado el usuario puede ver en el *Panel de Administración*, las opciones de POI (donde se listan los mismos) y *Estrategias de Coordinación* agregadas para nuestro prototipo. En la Figura 27 se muestra menú ampliado con estas incorporaciones.

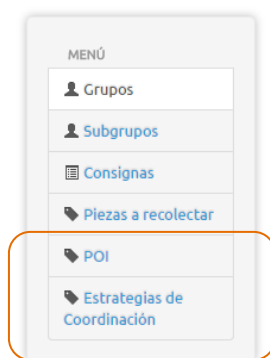


Figura 27 - Menú del juego

Veamos primero lo referido respecto a la opción POI del menú de opciones (mostrado en la Figura 27). La diferencia respecto al prototipo base es que un POI, en nuestro caso, pueden ser común o de coordinación (condición), y además ahora las consignas tienen asociado el POI que cumple con la misma (esta es la modificación del modelo presentada en la Figura 28<sup>3</sup>).

---

<sup>3</sup> Esta figura ya fue presentada anteriormente en el Capítulo 3 pero a fines de facilitar la comprensión del lector se vuelve a repetir en este capítulo.

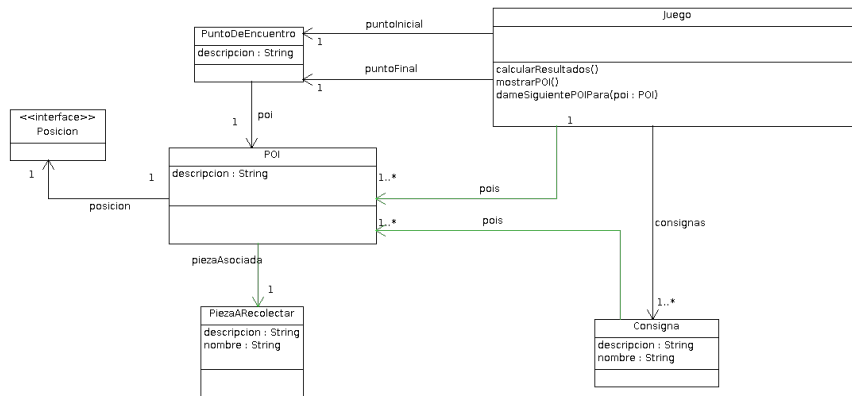


Figura 28 - Clase Consigna también pasa a conocer a los POI

Como dijimos anteriormente al presionar en la opción POI (Figura 27) aparecerá un listado de POI con los datos principales de cada uno y las respectivas acciones que se pueden realizar, y como se muestra en la Figura 29 se puede apreciar que lo nuevo que aparece de cada POI es la referencia (una descripción identificadora) y la condición (*De Coordinación* o *Común*).

| Referencia                  | Coordenada x | Coordenada y | Condición       | Acciones  |
|-----------------------------|--------------|--------------|-----------------|---|
| Aeropuerto La Plata         | -34.9618     | -57.8874     | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Estacion de Trenes          | -34.905      | -57.9492     | Comun           | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Estadio Unico               | -34.9168     | -57.9879     | Comun           | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Facultad de Informatica     | -34.9039     | -57.9379     | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Museo de Ciencias Naturales | -34.809      | -57.936      | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Parque Castelli             | -34.943      | -57.9529     | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Parque Saavedra             | -34.9308     | -57.9438     | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Parque San Martín           | -34.9333     | -57.9673     | Comun           | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Plaza 19 de Noviembre       | -34.9276     | -57.9738     | Comun           | <a href="#">Editar</a> <a href="#">Eliminar</a> |
| Plaza Alberti               | -34.923      | -57.9807     | De Coordinacion | <a href="#">Editar</a> <a href="#">Eliminar</a> |

Figura 29 - Lista de POI

Al seleccionar la creación de un nuevo POI se abrirá una pantalla como se muestra en la Figura 30. Allí, además de ingresar los datos utilizados en el prototipo base (latitud, longitud, etc.) tendremos que elegir también si es un POI *común* o *de coordinación* y una referencia (texto) para identificar al POI. Una vez completados estos ítems se presiona el

botón *Guardar* para que queden reflejados los cambios, de la misma manera que sucedía en el prototipo base.

The screenshot shows a web interface titled "Crear POI". Under the heading "Ubicación del punto de encuentro", there is a form with the following fields:

- Condición: A dropdown menu with "Comun" selected.
- Latitud: An empty text input field.
- Longitud: An empty text input field.
- Referencia: An empty text input field.
- Pieza: A dropdown menu with "Murcielago" selected.
- Consignas: A text input field containing the placeholder "Seleccione las consignas...".

Below the form is a Google Map of La Plata, Argentina, showing a grid of streets and landmarks like "Estadio Único Ciudad de La Plata". At the bottom of the interface are two buttons: "Guardar" (highlighted in blue) and "Cancelar".

Figura 30 - Crear nuevo POI

Para la edición de los POI, como se puede apreciar en la Figura 31 se tiene una pantalla similar a la mostrada anteriormente para la creación (Figura 30). Cabe destacar que desde esta pantalla se puede modificar si el POI es *común* o *de coordinación*.

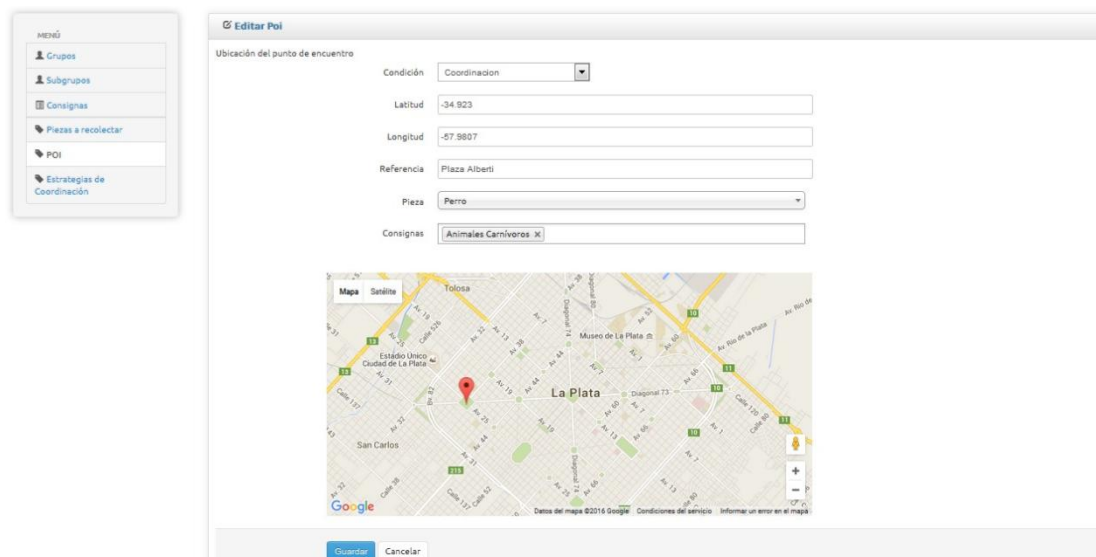


Figura 31 - Edición de un POI

De la misma manera que en la instanciación del prototipo base creamos dos grupos para nuestra instanciación, los grupos: *Rojo* (con *Subgrupo 1* y *Subgrupo 2*) y *Azul* (con *Subgrupo 3* y *Subgrupo 4*). La consigna de *Juego* para ambos grupos es diferente continuando con la instanciación del prototipo base. El *Grupo Rojo* debe identificar animales carnívoros mientras que *Grupo Azul* tiene que hacer lo mismo con animales herbívoros. Lo que varía en nuestro prototipo es que los POI se deben de recorrer en un orden secuencial, y en esto haremos hincapié a continuación.

A modo de ejemplo a continuación crearemos dos POI, a uno lo nombraremos como "*Plaza Italia*" y lo definiremos como *Común*, y otro lo denominaremos "*Plaza Moreno*" y lo definiremos como *de Coordinación*. Mantendremos las consignas y las piezas a recolectar usadas en el prototipo base detallado en el Anexo A.

En la Figura 32 se puede apreciar la creación del POI "*Plaza Italia*" donde del combo *Condición* elegimos "*Comun*", luego con el click derecho elegimos la ubicación del POI en el mapa (lo que hace que se seleccione latitud y longitud de dicho punto automáticamente), completamos el nombre, le asociamos la *Pieza* (*Serpiente*) y la *Consigna* (*Animales Carnívoros*).

Crear POI

Ubicación del punto de encuentro

Condición:

Latitud:

Longitud:

Referencia:

Pieza:

Consignas:

Guardar Cancelar

Figura 32 - Creación del POI "Plaza Italia"

Para crear el POI "Plaza Moreno" como de *Coordinación*, en el combo *Condición* elegimos "Coordinacion" como se puede apreciar en la Figura 33. Para completar la información de latitud y longitud asociada al POI, con el click derecho elegimos la ubicación en el mapa. Además completamos el nombre, le asociamos la *Pieza* (Mono) y la *Consigna* (Animales Carnívoros).

Crear POI

Ubicación del punto de encuentro

Condición:

Latitud:

Longitud:

Referencia:

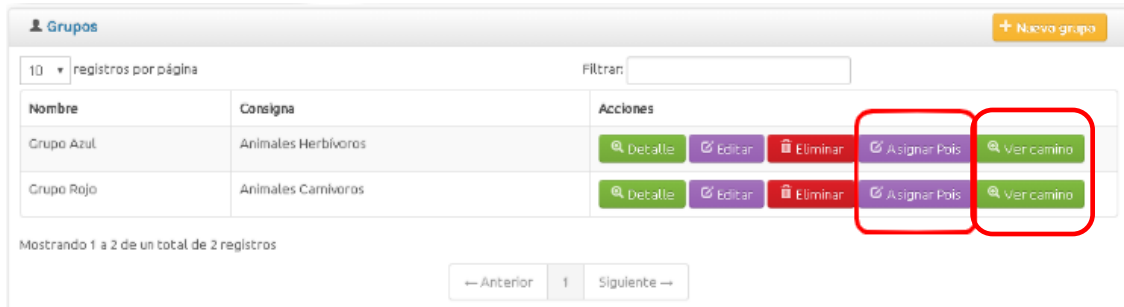
Pieza:

Consignas:

Guardar Cancelar

Figura 33- Creación del POI "Plaza Moreno"

Otra característica que varía respecto del prototipo base, es al elegir la opción *Grupos* del menú de opciones (mostrado en la Figura 27). Seleccionando dicha opción se muestra un listado de los grupos, tal como se aprecia en la Figura 34, en el cual agregamos los botones “Asignar Pois” y “Ver camino”.



| Nombre     | Consigna            | Acciones  |
|------------|---------------------|---|
| Grupo Azul | Animales Herbívoros | <a href="#">Detalle</a> <a href="#">editar</a> <a href="#">Eliminar</a> <a href="#">Asignar Pois</a> <a href="#">Ver camino</a> |
| Grupo Rojo | Animales Carnívoros | <a href="#">Detalle</a> <a href="#">editar</a> <a href="#">Eliminar</a> <a href="#">Asignar Pois</a> <a href="#">Ver camino</a> |

Mostrando 1 a 2 de un total de 2 registros

← Anterior 1 Siguiente →

**Figura 34 – Grupos y la opción Asignar Pois**

Al presionar sobre el botón “Asignar Pois” (de la Figura 34) tendremos la posibilidad de elegir los POI que serán parte del camino secuencial para el *Grupo*. Esta opción se utiliza para elaborar el camino secuencial que deben seguir todos los subgrupos de un grupo.

Para generar el camino secuencial, se deben agregar los POI que lo van a conformar, uno por uno. A manera de ejemplo, para agregar un POI al camino del Grupo *Rojo*, primero se debe elegir el POI que queremos asignar al recorrido mediante el combo denominado “POI”. Una vez seleccionado el POI, se nos muestra información detallada del POI elegido, como la ubicación (latitud/longitud), una descripción, un mapa de ubicación y la condición de dicho POI (si es *común* ó de *coordinación*). En la Figura 35 se puede apreciar que se eligió el POI “Plaza Italia”, y de esta elección resulta en la información mostrada en dicha figura.



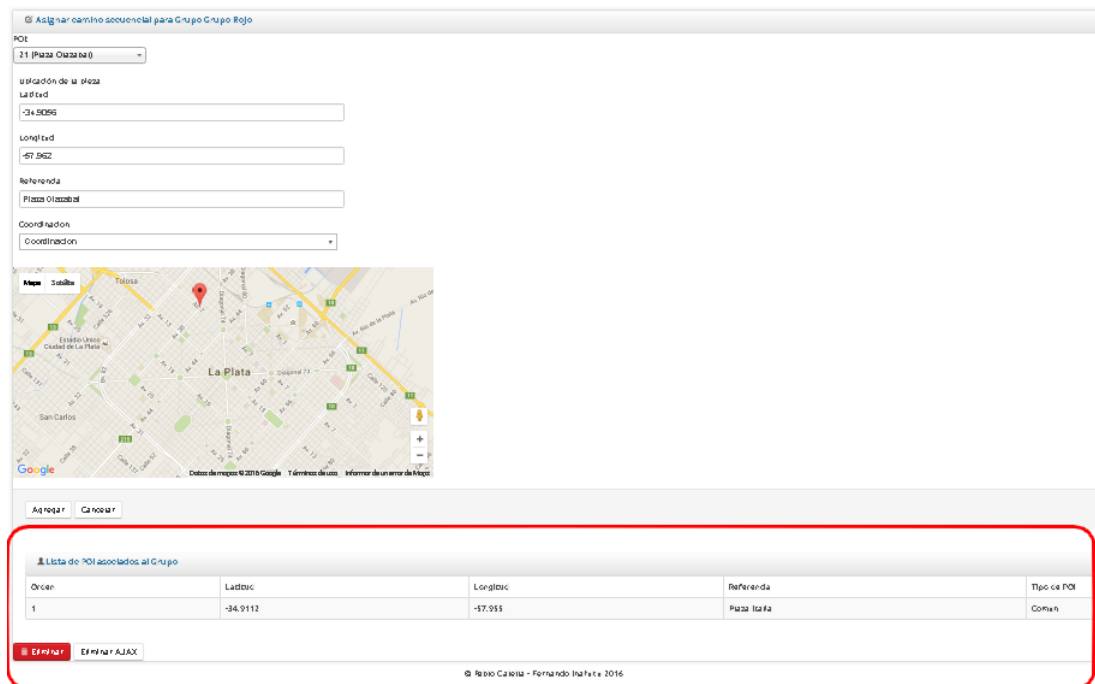


Figura 36 - Listado POI del Grupo Rojo

Asimismo, para los casos de error, tenemos un botón *Eliminar* que quita el último POI de la lista y nos asegura que siempre se preserve el orden del recorrido.

A continuación, a modo de ejemplo, se armarán caminos para cada uno de los grupos, el *Grupo Rojo* y *Grupo Azul*. Primero definiremos el camino secuencial del *Grupo Rojo* que se conforma de cuatro POI en el siguiente orden:

1. *Plaza Italia* (POI Común).
2. *Plaza 19 de Noviembre* (POI Común).
3. *Plaza Moreno* (POI de Coordinación).
4. *Plaza Matheu* (POI Común).

Como ya se mencionó anteriormente, cada POI se va agregando secuencialmente, uno por uno, siguiendo el orden mencionado. Cada uno de los POI, se elige de la misma manera tal como fue mostrado anteriormente (como se mostró en la Figura 35), y luego se presiona el botón *Agregar*. Así se van agregando uno a uno a la lista de POI del camino (como se mostró en la Figura 36).

Ya teniendo el primero POI "*Plaza Italia*" agregado al camino, pasemos a ilustrar de



manera simple y precisa como se agregan los siguientes tres POI restantes.

Pasemos a agregar el POI “Plaza 19 de Noviembre”, para lo cual debemos elegir ese POI en el combo denominado “POI”, y una vez seleccionado, se nos muestra información detallada del POI elegido, como la ubicación (latitud/longitud), una descripción, un mapa de ubicación y la condición de dicho POI (en este caso es *Común*). En la Figura 37 se puede apreciar que se eligió el POI “Plaza 19 de Noviembre”, y esta elección resulta en la información mostrada en dicha figura. Para que se realice la incorporación de este POI al camino hay que presionar el botón *Agregar*.

Asignar camino secuencial para Grupo Grupo Rojo

POI:  
29 (Plaza 19 de Noviembre)

Ubicación de la pieza

Latitud  
-34.9276

Longitud  
-57.9738

Referencia  
Plaza 19 de Noviembre

Coordinación  
Comun

Mapa Satélite

La Plata

Estadio Único Ciudad de La Plata

San Carlos

Tolosa

Av. 19

Av. 21

Av. 23

Av. 25

Av. 27

Av. 29

Av. 31

Av. 33

Av. 35

Av. 37

Av. 39

Av. 41

Av. 43

Av. 45

Av. 47

Av. 49

Av. 51

Av. 53

Av. 55

Av. 57

Av. 59

Av. 61

Av. 63

Av. 65

Av. 67

Av. 69

Av. 71

Av. 73

Av. 75

Av. 77

Av. 79

Av. 81

Av. 83

Av. 85

Av. 87

Av. 89

Av. 91

Av. 93

Av. 95

Av. 97

Av. 99

Av. 101

Av. 103

Av. 105

Av. 107

Av. 109

Av. 111

Av. 113

Av. 115

Av. 117

Av. 119

Av. 121

Av. 123

Av. 125

Av. 127

Av. 129

Av. 131

Av. 133

Av. 135

Av. 137

Av. 139

Av. 141

Av. 143

Av. 145

Av. 147

Av. 149

Av. 151

Av. 153

Av. 155

Av. 157

Av. 159

Av. 161

Av. 163

Av. 165

Av. 167

Av. 169

Av. 171

Av. 173

Av. 175

Av. 177

Av. 179

Av. 181

Av. 183

Av. 185

Av. 187

Av. 189

Av. 191

Av. 193

Av. 195

Av. 197

Av. 199

Av. 201

Av. 203

Av. 205

Av. 207

Av. 209

Av. 211

Av. 213

Av. 215

Av. 217

Av. 219

Av. 221

Av. 223

Av. 225

Av. 227

Av. 229

Av. 231

Av. 233

Av. 235

Av. 237

Av. 239

Av. 241

Av. 243

Av. 245

Av. 247

Av. 249

Av. 251

Av. 253

Av. 255

Av. 257

Av. 259

Av. 261

Av. 263

Av. 265

Av. 267

Av. 269

Av. 271

Av. 273

Av. 275

Av. 277

Av. 279

Av. 281

Av. 283

Av. 285

Av. 287

Av. 289

Av. 291

Av. 293

Av. 295

Av. 297

Av. 299

Av. 301

Av. 303

Av. 305

Av. 307

Av. 309

Av. 311

Av. 313

Av. 315

Av. 317

Av. 319

Av. 321

Av. 323

Av. 325

Av. 327

Av. 329

Av. 331

Av. 333

Av. 335

Av. 337

Av. 339

Av. 341

Av. 343

Av. 345

Av. 347

Av. 349

Av. 351

Av. 353

Av. 355

Av. 357

Av. 359

Av. 361

Av. 363

Av. 365

Av. 367

Av. 369

Av. 371

Av. 373

Av. 375

Av. 377

Av. 379

Av. 381

Av. 383

Av. 385

Av. 387

Av. 389

Av. 391

Av. 393

Av. 395

Av. 397

Av. 399

Av. 401

Av. 403

Av. 405

Av. 407

Av. 409

Av. 411

Av. 413

Av. 415

Av. 417

Av. 419

Av. 421

Av. 423

Av. 425

Av. 427

Av. 429

Av. 431

Av. 433

Av. 435

Av. 437

Av. 439

Av. 441

Av. 443

Av. 445

Av. 447

Av. 449

Av. 451

Av. 453

Av. 455

Av. 457

Av. 459

Av. 461

Av. 463

Av. 465

Av. 467

Av. 469

Av. 471

Av. 473

Av. 475

Av. 477

Av. 479

Av. 481

Av. 483

Av. 485

Av. 487

Av. 489

Av. 491

Av. 493

Av. 495

Av. 497

Av. 499

Av. 501

Av. 503

Av. 505

Av. 507

Av. 509

Av. 511

Av. 513

Av. 515

Av. 517

Av. 519

Av. 521

Av. 523

Av. 525

Av. 527

Av. 529

Av. 531

Av. 533

Av. 535

Av. 537

Av. 539

Av. 541

Av. 543

Av. 545

Av. 547

Av. 549

Av. 551

Av. 553

Av. 555

Av. 557

Av. 559

Av. 561

Av. 563

Av. 565

Av. 567

Av. 569

Av. 571

Av. 573

Av. 575

Av. 577

Av. 579

Av. 581

Av. 583

Av. 585

Av. 587

Av. 589

Av. 591

Av. 593

Av. 595

Av. 597

Av. 599

Av. 601

Av. 603

Av. 605

Av. 607

Av. 609

Av. 611

Av. 613

Av. 615

Av. 617

Av. 619

Av. 621

Av. 623

Av. 625

Av. 627

Av. 629

Av. 631

Av. 633

Av. 635

Av. 637

Av. 639

Av. 641

Av. 643

Av. 645

Av. 647

Av. 649

Av. 651

Av. 653

Av. 655

Av. 657

Av. 659

Av. 661

Av. 663

Av. 665

Av. 667

Av. 669

Av. 671

Av. 673

Av. 675

Av. 677

Av. 679

Av. 681

Av. 683

Av. 685

Av. 687

Av. 689

Av. 691

Av. 693

Av. 695

Av. 697

Av. 699

Av. 701

Av. 703

Av. 705

Av. 707

Av. 709

Av. 711

Av. 713

Av. 715

Av. 717

Av. 719

Av. 721

Av. 723

Av. 725

Av. 727

Av. 729

Av. 731

Av. 733

Av. 735

Av. 737

Av. 739

Av. 741

Av. 743

Av. 745

Av. 747

Av. 749

Av. 751

Av. 753

Av. 755

Av. 757

Av. 759

Av. 761

Av. 763

Av. 765

Av. 767

Av. 769

Av. 771

Av. 773

Av. 775

Av. 777

Av. 779

Av. 781

Av. 783

Av. 785

Av. 787

Av. 789

Av. 791

Av. 793

Av. 795

Av. 797

Av. 799

Av. 801

Av. 803

Av. 805

Av. 807

Av. 809

Av. 811

Av. 813

Av. 815

Av. 817

Av. 819

Av. 821

Av. 823

Av. 825

Av. 827

Av. 829

Av. 831

Av. 833

Av. 835

Av. 837

Av. 839

Av. 841

Av. 843

Av. 845

Av. 847

Av. 849

Av. 851

Av. 853

Av. 855

Av. 857

Av. 859

Av. 861

Av. 863

Av. 865

Av. 867

Av. 869

Av. 871

Av. 873

Av. 875

Av. 877

Av. 879

Av. 881

Av. 883

Av. 885

Av. 887

Av. 889

Av. 891

Av. 893

Av. 895

Av. 897

Av. 899

Av. 901

Av. 903

Av. 905

Av. 907

Av. 909

Av. 911

Av. 913

Av. 915

Av. 917

Av. 919

Av. 921

Av. 923

Av. 925

Av. 927

Av. 929

Av. 931

Av. 933

Av. 935

Av. 937

Av. 939

Av. 941

Av. 943

Av. 945

Av. 947

Av. 949

Av. 951

Av. 953

Av. 955

Av. 957

Av. 959

Av. 961

Av. 963

Av. 965

Av. 967

Av. 969

Av. 971

Av. 973

Av. 975

Av. 977

Av. 979

Av. 981

Av. 983

Av. 985

Av. 987

Av. 989

Av. 991

Av. 993

Av. 995

Av. 997

Av. 999

Av. 1001

Av. 1003

Av. 1005

Av. 1007

Av. 1009

Av. 1011

Av. 1013

Av. 1015

Av. 1017

Av. 1019

Av. 1021

Av. 1023

Av. 1025

Av. 1027

Av. 1029

Av. 1031

Av. 1033

Av. 1035

Av. 1037

Av. 1039

Av. 1041

Av. 1043

Av. 1045

Av. 1047

Av. 1049

Av. 1051

Av. 1053

Av. 1055

Av. 1057

Av. 1059

Av. 1061

Av. 1063

Av. 1065

Av. 1067

Av. 1069

Av. 1071

Av. 1073

Av. 1075

Av. 1077

Av. 1079

Av. 1081

Av. 1083

Av. 1085

Av. 1087

Av. 1089

Av. 1091

Av. 1093

Av. 1095

Av. 1097

Av. 1099

Av. 1101

Av. 1103

Av. 1105

Av. 1107

Av. 1109

Av. 1111

Av. 1113

Av. 1115

Av. 1117

Av. 1119

Av. 1121

Av. 1123

Av. 1125

Av. 1127

Av. 1129

Av. 1131

Av. 1133

Av. 1135

Av. 1137

Av. 1139

Av. 1141

Av. 1143

Av. 1145

Av. 1147

Av. 1149

Av. 1151

Av. 1153

Av. 1155

Av. 1157

Av. 1159

Av. 1161

Av. 1163

Av. 1165

Av. 1167

Av. 1169

Av. 1171

Av. 1173

Av. 1175

Av. 1177

Av. 1179

Av. 1181

Av. 1183

Av. 1185

Av. 1187

Av. 1189

Av. 1191

Av. 1193

Av. 1195

Av. 1197

Av. 1199

Av. 1201

Av. 1203

Av. 1205

Av. 1207

Av. 1209

Av. 1211

Av. 1213

Av. 1215

Av. 1217

Av. 1219

Av. 1221

Av. 1223

Av. 1225

Av. 1227

Av. 1229

Av. 1231

Av. 1233

Av. 1235

Av. 1237

Av. 1239

Av. 1241

Av. 1243

Av. 1245

Av. 1247

Av. 1249

Av. 1251

Av. 1253

Av. 1255

Av. 1257

Av. 1259

Av. 1261

Av. 1263

Av. 1265

Av. 1267

Av. 1269

Av. 1271

Av. 1273

Av. 1275

Av. 1277

Av. 1279

Av. 1281

Av. 1283

Av. 1285

Av. 1287

Av. 1289

Av. 1291

Av. 1293

Av. 1295

Av. 1297

Av. 1299

Av. 1301

Av. 1303

Av. 1305

Av. 1307

Av. 1309

Av. 1311

Av. 1313

Av. 1315

Av. 1317

Av. 1319

Av. 1321

Av. 1323

Av. 1325

Av. 1327

Av. 1329

Av. 1331

Av. 1333

Av. 1335

Av. 1337

Av. 1339

Av. 1341

Av. 1343

Av. 1345

Av. 1347

Av. 1349

Av. 1351

Av. 1353

Av. 1355

Av. 1357

Av. 1359

Av. 1361

Av. 1363

Av. 1365

Av. 1367

Av. 1369

Av. 1371

Av. 1373

Av. 1375

Av. 1377

Av. 1379

Av. 1381

Av. 1383

Av. 1385

Av. 1387

Av. 1389

Av. 1391

Av. 1393

Av. 1395

Av. 1397

Av. 1399

Av. 1401

Av. 1403

Av. 1405

Av. 1407

Av. 1409

Av. 1411

Av. 1413

Av. 1415

Av. 1417

Av. 1419

Av. 1421

Av. 1423

Av. 1425

Av. 1427

Av. 1429

Av. 1431

Av. 1433

Av. 1435

Av. 1437

Av. 1439

Av. 1441

Av. 1443

Av. 1445

Av. 1447

Av. 1449

Av. 1451

Av. 1453

Av. 1455

Av. 1457

Av. 1459

Av. 1461

Av. 1463

Av. 1465

Av. 1467

Av. 1469

Av. 1471

Av. 1473

Av. 1475

Av. 1477

Av. 1479

Av. 1481

Av. 1483

Av. 1485

Av. 1487

Av. 1489

Av. 1491

Av. 1493

Av. 1495

Av. 1497

Av. 1499

Av. 1501

Av. 1503

Av. 1505

Av. 1507

Av. 1509

Av. 1511

Av. 1513

Av. 1515

Av. 1517

Av. 1519

Av. 1521

Av. 1523

Av. 1525

Av. 1527

Av. 1529

Av. 1531

Av. 1533

Av. 1535

Av. 1537

Av. 1539

Av. 1541

Av. 1543

Av. 1545

Av. 1547

Av. 1549

Av. 1551

Av. 1553

Av. 1555

Av. 1557

Av. 1559

Av. 1561

Av. 1563

Av. 1565

Av. 1567

Av. 1569

Av. 1571

Av. 1573

Av. 1575

Av. 1577

Av. 1579

Av. 1581

Av. 1583

Av. 1585

Av. 1587

Av. 1589

Av. 1591

Av. 1593

Av. 1595

Av. 1597

Av. 1599

Av. 1601

Av. 1603

Av. 1605

Av. 1607

Av. 1609

Av. 1611

Av. 1613

Av. 1615

Av. 1617

Av. 1619

Av. 1621

Av. 1623

Av. 1625

Av. 1627

Av. 1629

Av. 1631

Av. 1633

Av. 1635

Av. 1637

Av. 1639

Av. 1641

Av. 1643

Av. 1645

Av. 1647

Av. 1649

Av. 1651

Av. 1653

Av. 1655

Av. 1657

Av. 1659

Av. 1661

Av. 1663

Av. 1665

Av. 1667

Av. 1669

Av. 1671

Av. 1673

Av. 1675

Av. 1677

Av. 1679

Av. 1681

Av. 1683

Av. 1685

Av. 1687

Av. 1689

Av. 1691

Av. 1693

Av. 1695

Av. 1697

Av. 1699

Av. 1701

Av. 1703

Av. 1705

Av. 1707

Av. 1709

Av. 1711

Av. 1713

Av. 1715

Av. 1717

Av. 1719

Av. 1721

Av. 1723

Av. 1725

Av. 1727

Av. 1729

Av. 1731

Av. 1733

Av. 1735

Av. 1737

Av. 1739

Av. 1741

Av. 1743

Av. 1745

Av. 1747

Av. 1749

Av. 1751

Av. 1753

Av. 1755

Av. 1757

Av. 1759

Av. 1761

Av. 1763

Av. 1765

Av. 1767

Av. 1769

Av. 1771

Av. 1773

Av. 1775

Av. 1777

Av. 1779

Av. 1781

Av. 1783

Av. 1785

Av. 1787

Av. 1789

Av. 1791

Av. 1793

Av. 1795

Av. 1797

Av. 1799

Av. 1801

Av. 1803

Av. 1805

Av. 1807

Av. 1809

Av. 1811

Av. 1813

Av. 1815

Av. 1817

Av. 1819

Av. 1821

Av. 1823

Av. 1825

Av. 1827

Av. 1829

Av. 1831

Av. 1833

Av. 1835

Av. 1837

Av. 1839

Av. 1841

Av. 1843

Av. 1845

Av. 1847

Av. 1849

Av. 1851

Av. 1853

Av. 1855

Av. 1857

Av. 1859

Av. 1861

Av. 1863

Av. 1865

Av. 1867

Av. 1869

Av. 1871

Av. 1873

Av. 1875

Av. 1877

Av. 1879

Av. 1881

Av. 1883

Av. 1885

Av. 1887

Av. 1889

Av. 1891

Av. 1893

Av. 1895

Av. 1897

Av. 1899

Av. 1901

Av. 1903

Av. 1905

Av. 1907

Av. 1909

Av. 1911

Av. 1913

Av. 1915

Av. 1917

Av. 1919

Av. 1921

Av. 1923

Av. 1925

Av. 1927

Av. 1929

Av. 1931

Av. 1933

Av. 1935

Av. 1937

Av. 1939

Av. 1941

Av. 1943

Av. 1945

Av. 1947

Av. 1949

Av. 1951

Av. 1953

Av. 1955

Av. 1957

Av. 1959

Av. 1961

Av. 1963

Av. 1965

Av. 1967

Av. 1969

Av. 1971

Av. 1973

Av. 1975

Av. 1977

Av. 1979

Av. 1981

Av. 1983

Av. 1985

Av. 1987

Av. 1989

Av. 1991

Av. 1993

Av. 1995

Av. 1997

Av. 1999

Av. 2001

Av. 2003

Av. 2005

Av. 2007

Av. 2009

Av. 2011

Av. 2013

Av. 2015

Av. 2017

Av. 2019

Av. 2021

Av. 2023

Av. 2025

Av. 2027

Av. 2029

Av. 2031

Av. 2033

Av. 2035

Av. 2037

Av. 2039

Av. 2041

Av. 2043

Av. 2045

Av. 2047

Av. 2049

Av. 2051

Av. 2053

Av. 2055

Av. 2057

Av. 2059

Av. 2061

Av. 2063

Av. 2065

Av. 2067

Av. 2069

Av. 2071

Av. 2073

Av. 2075

Av. 2077

<

Para agregar el POI “Plaza Moreno”, debemos elegir ese POI en el combo denominado “POI”, y una vez seleccionado, se nos muestra información detallada del POI elegido, como la ubicación (latitud/longitud), una descripción, un mapa de ubicación y la condición de dicho POI (en este caso es de *Coordinación*). En la Figura 38 se puede apreciar que se eligió el POI “Plaza Moreno”, y esta elección resulta en la información mostrada en dicha figura. Para que se realice la incorporación de este POI al camino hay que presionar el botón *Agregar*.

Asignar camino secuencial para Grupo Grupo Rojo

POI:  
28 (Plaza Moreno)

Ubicación de la pieza  
Latitud  
-34.9213

Longitud  
-57.9545

Referencia  
Plaza Moreno

Coordinación  
Coordinación

Mapa Satélite

Google

Datos de mapas ©2011 Google Términos de uso Informar de un error de Mapa

Agregar Cancelar

Figura 38 – Tercer POI en el camino del Grupo Rojo

Observemos ahora que en el sector inferior de la pantalla (Figura 39) se puede apreciar cómo está conformado el camino de POI para el *Grupo Rojo* hasta el momento.

| Lista de POI asociados al Grupo |          |          |                       |              |
|---------------------------------|----------|----------|-----------------------|--------------|
| Orden                           | Latitud  | Longitud | Referencia            | Tipo de POI  |
| 1                               | -34.9112 | -57.955  | Plaza Italia          | Comun        |
| 2                               | -34.9276 | -57.9738 | Plaza 19 de Noviembre | Comun        |
| 3                               | -34.9213 | -57.9545 | Plaza Moreno          | Coordinacion |

Eliminar EliminarAJAX

Figura 39 - Detalle del camino del Grupo Rojo

Falta sólo agregar el último POI "Plaza Matheu" de condición *Común*. En la Figura 40 se puede apreciar que se eligió el POI "Plaza Matheu", y esta elección resulta en la información mostrada en dicha figura. Para que se realice la incorporación de este POI al camino hay que presiona el botón *Agregar*.

Asignar camino secuencial para Grupo Grupo Rojo

POI:

Ubicación de la pieza

Latitud

Longitud

Referencia

Coordinación

Volviendo al menú de opciones de grupos (mostrado en la Figura 34), vemos que al presionar el botón “Ver camino” a cada grupo se le despliega en un mapa los POI que deben visitar los subgrupos de su grupo (Figura 41) junto con el orden en que deben ser recorridos y un color distintivo para los POI de *Coordinación* y los *Comunes*. Cabe destacar que en verde, tenemos señalizados los POI *Común* y en celeste los POI de *Coordinación*.

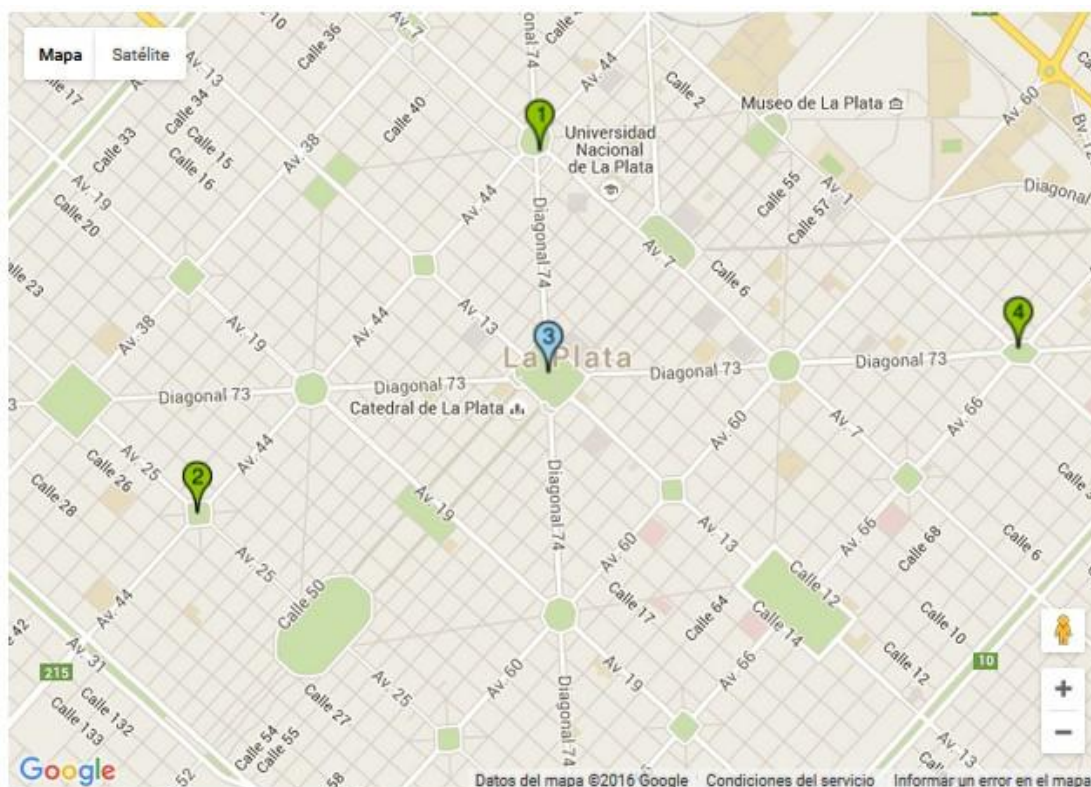


Figura 41: Recorrido para los subgrupos de Grupo Rojo

Ahora detallamos el camino secuencial del *Grupo Azul*, el cual se compone de los siguientes POI:

1. *Parque Saavedra* (POI de *Coordinación*).
2. *Plaza Sarmiento* (POI *Común*).
3. *Estación de Trenes* (POI *Común*).
4. *Facultad de Informática* (POI de *Coordinación*).

Para su conformación se siguen los mismos pasos ya explicados para la definición de un camino secuencial y en la Figura 42 se puede apreciar la lista final resultante, quedando así definido el camino secuencial para el *Grupo Azul*.

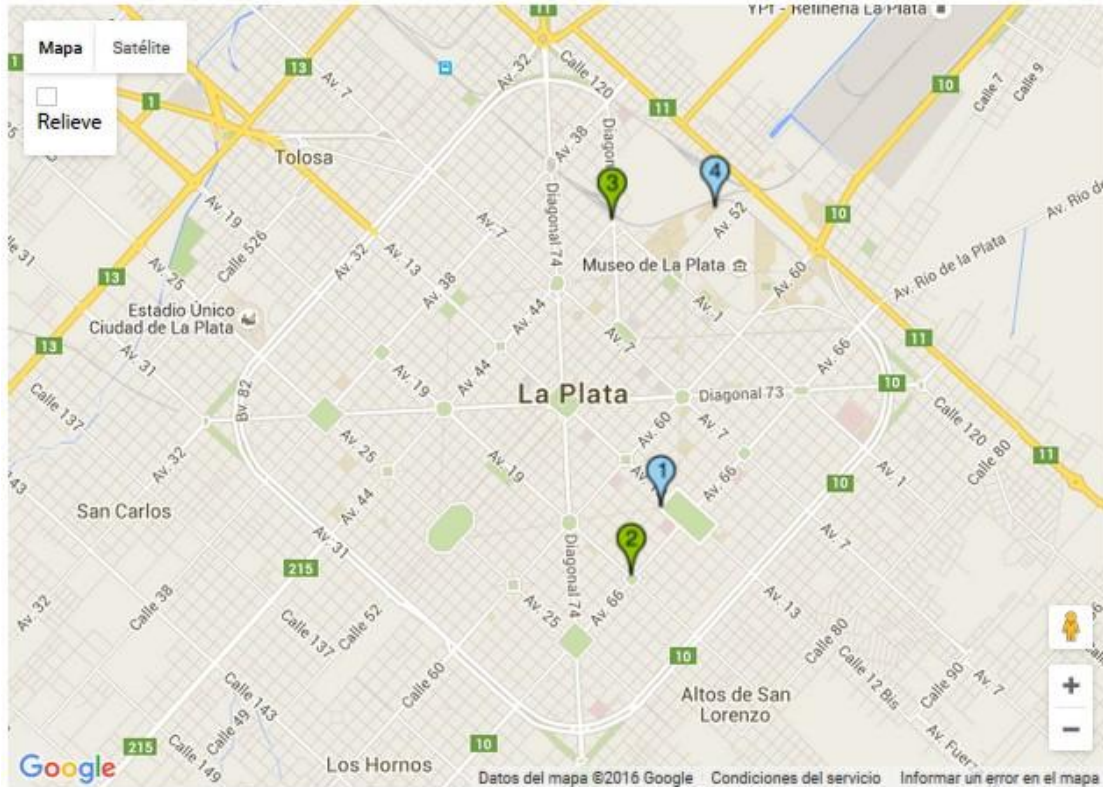


Figura 42: Camino secuencial definido para el Grupo Azul

Veamos ahora más detalles de la opción *Estrategias de Coordinación* del menú de opciones (mostrado en la Figura 27). Al seleccionar esta opción aparece una pantalla como se muestra en la Figura 43. Se puede apreciar un listado de estrategias de coordinación que se pueden elegir para utilizarse en el juego. De cada una se detalla el nombre, la descripción, el estado y diferentes operaciones que se pueden llevar a cabo. Estas opciones son: *“Editar”*, *“Detalle”* y *“Set Active”*. Esta última operación (*Set Active*) establece la estrategia activa a utilizar en la instanciación del siguiente juego.

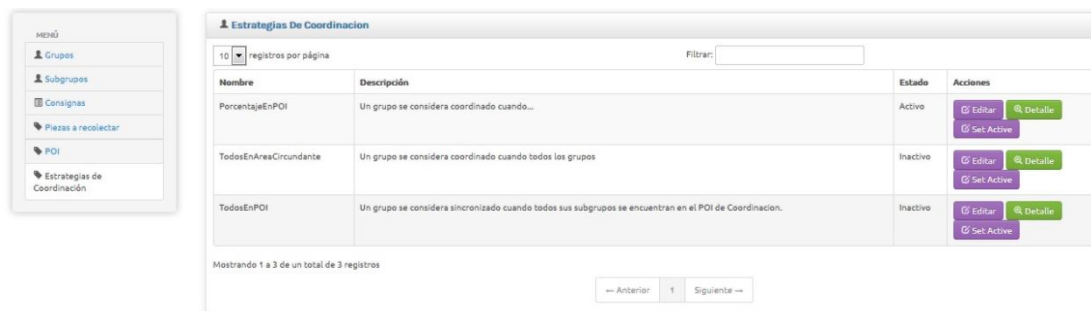


Figura 43 - Listado de estrategias de coordinación

La operación *Editar* (de la Figura 43) permite configurar para cada estrategia

algunos parámetros propios de la misma. Es decir, cada estrategia de coordinación puede tener parámetros variables que pueden ser editables. En la Figura 44, podemos apreciar la edición de la estrategia *PorcentajeEnPOI*, en la cual podemos editar el valor del porcentaje (valor entre 1 y 100) con el cuál, el juego considera coordinado a un grupo en un POI de Coordinación.

Recordemos que la estrategia *PorcentajeEnPOI* define un porcentaje y considerará coordinado un grupo, si la cantidad de subgrupos de dicho grupo representa (como mínimo) el porcentaje definido para dicho *POI* de Coordinación.

The screenshot shows a web interface for editing a coordination strategy. On the left is a vertical menu with options: Grupos, Subgrupos, Consignas, Piezas a recolectar, POI, and Estrategias de Coordinación. The main area is titled 'Editar Estrategia De Coordinación' and contains the following fields: 'Nombre' (PorcentajeEnPOI), 'Descripcion' (Un grupo se considera coordinado cuando...), and 'Porcentaje' (50). At the bottom are 'Guardar' and 'Cancelar' buttons.

Figura 44 - Edición de la Estrategia de Coordinación “PorcentajeEnPOI”

En la Figura 45 se presenta la pantalla para configurar la estrategia *TodosEnAreaCircundante*, en la cual se puede editar el área (radio de distancia).

The screenshot shows the same web interface as Figure 44, but for a different strategy. The 'Nombre' field is 'TodosEnAreaCircundante' and the 'Descripcion' is 'Un grupo se considera coordinado cuando todos los grupos'. The 'Area' field is set to '1500'. The 'Guardar' and 'Cancelar' buttons are at the bottom.

Figura 45 - Edición de la Estrategia de Coordinación “TodosEnAreaCircundante”

El detalle de cada estrategia de coordinación puede ser visualizado seleccionando la operación *Detalle* que vemos en la Figura 43. Por ejemplo, al seleccionar opción *Detalle* para la estrategia *TodosEnAreaCircundante* se visualiza una pantalla como la mostrada en la Figura 46, donde se puede apreciar el nombre, la descripción y ver si es la estrategia activa (valor 1 si es la estrategia Activa o 0 en caso contrario).

| Estrategia De Coordinación |  |
|----------------------------|--|
| Nombre                     | TodosEnAreaCircundante                                   |
| Descripción                | Un grupo se considera coordinado cuando todos los grupos |
| Is Activa                  | 0  |

Figura 46 - Detalles de la estrategia

En lo que respecta a la parte cliente del prototipo, no hay cambios en cuanto a la experiencia del usuario. Es decir, de la misma manera que en el prototipo definido en [Apezteguía and Rapetti, 2014], el usuario una vez logueado en el juego; debe recorrer los diferentes POI que van apareciendo en su pantalla. Por cada uno de ellos, debe dar una respuesta sobre la pieza que se le presente de acuerdo a la consigna asignada a su grupo. La única diferencia de nuestro prototipo, respecto del tomado como base, es que en el nuestro los POI van siendo visualizados uno a uno, conforme al camino secuencial definido para el grupo. En [Apezteguía and Rapetti, 2014], una vez alcanzada la posición del POI inicial, el subgrupo visualizaba en su mapa todos los POI que tenía que recorrer para luego ir caminándolos de acuerdo a su propio criterio. En la Figura 47 se puede apreciar cómo en nuestro prototipo, se le presentan los diferentes POI del camino a recorrer, por ejemplo, al *Subgrupo 1*, uno a uno de manera consecutiva se muestran las distintas pantallas.

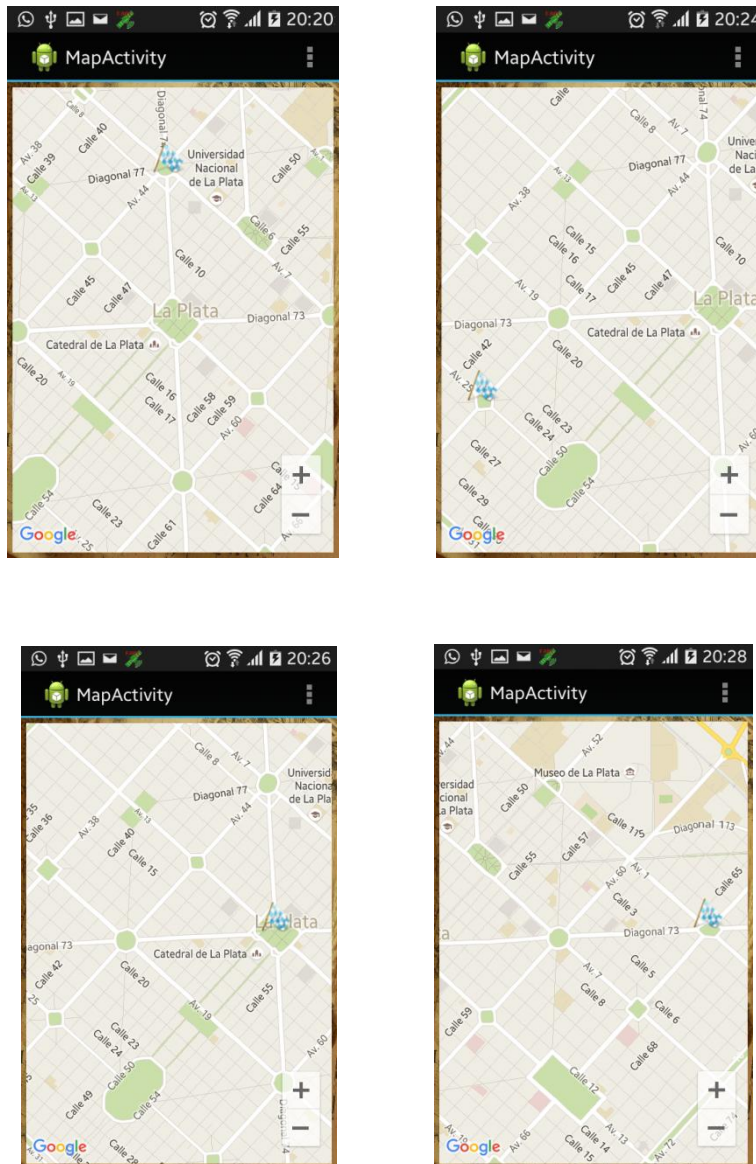
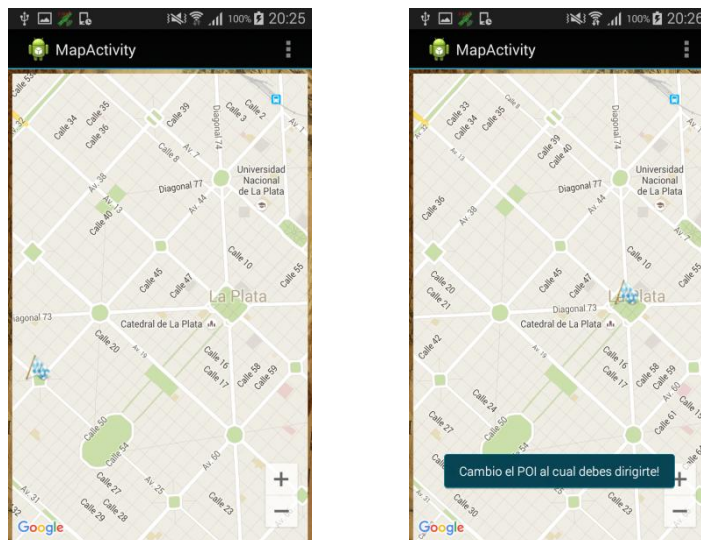


Figura 47- *Subgrupo 1* va descubriendo los diferentes POI destino, uno por uno; en la medida que van avanzando en el juego.

En nuestro prototipo, puede darse el caso en que un subgrupo que se encuentra dirigiéndose a un POI, cambie inmediatamente de POI destino, dado que algún subgrupo del grupo pudo ya haber arribado al POI al que se dirigían y haber dado una respuesta para la pieza asociada. Para graficarlo con un ejemplo, supongamos que tenemos a *Subgrupo 1* y *Subgrupo 2* (que pertenecen al mismo *Grupo Rojo*) dirigiéndose hacia el POI “*Plaza 19 de Noviembre*”. Suponiendo que *Subgrupo 1* llega primero a dicho POI, da una respuesta para la pieza asociada y a continuación se le presenta a ambos subgrupos el siguiente POI que es “*Plaza Moreno*”. En este caso, el *Subgrupo 2* que se dirigía hacia “*Plaza*



19 de Noviembre” recibe una notificación del cambio de POI destino y se le presenta el próximo POI actualizado. Esta notificación de cambio de POI se ve reflejada en la Figura 48.



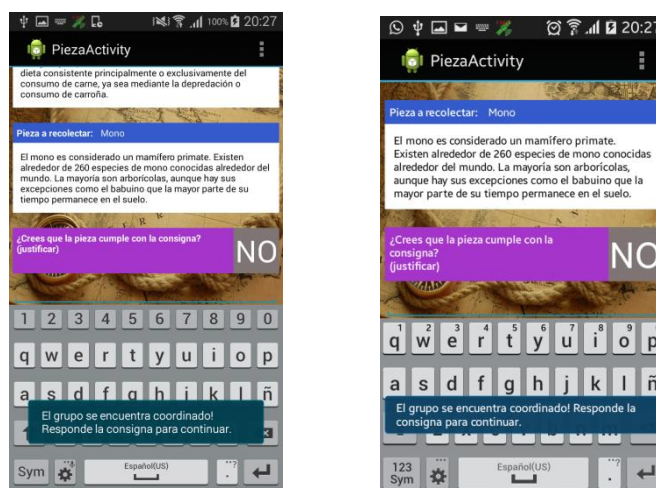
**Figura 48 - (Izq.) Subgrupo 2 tiene como destino “Plaza 19 de Noviembre” (Der.) Subgrupo 2 recibe su próximo POI actualizado y es notificado del cambio de destino.**

Una de las características fundamentales introducidas en nuestro trabajo, es la posibilidad de coordinar subgrupos en los POI de *Coordinación*, para resolver consignas o tareas en conjunto. Cuando un subgrupo arriba a un POI de *Coordinación*, el *Juego* detecta la presencia del subgrupo en el POI mediante sensores (usando el GPS del dispositivo), y le envía un mensaje para que espere a que la coordinación del grupo se complete para poder continuar. Se puede ver el mensaje que recibe un subgrupo al arribar a un POI de *Coordinación* en la Figura 49.



**Figura 49 - Mensaje que reciben los subgrupos al arribar a un POI De Coordinación.**

Cuando el criterio de coordinación se cumple para un grupo, en un POI de *Coordinación*, el *Juego* notifica a todos los subgrupos del grupo que la coordinación del grupo se encuentra realizada y les presenta la pieza asociada al POI para que puedan resolver la consigna. En la Figura 50, se puede ver los mensajes que reciben los subgrupos una vez coordinados y además la presentación de la pieza asociada.



**Figura 50 – (Izq.) Subgrupo 1 es notificado que la coordinación se encuentra completa y recibe la pieza (Der.) Subgrupo 2 también es notificado que la coordinación se encuentra completa y recibe la pieza**

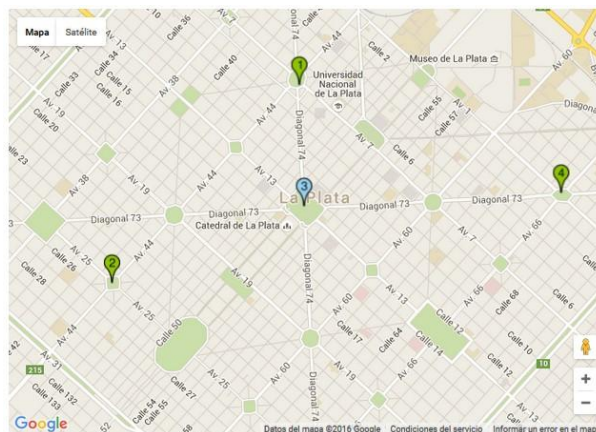
## 5. Simulación de un Juego

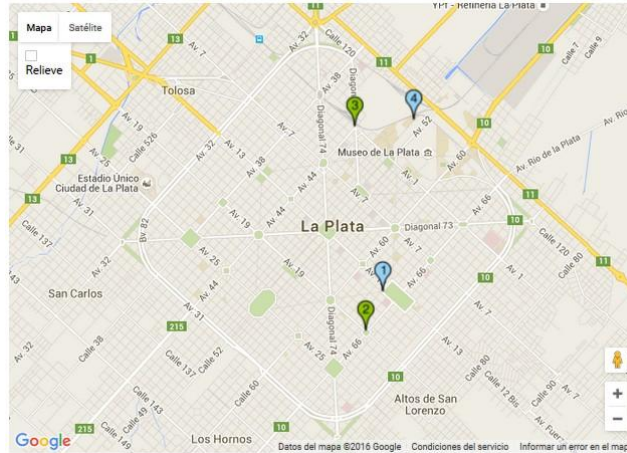
En este capítulo se presentará una simulación del prototipo presentado en el Capítulo 4. Es decir, se va a simular un juego donde vamos a mostrar cómo es la interacción de los subgrupos y cómo se desarrolla el mismo con las nuevas características de coordinación presentadas en nuestro prototipo.

Para la simulación del juego utilizamos aquellos elementos presentados en el capítulo anterior como grupos, subgrupos y los caminos secuenciales generados para ellos. También hacemos uso de algunas de las estrategias de coordinación presentadas en el Capítulo 3.

A continuación se listan algunas consideraciones a tener en cuenta para la simulación presentada:

- Participan los dos grupos presentados en el Capítulo 4 en las simulaciones (*Grupo Rojo* y *Grupo Azul*) donde cada uno tiene dos subgrupos asociados. Del *Grupo Rojo* forman parte *Subgrupo 1* y *Subgrupo 2*. A su vez, el *Grupo Azul* está conformado por *Subgrupo 3* y *Subgrupo 4*.
- La consigna de *Juego* para ambos grupos es diferente. *Grupo Rojo* debe identificar animales carnívoros mientras que *Grupo Azul* tiene que hacer lo mismo con animales herbívoros.
- Cada grupo tiene definido un camino secuencial a recorrer. Ambos caminos son distintos (conformados por POI diferentes) entre sí y están conformados por cuatro POI (algunos POI *Común* y otros POI *de Coordinación*). En las Figuras 51 y 52 apreciamos los caminos ya definidos para ambos grupos. (en el Capítulo 4)





**Figura 52: Camino definido para el Grupo Azul**

Realizamos dos simulacros de juego. En el primer simulacro, hacemos enfoque en la participación del *Grupo Rojo*, con *Subgrupo 1* y *Subgrupo 2* y para el segundo, el enfoque está puesto en la participación del *Grupo Azul*, con *Subgrupo 3* y *Subgrupo 4*. La estrategia de coordinación a utilizar en el primer simulacro es “*TodosEnPOI*”, mientras que para el segundo utilizamos “*PorcentajeEnPOI*”. En estos simulacros, cada subgrupo recorre secuencialmente los POI del camino asignado a su grupo, y por cada POI, se le presenta la pieza asociada y responde si cumple o no con la consigna.

Cuando un subgrupo llega a un POI *Común* se le presenta la pieza asociada a dicho POI, solamente al subgrupo que acaba de arribar. El resto de los subgrupos del POI, en un principio, no se entera de que un compañero de grupo ya ha llegado al destino buscado (eventualmente podrían ser consultados para ayudar a dar la respuesta, como en el prototipo original).

Cuando un subgrupo llega a un POI *de Coordinación* y se cumple el criterio de coordinación, según la estrategia de coordinación activa para el juego en curso, se le presenta la pieza asociada a dicho POI a todos los subgrupos del grupo recientemente coordinado. Cualquiera de éstos puede dar la respuesta a la consigna y se toma como válida la primera que el juego procese.

Habiendo enunciado las diferentes consideraciones a tener en cuenta para nuestros simulacros, comenzamos explicando detalladamente las interacciones de los subgrupos y el desarrollo del juego.

- **Primera Simulación: Foco en el Grupo Rojo**

En esta simulación hacemos foco en la participación del *Grupo Rojo*, con *Subgrupo 1* y *Subgrupo 2*, y se está utilizando la *Estrategia de Coordinación "TodosEnPOI"*.

Como primer paso, ambos subgrupos se loguean en sus dispositivos correspondientes, la forma de hacerlo no varía con respecto al prototipo tomado como base. En la Figura 53 se puede apreciar el login de ambos subgrupos. A la izquierda de la figura visualizamos la pantalla para el *Subgrupo 1* y a la derecha la del *Subgrupo 2*. Cabe aclarar que esta será la manera de presentar las pantallas de ambos subgrupos durante todo el trabajo.

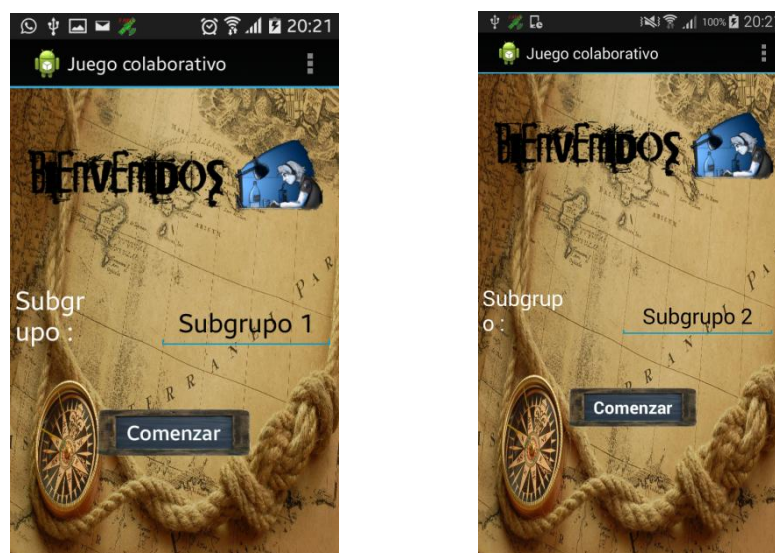
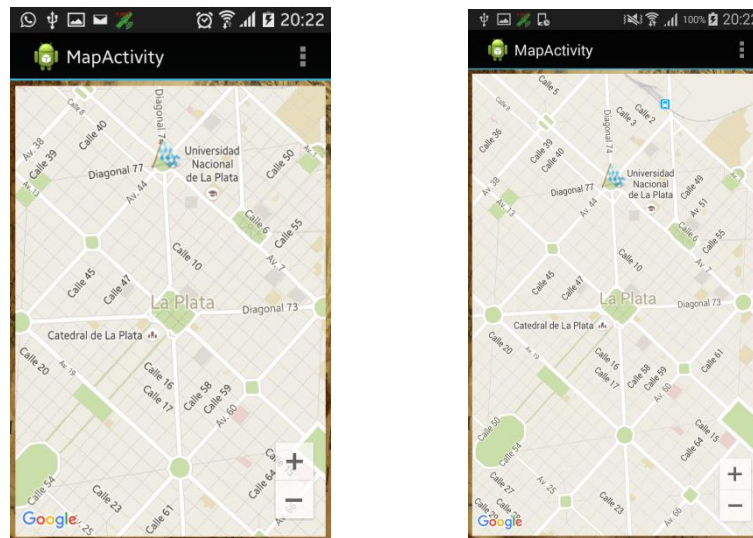


Figura 53: Login de *Subgrupo 1* y *Subgrupo 2* en la aplicación (Izq: *Subgrupo 1*, Der: *Subgrupo 2*)

De la misma manera que en el prototipo base, al loguearse el usuario, se hacen las validaciones correspondientes y si los datos proporcionados son correctos, se visualiza en el mapa el primer POI en el recorrido del subgrupo que se acaba de loguear. Esto se puede apreciar en la Figura 54. Tanto el *Subgrupo 1* como el *Subgrupo 2* reciben el mismo POI "Plaza Italia" como POI inicial.



**Figura 54: Ambos subgrupos reciben el mismo primer POI al cual deben dirigirse  
(Izq: Subgrupo 1, Der: Subgrupo 2)**

A continuación, ambos subgrupos se desplazan en dirección al punto marcado en el mapa (en este caso “*Plaza Italia*”) y una vez que alguno de los dos arriba al lugar, al ser éste un POI *Común*, se le presenta inmediatamente la pieza asociada a dicho POI, para que el subgrupo pueda dar una respuesta.

Supongamos que *Subgrupo 1* es el primero en llegar a “*Plaza Italia*”, por lo cual la pieza asociada se le presenta a este subgrupo como se puede apreciar en la Figura 55. También podemos ver en esta misma figura que al momento de dar la respuesta el *Subgrupo 1* puede consultar con sus compañeros mediante la opción “*Consultar*” (de la misma manera que se hacía en el prototipo base) o simplemente puede dar su respuesta con la opción “*Enviar*”. Esa característica se mantiene y funciona de la misma manera del prototipo base. Por otro lado, el *Subgrupo 2* sigue visualizando en el mapa “*Plaza Italia*” hasta tanto el *Subgrupo 1* no termine de dar respuesta a la consigna.

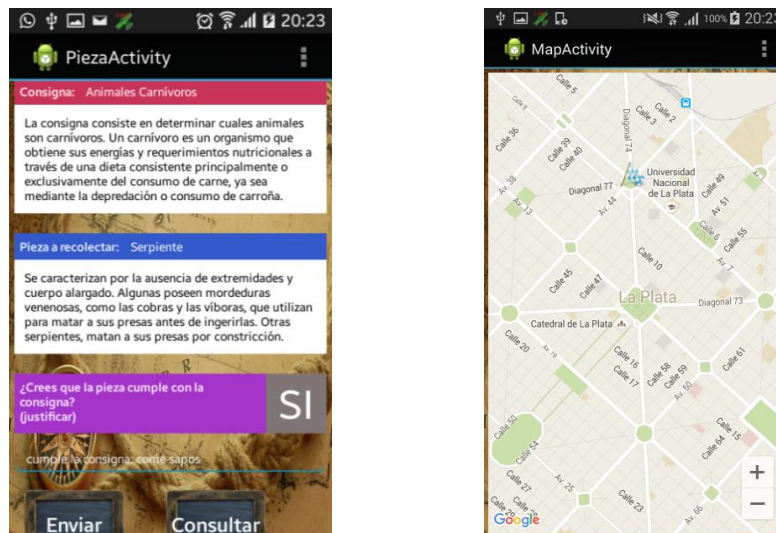


Figura 55: Al *Subgrupo 1* se le presenta la pieza asociada a “Plaza Italia”, mientras que el *Subgrupo 2* sigue viendo este POI en su mapa (Izq: *Subgrupo 1*, Der: *Subgrupo 2*)

Una vez dada la respuesta por parte del *Subgrupo 1*, se considera resuelta la consigna para dicho POI y se obtiene el próximo POI en el recorrido para los subgrupos del *Grupo Rojo*. Como se puede observar en la pantalla de la izquierda en la Figura 56, tras dar la respuesta el *Subgrupo 1* simplemente visualiza en el mapa el próximo POI en el recorrido, mientras que al *Subgrupo 2* (la pantalla de la derecha en la Figura 56) se le notifica que el POI al cual debe dirigirse ha cambiado al mismo tiempo que visualiza el nuevo destino. Ambos subgrupos ven en sus pantallas el nuevo POI destino, “Plaza 19 de Noviembre” y se dirigen hacia allí.

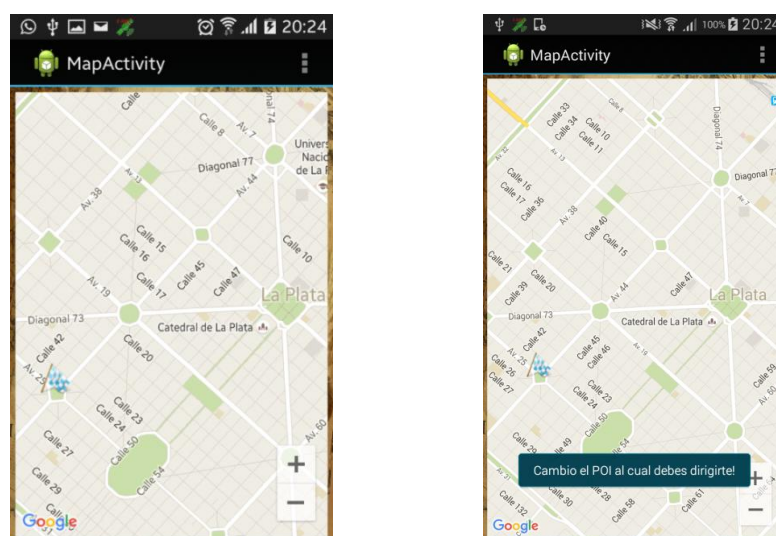
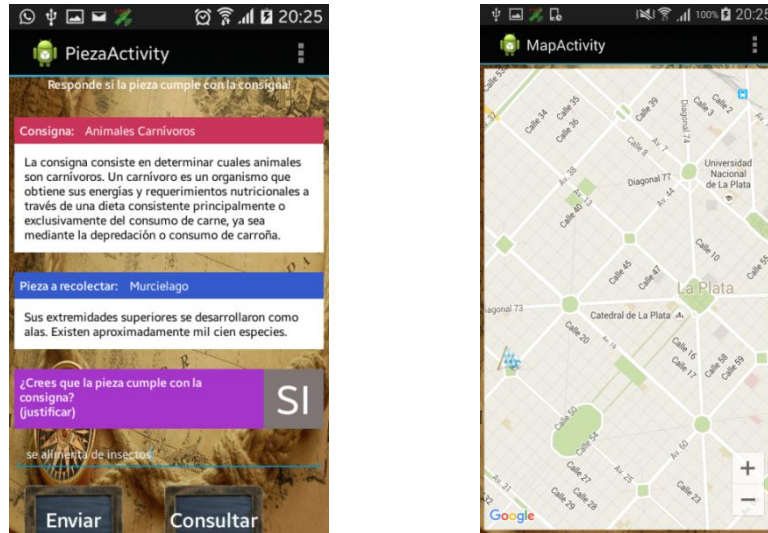


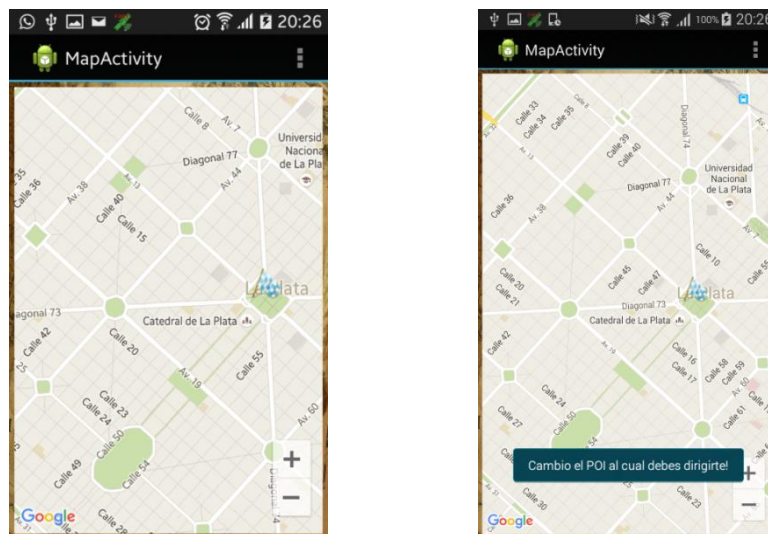
Figura 56: El *Subgrupo 1* visualiza el nuevo destino. El *Subgrupo 2* es notificado que el POI destino ha cambiado al mismo tiempo que visualiza el segundo POI del recorrido (Izq: *Subgrupo 1*, Der: *Subgrupo 2*)

Nuevamente, supongamos que el *Subgrupo 1* es el primero en llegar, en este caso, a “Plaza 19 de Noviembre”. Al ser también éste un POI Común, al *Subgrupo 1* se le presenta inmediatamente la consigna como apreciamos en la Figura 57.



**Figura 57:** Mientras el *Subgrupo 1* responde a la consigna de la pieza asociada a “Plaza 19 de Noviembre”, el *Subgrupo 2* continua visualizando ese mismo POI en su mapa (Izq: Subgrupo 1, Der: Subgrupo 2)

Una vez dada la respuesta correspondiente, por parte del *Subgrupo 1*, se le presenta a ambos subgrupos el siguiente POI en el recorrido, que para este caso es “Plaza Moreno”. En la Figura 58 se puede observar cómo ambos subgrupos reciben el POI “Plaza Moreno” en sus pantallas. En este caso, el *Subgrupo 2* es avisado del cambio de destino.

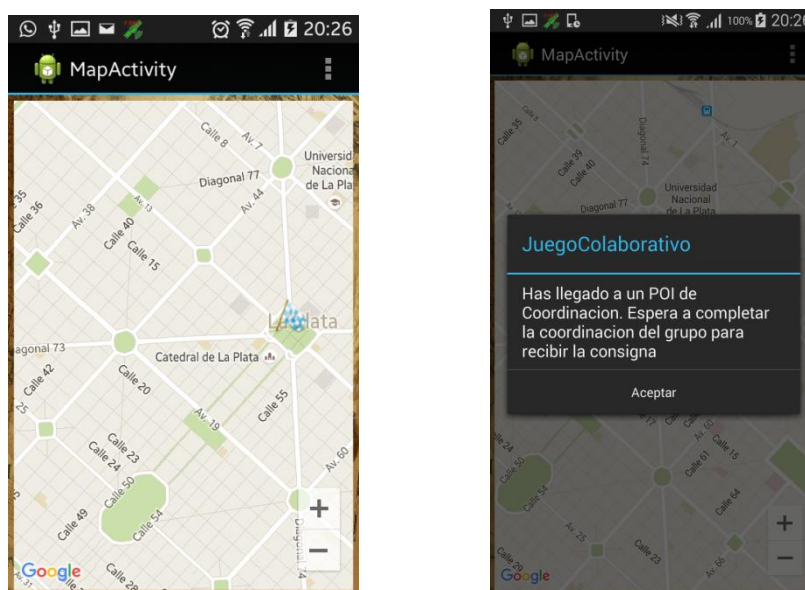


**Figura 58:** El *Subgrupo 1* visualiza el próximo POI tras dar la respuesta. El *Subgrupo 2* es notificado que el destino ha cambiado y visualiza el próximo destino “Plaza Moreno” (Izq: Subgrupo 1, Der: Subgrupo 2)



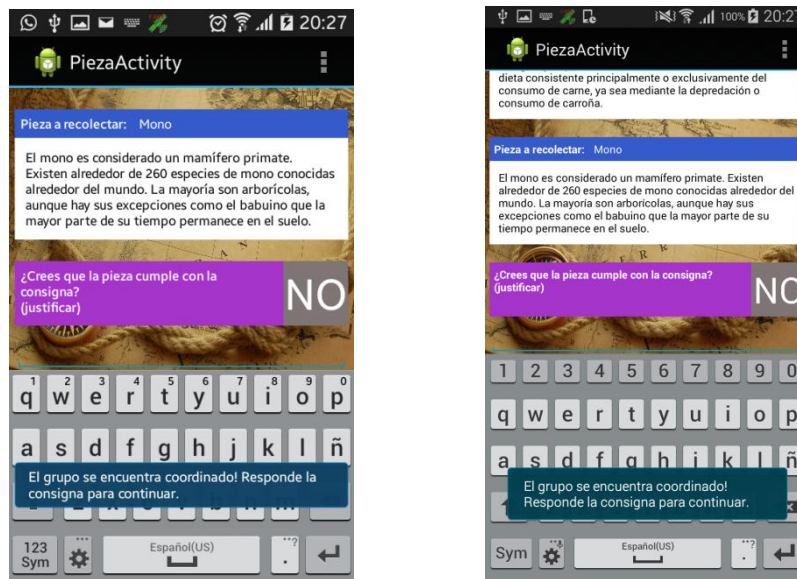
Recordemos que el tercer POI en el recorrido “*Plaza Moreno*” es un POI de *Coordinación*. Supongamos que en esta ocasión el primer subgrupo en arribar es el *Subgrupo 2*. El POI al detectar la presencia del *Subgrupo 2*, le envía un mensaje a éste informándole que se encuentra en un POI de *Coordinación* y que debe esperar a que el grupo esté coordinado para poder continuar. Para esta primera simulación utilizamos la estrategia de coordinación “*TodosEnPOI*”, por lo tanto el juego va a considerar coordinado al *Grupo Rojo*, recién cuando tanto el *Subgrupo 1* como el *Subgrupo 2* estén en la ubicación del POI “*Plaza Moreno*”.

Esta situación es transparente para el *Subgrupo 1*, es decir; mientras el *Subgrupo 2* se queda a la espera de que se complete la coordinación en “*Plaza Moreno*”, el *Subgrupo 1* continúa caminando acorde al mapa en la búsqueda de “*Plaza Moreno*”. Este escenario se puede ver reflejado en las pantallas correspondientes de cada subgrupo en la Figura 59.



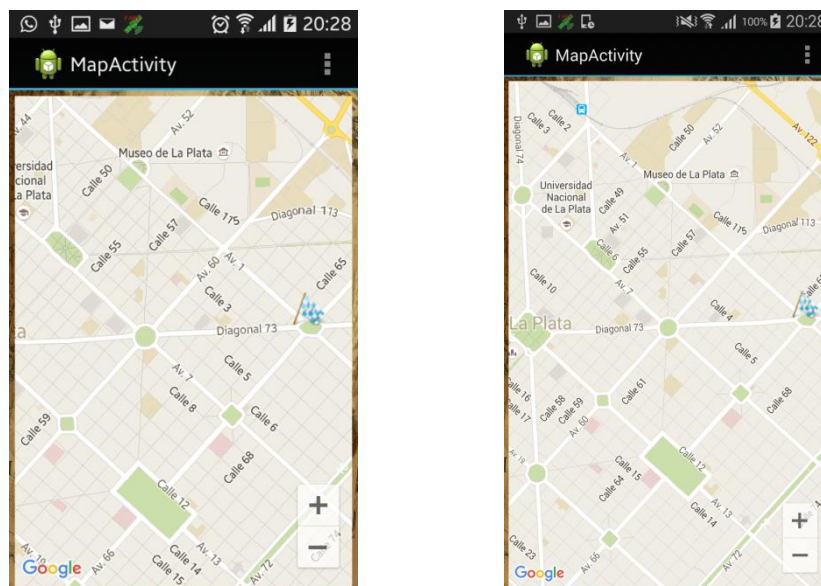
**Figura 59:** El *Subgrupo 1* continúa caminando para llegar a “*Plaza Moreno*” mientras que el *Subgrupo 2*, al ya haber arribado a dicho POI es notificado de la espera para poder continuar (Izq: *Subgrupo 1*, Der: *Subgrupo 2*)

Continuando con la simulación, supongamos que el *Subgrupo 1* llega a “*Plaza Moreno*” y tras ser detectado por el POI, el juego notifica a ambos subgrupos que el grupo ya se encuentra coordinado y les presenta la consigna a ambos para que puedan responder y continuar. Esto se puede apreciar en la Figura 60. Cualquiera de los dos subgrupos puede responder la consigna (se toma válida la primera respuesta recibida) y al responderla, se obtiene el próximo POI para los subgrupos del *Grupo Rojo*.



**Figura 60: Ambos subgrupos son notificados que la coordinación se completó y se les presenta la pieza para que den su respuesta (Izq: Subgrupo 1, Der: Subgrupo 2)**

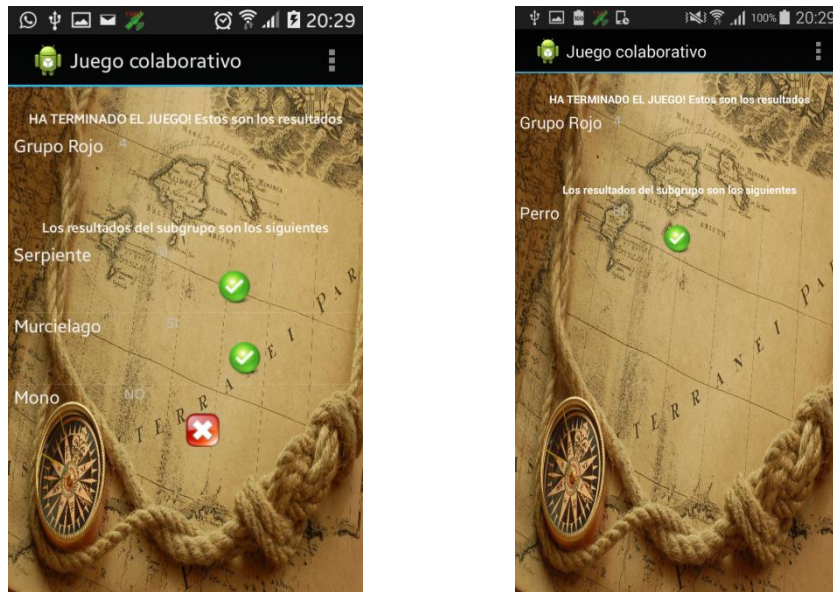
Para esta simulación, supongamos que el *Subgrupo 1* fue el primero en responder. Tras dar una respuesta para la pieza asociada a “*Plaza Moreno*”, se les presenta a ambos subgrupos el POI “*Plaza Matheu*”. En las pantallas de la Figura 61 se visualiza el último POI de este recorrido.



**Figura 61: Presentación de “Plaza Matheu” para ambos subgrupos (Izq: Subgrupo 1, Der: Subgrupo 2)**

Finalmente, para el último POI del recorrido suponemos nuevamente que el *Subgrupo 2* arriba primero a este POI, e inmediatamente se le presenta la consigna (dado que estamos ante la presencia de un POI *Común*). Mientras tanto el *Subgrupo 1*





**Figura 63: Presentación de resultados para Subgrupo 1 y Subgrupo 2**  
(Izq: Subgrupo 1, Der: Subgrupo 2)

- **Segunda Simulación: Foco en el Grupo Azul**

En esta simulación nos enfocaremos en el juego del *Grupo Azul*, recordemos que está conformado por los *Subgrupo 3* y *Subgrupo 4*. La estrategia de coordinación a utilizar en esta ocasión es “*PorcentajeEnPOI*” y el valor seteado de porcentaje para considerar coordinado a un grupo en un *POI de Coordinación* es cincuenta (50). Es decir, el juego considera coordinado a un grupo en un *POI de Coordinación* cuando la mitad de los subgrupos (del grupo) se encuentra en el *POI*. Teniendo en cuenta estas consideraciones comenzamos con la segunda simulación.

Para comenzar, como podemos apreciar en la Figura 64, ambos subgrupos se loguean en el juego desde sus respectivos dispositivos. Para esta ocasión, tenemos a la izquierda de la figura la pantalla correspondiente al *Subgrupo 3* y a la derecha de la misma, la correspondiente a *Subgrupo 4*. Esta forma de presentar ambas pantallas se conserva en las subsiguientes figuras.

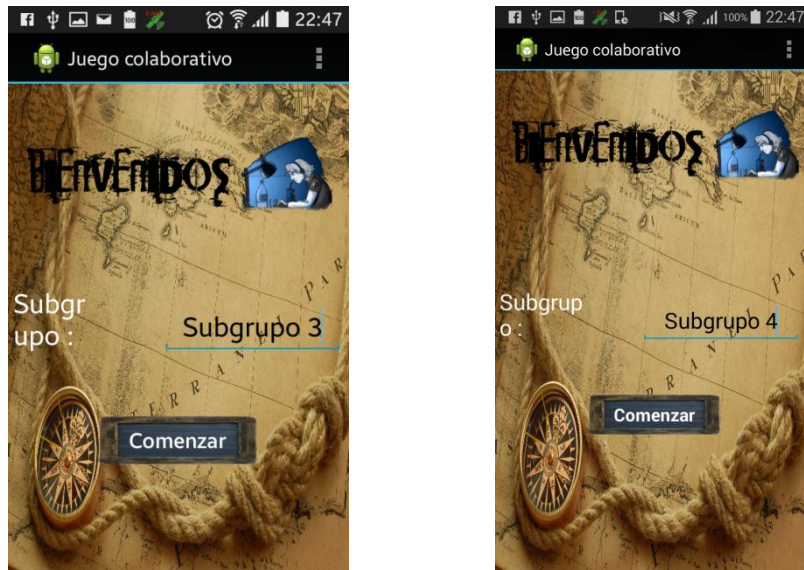


Figura 64: Login de los subgrupos de *Grupo Azul* (Izq: Subgrupo 3, Der: Subgrupo 4)

Tras efectuarse las validaciones correspondientes, cada subgrupo visualiza en su pantalla el primer POI en el recorrido del *Grupo Azul*. De esta manera, vemos en la Figura 65, que tanto el *Subgrupo 3* como el *Subgrupo 4* reciben el POI “*Parque Saavedra*” como primer destino.

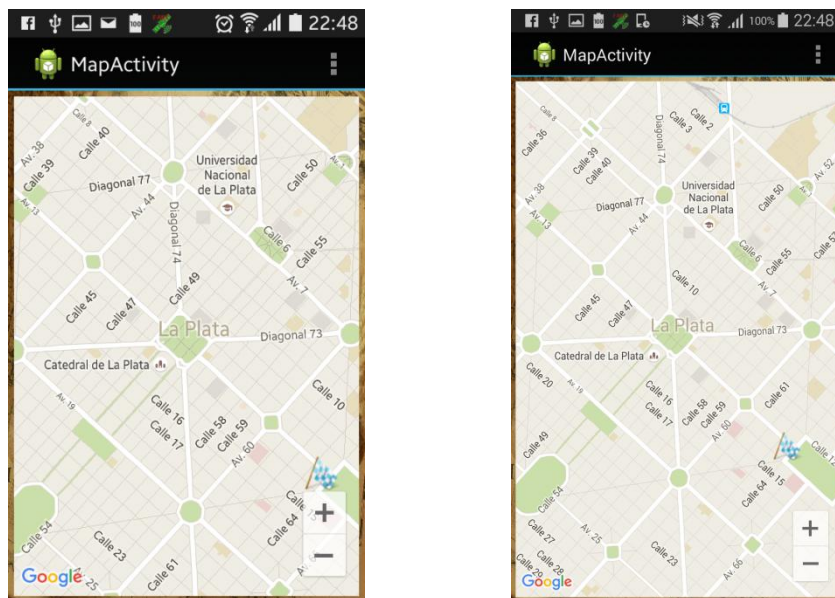
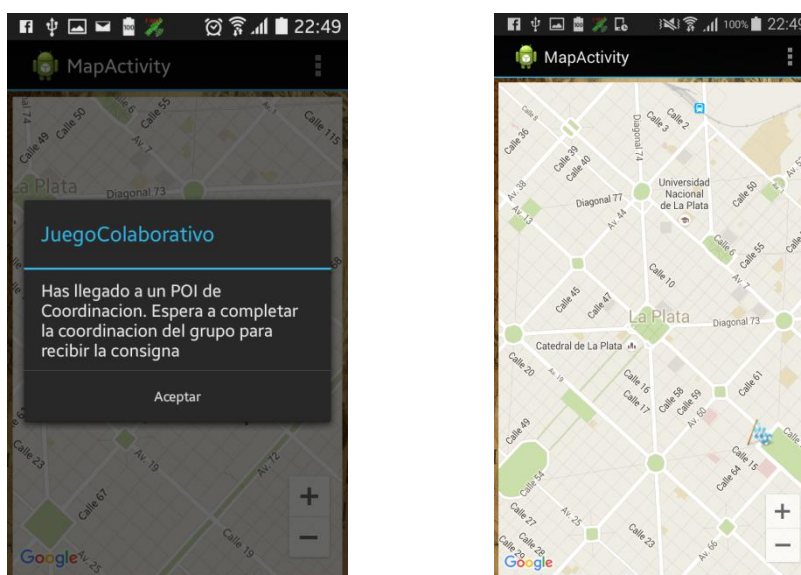


Figura 65: “*Parque Saavedra*” se muestra como el primer POI del recorrido para ambos subgrupos (Izq: Subgrupo 3, Der: Subgrupo 4)

Con el primer POI ya presentado, ambos subgrupos se desplazan por el terreno de juego en busca de “*Parque Saavedra*”. Recordemos que éste es un POI de

*Coordinación* y por tal motivo, requiere que se complete la coordinación del grupo para poder responder la consigna y continuar.

Supongamos que el *Subgrupo 3* es el primero en arribar a “*Parque Saavedra*”. El POI al detectar la presencia de este subgrupo le notifica que se encuentra en un POI de *Coordinación* y que debe esperar a que el grupo este coordinado para poder continuar con el juego. Recordemos que para esta segunda simulación utilizamos la estrategia “*PorcentajeEnPOI*”, con un porcentaje del 50%. Es decir, el grupo va a considerarse coordinado con la presencia en el POI de la mitad de los subgrupos del grupo. Para nuestro caso, con la sola presencia del *Subgrupo 3* o del *Subgrupo 4* ya se considera al grupo coordinado para el POI de *Coordinación*. En la Figura 66, vemos el mensaje que recibe el *Subgrupo 3* al llegar a “*Parque Saavedra*” y además vemos que el mapa que ve el *Subgrupo 4* que no sufre alteraciones.



**Figura 66:** El *Subgrupo 3* llega a “*Parque Saavedra*” y se le notifica que debe esperar a que la coordinación del grupo se complete para continuar. Mientras tanto, el *Subgrupo 4* sigue caminando al “*Parque Saavedra*”

**(Izq: Subgrupo 3, Der: Subgrupo 4)**

Tras hacerse las validaciones correspondientes, el *Juego* deduce que con la sola presencia del *Subgrupo 3* en “*Parque Saavedra*” el grupo ya se encuentra coordinado. Por lo tanto, vemos en la Figura 67, cómo ambos subgrupos son notificados que la coordinación es exitosa al mismo tiempo que se les presenta la pieza asociada para que puedan responder a la consigna. En este caso, el juego va a tomar la primera respuesta que reciba de cualquiera de los dos subgrupos como la válida.

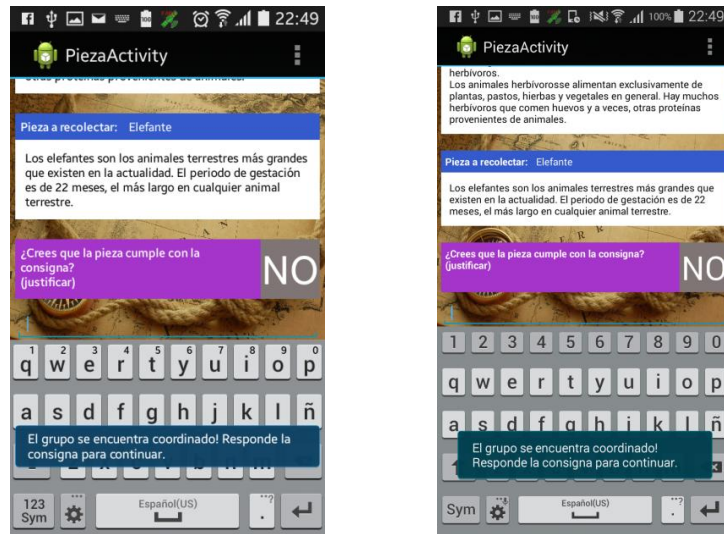


Figura 67: El Juego, tras hacer las validaciones, comprueba que el grupo está coordinado para el POI “Parque Saavedra” y presenta la pieza asociada tanto al Subgrupo 3 como al Subgrupo 4 (Izq: Subgrupo 3, Der: Subgrupo 4)

Supongamos que para “Parque Saavedra” la respuesta fue provista por el Subgrupo 3. El juego, tras procesar la respuesta, obtiene el segundo POI en el recorrido para el Grupo Azul. De esta manera, el POI “Plaza Sarmiento” es presentado a ambos subgrupos y lo ven en la pantalla de sus dispositivos como se puede apreciar en la Figura 68.

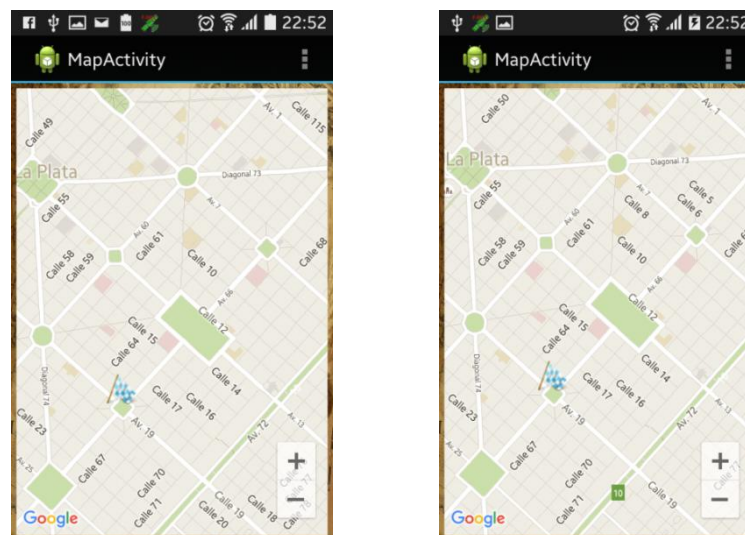
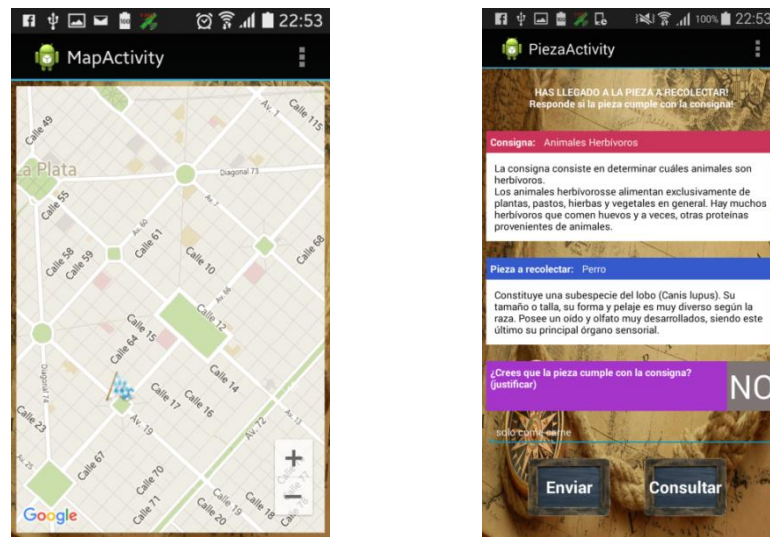


Figura 68: El segundo POI en el recorrido, “Plaza Sarmiento”, es presentado a ambos subgrupos (Izq: Subgrupo 3, Der: Subgrupo 4)

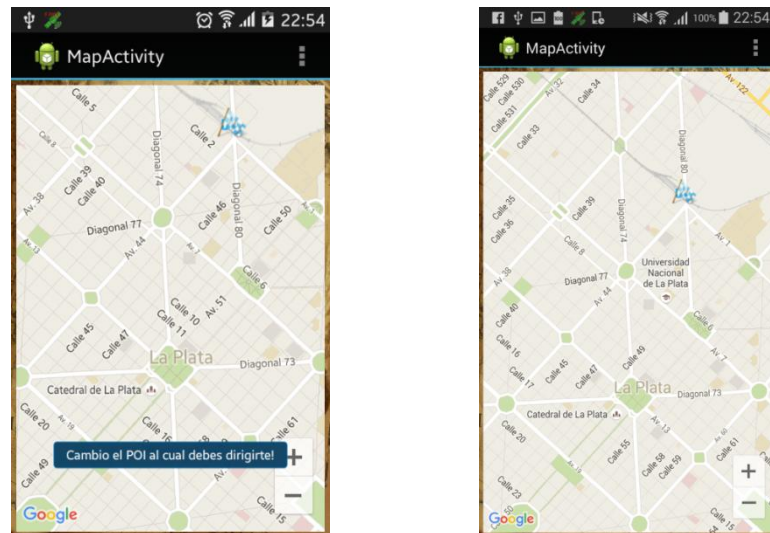
Para el POI “Plaza Sarmiento”, suponemos que el Subgrupo 4 llega en primer lugar, y al tratarse éste de un POI Común, inmediatamente se le presenta la pieza

asociada para que pueda responder. Esto se puede apreciar en la Figura 69.



**Figura 69: Subgrupo 3 sigue viendo en el mapa a “Plaza Sarmiento”, mientras que a Subgrupo 4 se le presenta la pieza asociada a dicho POI, por haber ya alcanzado dicha posición (Izq: Subgrupo 3, Der: Subgrupo 4)**

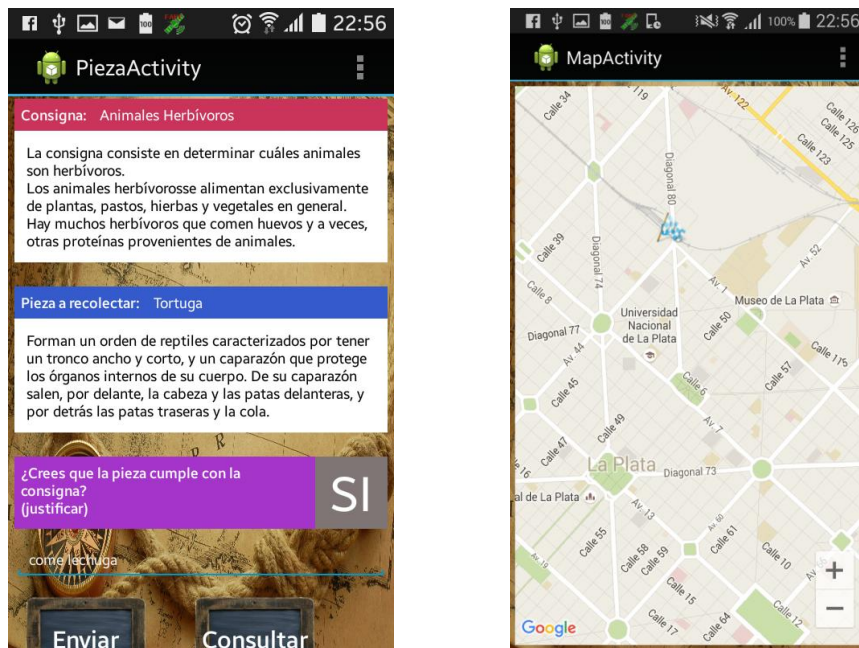
Una vez que el *Subgrupo 4* da la respuesta, nuevamente el *Juego* obtiene el siguiente POI del recorrido. El tercer POI en el camino es “Estación de Trenes” y se presenta a ambos subgrupos, como se observa en la Figura 70.



**Figura 70: El Subgrupo 3 que se estaba dirigiendo a “Plaza Sarmiento” es notificado que cambió el destino al cual debe dirigirse y visualiza el nuevo destino “Estacion de Trenes”. El Subgrupo 4 visualiza inmediatamente el POI “Estacion de Trenes” tras haber dado la respuesta para la pieza asociada a “Plaza Sarmiento” (Izq: Subgrupo 3, Der: Subgrupo 4)**

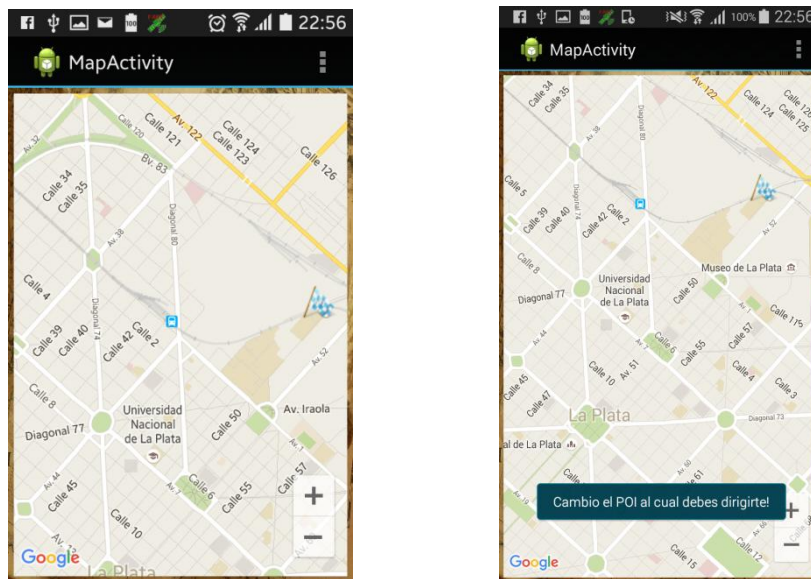


A continuación, ambos subgrupos se desplazan en búsqueda del POI “Estación de Trenes”. Suponemos que para este POI, el *Subgrupo 3* es el primero en llegar. Al ser “Estación de Trenes” también un POI Común, el *Subgrupo 3* recibe inmediatamente información sobre la pieza asociada para que pueda dar una respuesta (ver Figura 71).



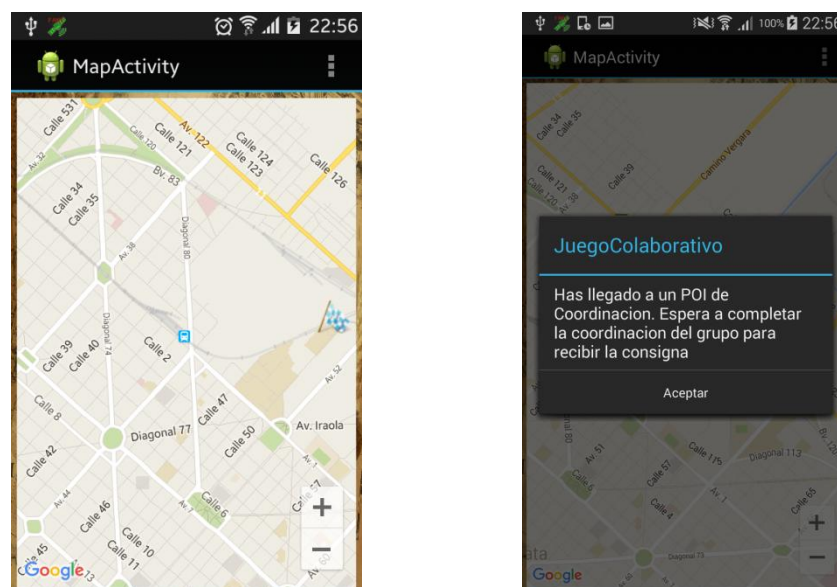
**Figura 71: El Subgrupo 3 llega a “Estacion de Trenes” y recibe inmediatamente información sobre la pieza asociada para responder. El Subgrupo 4, mientras tanto, continua buscando ese mismo POI (Izq: Subgrupo 3, Der: Subgrupo 4)**

Continuando con la simulación, el juego procesa la respuesta provista por el *Subgrupo 3* para la pieza asociada a “Estación de Trenes” y obtiene el cuarto y último POI para ambos subgrupos, siendo éste el POI de Coordinación “Facultad de Informática”. Este último POI, se les presenta a ambos subgrupos tal como podemos apreciarlo en la Figura 72. El *Subgrupo 3* (que es quién dio la última respuesta para una pieza) simplemente visualiza en el mapa el último POI del recorrido mientras que a *Subgrupo 4* se le notifica del cambio del POI destino al mismo tiempo que ve en su mapa el cuarto y último POI del recorrido.



**Figura 72: El Subgrupo 3 visualiza inmediatamente el último POI, mientras que al Subgrupo 4, se le notifica del cambio de destino al presentársele “Facultad de Informática” (Izq: Subgrupo 3, Der: Subgrupo 4)**

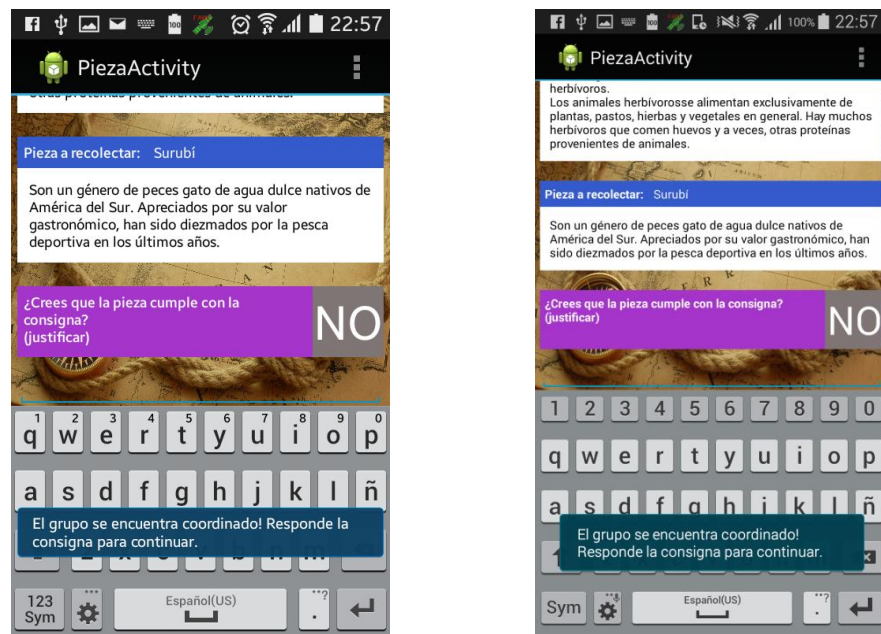
Finalmente, supongamos que el primero en arribar a este último POI es el Subgrupo 4. El POI, al detectar la presencia del Subgrupo 4, le envía un mensaje notificándole que debe esperar a que la coordinación del grupo se complete para poder continuar. Mientras tanto, el Subgrupo 3, continua visualizando a “Facultad de Informática” en su mapa. Esta situación se ve reflejada en la Figura 73.



**Figura 73: El Subgrupo 3 sigue caminando a la “Facultad de Informática” mientras que el Subgrupo 4 es notificado que debe esperar a que se complete la coordinación para continuar (Izq: Subgrupo 3, Der: Subgrupo 4)**

Nuevamente recordemos que para esta segunda simulación utilizamos la estrategia de coordinación “PorcentajeEnPOI” con un porcentaje del 50%. Esto implica que con la sola presencia de uno de los subgrupos en el POI, el criterio de coordinación se cumple para el mismo.

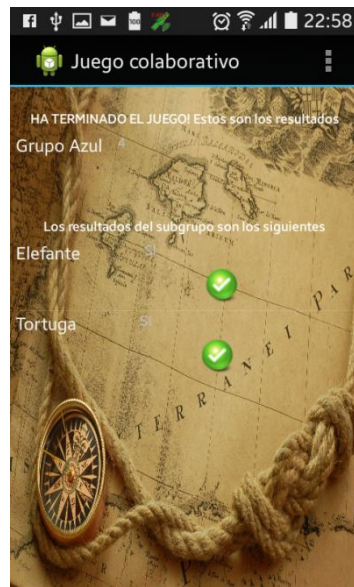
Luego, con la sola presencia del *Subgrupo 4* el juego considera coordinado al grupo y presenta la pieza asociada al POI “*Facultad de Informática*” tanto al *Subgrupo 3* como al *Subgrupo 4*, para que estos den la respuesta correspondiente. Esto se puede visualizar en la Figura 74.



**Figura 74: Se presenta la última pieza, asociada a “Facultad de Informática” a los dos subgrupos (Izq: Subgrupo 3, Der: Subgrupo 4)**

Cualquiera de los dos subgrupos puede responder a la consigna y el juego toma como válida la primera que se envié. Para este caso, suponemos que el *Subgrupo 4* da la respuesta para la última pieza.

Por último, el juego detecta que ya no quedan más POI por recorrer y que se han contestado todas las consignas, por lo cual procede al cálculo de los puntajes finales y los muestra en la pantalla de los dispositivos. Esto se puede apreciar en la Figura 75.



**Figura 75: Pantalla de resultados finales para los subgrupos de *Grupo Azul***  
**(Izq: Subgrupo 3, Der: Subgrupo 4)**

## 6. Conclusiones y Trabajos Futuros

En este capítulo se presentarán primero algunas conclusiones para luego mencionar algunos posibles trabajos futuros que se pueden realizar a partir de dicha tesis.

### 6.1 Conclusiones

A partir del análisis de distintos enfoques sobre la coordinación de grupos de personas y su aplicación en Juegos Móviles basados en posicionamiento; se determinaron distintas estrategias para coordinar grupos acordes a una posición determinada. A partir de esto se presentó una solución de modelado la cual se planteó usando como base el modelo presentado en [Apezteguía and Rapetti, 2014]. Este modelo usado como base fue ampliado para brindar soporte a la coordinación de grupos para este tipo de juegos. Es decir, el nuevo modelo propuesto en esta tesis contempla características de coordinación acorde a distintas estrategias. Se propusieron tres estrategias de coordinación en el modelo propuesto:

- La estrategia *TodosEnPOI* considera coordinado un grupo cuando la totalidad de los subgrupos de dicho grupo, se encuentre presente en una posición.
- La estrategia *TodosEnAreaCircundante* considera coordinado un grupo, si la totalidad de los subgrupos de dicho grupo, se encuentra presente dentro de un área física.
- La estrategia *PorcentajeEnPOI* considera coordinado un grupo en una posición, si la cantidad de subgrupos de dicho grupo representa (como mínimo) a un cierto porcentaje definido.

Es importante destacar que por simplicidad el recorrido de los subgrupos es lineal (secuencial) y predefinido para todos los subgrupos. Esto facilita la resolución de las estrategias de coordinación. Esto varía de lo propuesto en [Apezteguía and Rapetti, 2014] donde el recorrido era aleatorio (sin orden) y los subgrupos recibían la información de la posición de todos los POI al inicio. Se eligió realizar un recorrido lineal para poder acotar las posibilidades de coordinación. Dentro de este recorrido lineal hay algunos POI que requieren coordinación del grupo mientras otros POI son comunes (sin el requisito de coordinación, cuando llega alguno de los subgrupos se entrega la consigna a los mismo).

Cabe destacar que el modelo propuesto es flexible y extensible permitiendo que el

mismo pueda evolucionar en el tiempo agregándole otras características aparte de las presentadas en esta tesis. La ventaja de contar con un modelo viene dada por la reutilización de conceptos, como así también poder extenderlo fácilmente con nuevas funcionalidades. Si bien el modelo propuesto podría ser usado para la lógica de coordinación en los juegos presentados en el Capítulo 2, cabe destacar que cada uno de los juegos móviles analizados tiene una lógica de dominio particular que el modelo propuesto en esta tesis no cuenta aún. Para poder implementar los juegos móviles analizados en el Capítulo 2 con el modelo propuesto se deberían hacer las extensiones necesarias, por ejemplo, considerar estrategias de caza en el juego de los leones [Benford et al., 2005].

En base al modelo expuesto, se implementó un prototipo funcional para mostrar cómo se puede aplicar la coordinación de grupos en el contexto de un Juego Móvil basado en posicionamiento. Dicho prototipo, se implementó como una modificación del desarrollado en [Apezteguía and Rapetti, 2014]. Esto fue un desafío para nosotros al no estar familiarizados con la tecnología con la cual estaba definido el prototipo usado como base.

Se logró la coordinación de grupos conservando aspectos importantes del juego tales como; la organización de los grupos y subgrupos, el sensado de los mismos, el cálculo de los puntajes para cada grupo en base a las respuestas dadas, la interacción entre subgrupos a la hora de brindar respuestas para las consignas. La conservación de estos aspectos estaban dados ya por el prototipo propuesto en [Apezteguía and Rapetti, 2014]. En nuestro prototipo funcional se modificó la lógica relacionada al recorrido lineal y las estrategias de coordinación relacionadas algunos POI.

De esta manera los objetivos propuestos fueron abordados, el modelo propuesto fue descrito en el Capítulo 3, mientras que el prototipo funcional fue presentado en el Capítulo 4.

Se mostró en el Capítulo 5 el prototipo con distintos casos de prueba que permiten ver el funcionamiento del mismo, destacando los aspectos relevantes de la coordinación acorde a las estrategias modeladas.

## 6.2 Trabajos Futuros

A partir de nuestro trabajo destacamos los siguientes puntos como posibles extensiones o mejoras a trabajar:

- Para hacer posible la coordinación en los POI de Coordinación, propusimos una “*Estrategia de Coordinación*” asociada al Juego. Dicha estrategia, en nuestro trabajo, no varía durante todo el desarrollo del Juego y es aplicada por el mismo en cada POI de Coordinación. Una mejora posible, con respecto a este tema, es que la estrategia pueda cambiar a lo largo del *Juego* en forma dinámica, sin que se vea afectado el desarrollo del *Juego*. Por otra parte, otra posible mejora, podría ser la posibilidad de definir una estrategia de coordinación diferente en cada POI de Coordinación en lugar de que sea una única para todo el Juego.
- En el modelo propuesto, la presentación de los POI a recorrer para cada grupo se hace de a uno en uno, de manera secuencial, en la medida que se va avanzando en el recorrido; en contraste con el juego propuesto en [Apezteguía and Rapetti, 2014], donde todos los elementos (POI) a recorrer se visualizan todos de una vez al arribar al punto inicial. Sería deseable y una importante mejora, la posibilidad de poder recorrer los elementos de manera aleatoria (sin orden preestablecido) sin perder de vista los posibles problemas que puedan surgir al combinar un recorrido libre con la coordinación de grupos. También se podría extender el modelo para realizar otro tipo de recorridos por ejemplo, POI organizados dentro de un grafo, y contar con POI de coordinación dentro del mismo. Esto requiere analizar cómo se podría llevar a cabo la coordinación, ya que los subgrupos podrían ir avanzando en el juego por distintas ramas del grafo.
- Para nuestro modelo, propusimos tres clases diferentes de “*Estrategia de Coordinación*”, aunque por supuesto, la cantidad y variedad de estrategias no se limitan solo a éstas. En caso de agregarse nuevas, por un lado se deben agregar las mismas a nivel de modelado. Esto también implicará modificaciones en el prototipo, para esto es necesario llevar a cabo modificaciones en la consola de administración del servidor para contemplar las mismas y poder configurarlas. Esto podría mejorarse y permitir la configuración dinámica de una nueva estrategia.
- Para la coordinación de los grupos y correcta aplicación de las estrategias, es necesario un estado inicial del servidor para el desarrollo del juego. Al finalizar un juego, no se reinicia a dicho estado automáticamente. Esto podría implementarse mediante un script al término de una partida.

- La verificación del próximo destino para los subgrupos, el chequeo de la coordinación de los grupos en los POI de Coordinación y la ayuda colaborativa para resolver consignas; todas estas características se resuelven mediante el mismo mecanismo de barreras, que hace polling constantemente ejecutando métodos del servidor para determinar cambios de estados en la aplicación. Esta modalidad, si bien resuelve correctamente el problema presentado, es algo ineficiente porque genera una cierta sobrecarga del servidor. Sería deseable hallar una alternativa, que provea comunicación sincrónica sin que se vea afectado el rendimiento de los componentes.
- Al igual que en el prototipo tomado como base, el cliente no tiene recuperación ante caídas o fallos del sistema. Si ocurre un error, la aplicación se detiene o el dispositivo se apaga, hay que volver a iniciar el juego. Una mejora es poder brindar un mejor soporte en estos casos.
- Otro trabajo a futuro es realizar pruebas de campo para poder probar el prototipo en el lugar, para poder determinar otras situaciones que no pueden ser simuladas. Esto permitiría detectar, por ejemplo, como funciona el GPS, tiempo que tarda en que se actualice a todos los integrantes del grupo la información, etc.



## Bibliografía

- [Apezteguía and Rapetti, 2014] Apezteguía, M., & Rapetti, D. (2014). Juego educativo móvil colaborativo (Tesis de Grado, Facultad de Informática, UNLP. 2014.
- [Benford et al, 2005] Benford, S., Rowland, D., Flintham, M., Drozd, A., Hull, R., Reid, J. & Facer, K.: Life on the edge: supporting collaboration in location-based experiences. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 721-730). ACM. 2005.
- [Gamma et al, 1994] Gamma, E., Helm, R., Johnson, R., & Vlissides, J.: Design patterns: Abstraction and reuse of object-oriented design. In European Conference on Object-Oriented Programming (pp. 406-431). Springer Berlin Heidelberg. 1994.
- [Gu et al., 2009] Gu, J., He, L., Yang, J., & Lu, Z.: Location aware mobile cooperation-design and system. International Journal of Signal Processing, Image Processing and Pattern Recognition, 2(4), 49-60. 2009.
- [Lundgren, Fischer, Reeves, Torgersson; 2015] Lundgren, S., Fischer, J. E., Reeves, S., & Torgersson, O.: Designing mobile experiences for collocated interaction. In Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (pp. 496-507). ACM. 2015
- [Nova et al., 2009] Nova, N., Girardin, F., & Dillenbourg, P: The effects of mutual location-awareness on group coordination. International journal of human-computer studies, 68(7), 451-467. 2009.
- [Oulasvirta et al., 2005] Oulasvirta, A., Raento, M., & Tiitta, S.: ContextContacts: re-designing SmartPhone's contact book to support mobile awareness and collaboration. In Proceedings of the 7th international conference on Human computer interaction with mobile devices & services (pp. 167-174). ACM. 2005.
- [Paay et al., 2007] Paay, J., Kjeldskov, J., Christensen, A., Ibsen, A., Jensen, D., Nielsen, G., & Vutborg, R.: Location-based storytelling in the urban environment. In Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat (pp. 122-129). ACM. 2007
- [Ren et al., 2007] Kiesler, S., Fussell, S., & Scupelli, P.: Trajectories in multiple group coordination: a field study of hospital operating suites. In System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on (pp. 138-138). IEEE. 2007.

## Anexo A: Funcionalidad del Prototipo tomado como base

En este anexo se presentan las funcionalidades que en nuestro prototipo se toman del prototipo presentado en [Apezteguía and Rapetti, 2014]; es decir, no sufrieron ninguna modificación en nuestra implementación.

Comencemos describiendo el módulo de administración web desarrollado por [Apezteguía and Rapetti, 2014]. El primer paso para entrar al sistema es el login, que brinda una seguridad básica a la aplicación. Este se puede visualizar en la Figura A1.



**Figura A1 - Pantalla de login para ingreso al Sistema.**

Una vez logueado, la primera pantalla que vemos es el listado de los grupos dados de alta en el juego; y para cada uno de ellos se visualizan los datos principales, junto a botones que permiten acceder al detalle de los grupos, editarlos, eliminarlos o crear uno nuevo.

Veamos a continuación, la creación de nuevos grupos y subgrupos. Esta funcionalidad se presenta en una misma pantalla, lo cual simplifica la configuración al usuario. Primero se completan los campos que son propios del grupo, tales como el nombre y la consigna asociada. Luego, se configuran los subgrupos. En la Figura A2 se muestra la pantalla de creación, apreciándose cómo se completan los datos del grupo, y debajo los de los subgrupos.

Figura A2 – Alta de grupos y subgrupos.

En la Figura A3 vemos en detalle cómo se van agregando los subgrupos, para esto, se espera a que se complete su nombre.

Figura A3 – Alta de subgrupos dentro del alta de grupos.

Una vez tildado el nombre elegido, se agrega el subgrupo al grupo y aparecen las opciones para ir agregando uno a uno a los participantes que van a conformar el subgrupo recién agregado. Esto se puede apreciar en la Figura A4.

Figura A4 – Subgrupos y participantes.

Una vez creado el subgrupo, se pueden ir agregando a los participantes que lo conforman, simplemente pulsando el botón amarillo “Nuevo participante”. En la Figura A5, podemos apreciar el dialogo que se abre al seleccionar la opción mencionada. Allí se

completa el nombre y la edad del participante y finalmente se presiona sobre el botón “Crear” para que el participante sea agregado al subgrupo.

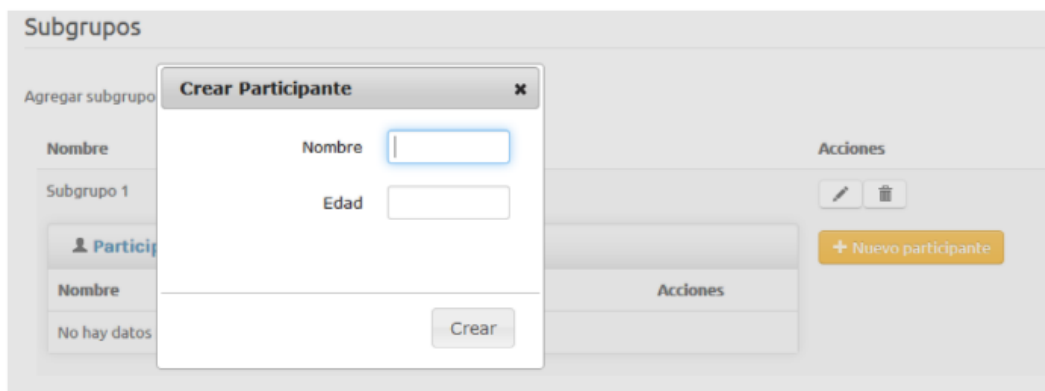


Figura A5 – Creación de participantes

Repetiendo estos mismos pasos, se van creando uno a uno los participantes que conforman a los subgrupos y los subgrupos que son parte de los grupos. De esta manera, se centraliza toda la generación de un grupo (con sus subgrupos y participantes) en una misma pantalla haciendo la carga bien dinámica. En la Figura A6 podemos apreciar un grupo creado, con sus subgrupos y los participantes de los mismos. En este ejemplo, se crea el “Grupo Azul”, con la consigna “Animales Mamíferos” y conformado por dos subgrupos, “Subgrupo 1” y “Subgrupo 2”. Asimismo, ambos subgrupos están conformados por dos participantes cada uno; “Fernando” y “Pablo” en el caso de “Subgrupo 1”, y “Fernanda” y “Lucía” para “Subgrupo 2”.

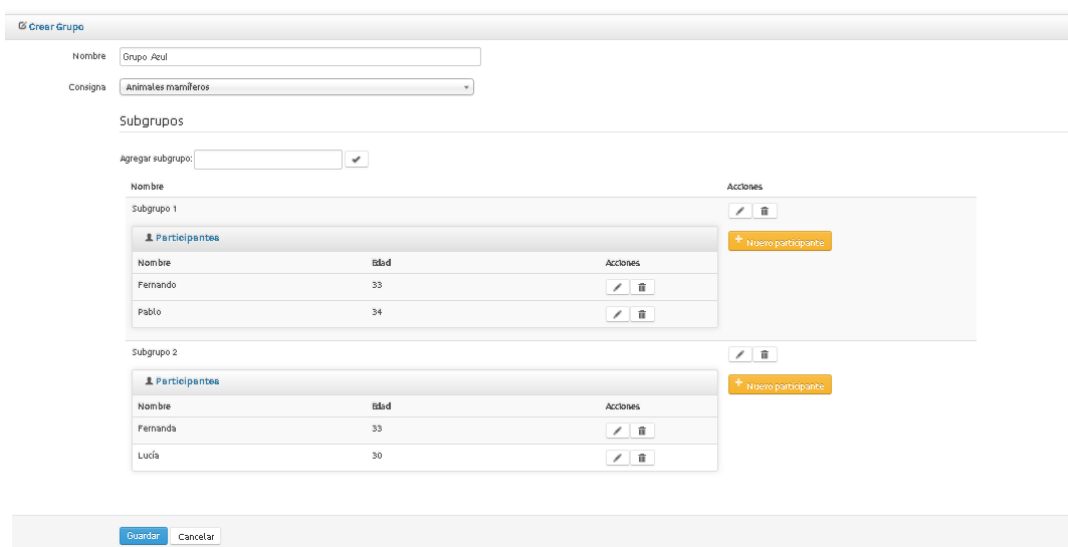


Figura A6 – Creando grupos, subgrupos y participantes.

Cabe destacar que en cualquier momento se pueden corregir datos, por ejemplo, si hubo un error en la carga de los mismos. Para ello se pueden utilizar los botones mostrados en la Figura A7.



Figura A7 – Botones para editar o eliminar subgrupos y/o participantes.

De confirmarse los datos ingresados, al volver a la pantalla principal de los grupos, el sistema avisa mediante un mensaje que no hubo inconvenientes en la operación realizada y muestra al nuevo grupo en el listado de grupos. Esto se puede apreciar en la Figura A8.

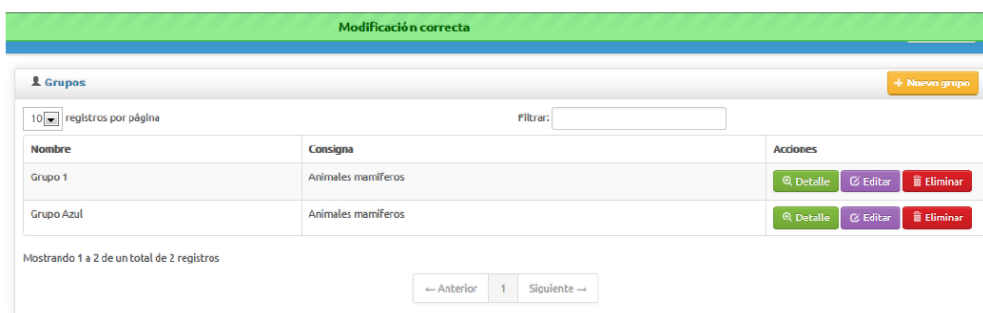


Figura A8 – Pantalla principal de los grupos con el nuevo grupo creado.

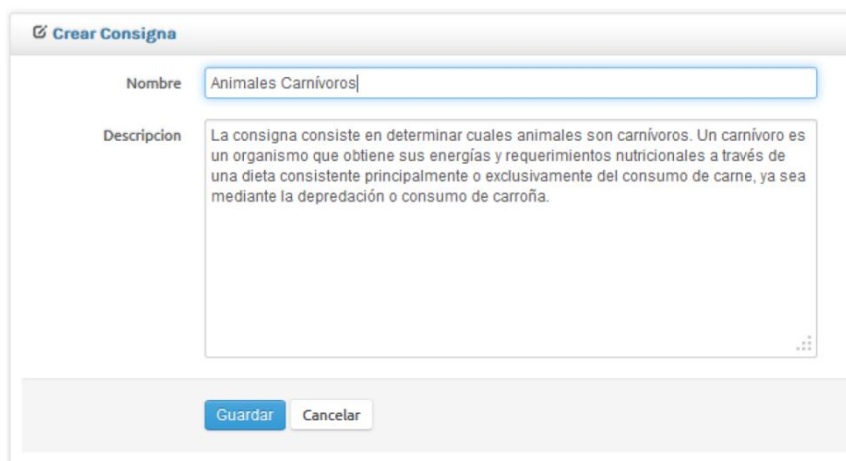
Continuemos con la administración de las consignas. Al igual que los grupos, posee una pantalla inicial conformada por una grilla que lista a todas las consignas existentes en el juego con sus datos principales. Un ejemplo de listado se puede apreciar en la Figura A9.



Figura A9 – Listado de consignas.

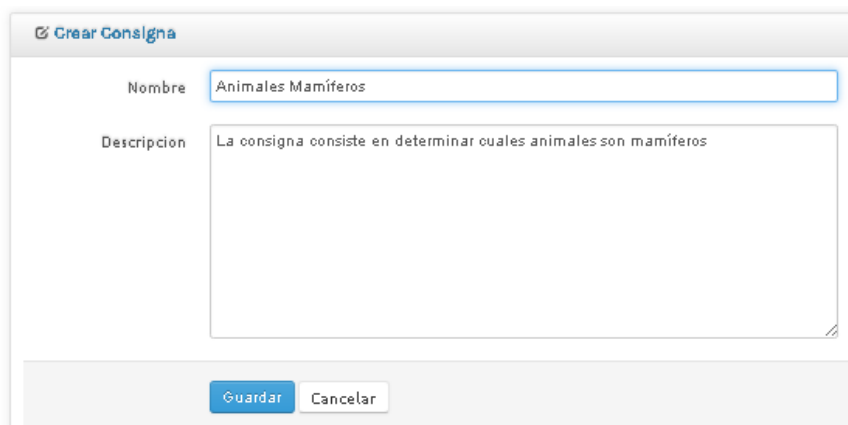
Para la creación de una nueva consigna, simplemente se elige la opción “Nueva Consigna” que abre una pantalla donde completamos los datos básicos como son “Nombre” y “Descripción”. En las Figuras A10 y A11, visualizamos dos ejemplos de creación de consignas. Como mencionamos anteriormente, son solo dos campos a completar, y una vez

finalizado, se selecciona la opción “*Guardar*” para hacer efectiva el alta de la consigna o “*Cancelar*” en el caso que de querer dejar sin efecto la operación.



The screenshot shows a web form titled "Crear Consigna". It has two main input fields: "Nombre" and "Descripción". The "Nombre" field contains the text "Animales Carnívoros". The "Descripción" field contains a paragraph of text: "La consigna consiste en determinar cuales animales son carnívoros. Un carnívoro es un organismo que obtiene sus energías y requerimientos nutricionales a través de una dieta consistente principalmente o exclusivamente del consumo de carne, ya sea mediante la depredación o consumo de carroña." At the bottom of the form, there are two buttons: "Guardar" (highlighted in blue) and "Cancelar".

**Figura A10 – Creación de consigna Animales Carnívoros.**



The screenshot shows a web form titled "Crear Consigna". It has two main input fields: "Nombre" and "Descripción". The "Nombre" field contains the text "Animales Mamíferos". The "Descripción" field contains a single line of text: "La consigna consiste en determinar cuales animales son mamíferos". At the bottom of the form, there are two buttons: "Guardar" (highlighted in blue) and "Cancelar".

**Figura A11 – Creación de consigna Animales Mamíferos.**

## Anexo B: Modificaciones de código realizadas

En este anexo se presentarán aquellas modificaciones de códigos realizadas tanto en el cliente como en el servidor respecto de lo presentado en [Apezteguía and Rapetti, 2014].

- **Comunicación entre Cliente y Servidor**

El método de comunicación entre cliente y servidor no varió respecto de lo definido en [Apezteguía and Rapetti, 2014]. Es decir, se lleva a cabo mediante *Web Services* bajo el protocolo *SOAP*. Veamos algunos detalles de este funcionamiento para comprender mejor la modificación realizada. Del lado del servidor, se utiliza un plugin de *Symfony* llamado *ckWebServicePlugin*<sup>4</sup> que facilita la construcción y configuración de los servicios. En una clase particular del servidor, se desarrollan los métodos del *Web Service* y en la firma de cada uno (en los comentarios que preceden al encabezado de un método) se definen aspectos como nombre del servicio y parámetros. Una vez que se tienen estos métodos elaborados, se ejecutan ciertos comandos del plugin que generan el *wsdl* del *Web Service* y lo exponen en una URL del *Servidor Web*.

En la Figura B1, vemos algunos de los métodos que se agregaron en nuestra extensión, en particular los relacionados a las estrategias de coordinación como

---

<sup>4</sup> <http://www.symfony-project.org/plugins/ckwebserviceplugin> (Último acceso: 1/8/2016)

```

/**
 * Actualiza la tabla de subgrupo_destino para el subgrupo pasado como parametro y sus companeros
 * @WSMethod(name='updateDestinos',webservice='WSJuegoColaborativo')
 * @param integer $idSubgrupo Identificador del subgrupo
 * @param integer $idPoi Identificador del poi destino inicial
 * @return ObjetoSimple devuelve 1 si esta todo ok, 0 de lo contrario
 */
public function executeUpdateDestinos($request) {
    $id_subgrupo = $request->getParameter('idSubgrupo');
    $id_poi = $request->getParameter('idPoi');

    $subgrupo = Doctrine::getTable('Subgrupo')->find($id_subgrupo);
    $grupo = $subgrupo->getGrupo();

    $obj = new ObjetoSimple();
    try {
        foreach ($grupo->getSubgrupo() as $subg) {
            $subgrupo_destino = Doctrine::getTable('SubgrupoDestino')->findOneBy("id_subgrupo",$subg->getId());
            $subgrupo_destino->setIdPoi($id_poi);
            $subgrupo_destino->save();
        }

        $obj->setValorInteger('1');
    } catch (Exception $e) {
        $obj->setValorInteger('0');
        $obj->setValorString($e->getMessage());
    }

    $this->result = $obj;
    return sfView::SUCCESS;
}

/**
 * Devuelve la EstrategiaDeCoordinacion que se encuentra activa
 * @WSMethod(name='getEstrategiaDeCoordinacion',webservice='WSJuegoColaborativo')
 * @return EstrategiaDeCoordinacionWS devuelve la estrategia de coordinacion activa para el juego
 */
public function executeGetEstrategiaDeCoordinacion($request) {
    $estrategia_de_coordinacion = Doctrine::getTable('EstrategiaDeCoordinacion')->findOneBy("active",1);

    $obj = new EstrategiaDeCoordinacionWS();
    $obj->setId($estrategia_de_coordinacion->getId());
    $obj->setNombre($estrategia_de_coordinacion->getNombre());
    $obj->setDescripcion($estrategia_de_coordinacion->getDescripcion());
    $obj->setArea(($estrategia_de_coordinacion->getArea()!=null)?$estrategia_de_coordinacion->getArea():-1);
    $obj->setPorcentaje(($estrategia_de_coordinacion->getPorcentaje()!=null)?$estrategia_de_coordinacion->getPorcentaje():-1);

    $this->result = $obj;
    return sfView::SUCCESS;
}

```

**Figura B1 - Algunos de los métodos nuevos que expone el *Web Service* en nuestra extensión**

Desde el cliente, la manera de consumir los servicios web también se mantiene con respecto al trabajo original. Es decir, cada vez que se quiera consumir un servicio, se crea una instancia de la clase *WSTask* (que ejecuta tareas asincrónicas en background mediante *Threads*) y se consume el servicio invocando al método `executeTask()` pasándole oportunamente el nombre del servicio y los parámetros necesarios. En la Figura B2, vemos un ejemplo de cómo se consume uno de los servicios creados para nuestro trabajo.



```

ArrayList<NameValuePair> nameValuePairs = new ArrayList<>(2);
nameValuePairs.add(new BasicNameValuePair("idGrupo", Integer.toString(getSubgrupo().getGrupo().getId())));
nameValuePairs.add(new BasicNameValuePair("idEstado", Integer.toString(getSubgrupo().ESTADO_COORDINADO)));

//Cambia el estado de los subgrupos
WSTask setJugandoTask = new WSTask();
setJugandoTask.setReferer(this);
setJugandoTask.setMethodName(SoapManager.METHOD_CAMBIAR_ESTADO_GRUPO);
setJugandoTask.setParameters(nameValuePairs);
setJugandoTask.executeTask("completeUpdateGrupoCoordinado", "errorUpdateGrupoCoordinado");

```

Figura B2- Invocación de un servicio desde un cliente

- **Sincronización de subgrupos**

Para la sincronización de los grupos utilizamos el mismo mecanismo de “barreras” de [Apezteguía and Rapetti, 2014]. En el citado trabajo, se usaban las barreras para reunir a los subgrupos en ciertos puntos particulares (punto inicial y final de cada juego) y para realizar consultas respecto a una pieza entre los subgrupos de un mismo grupo. Nosotros extendemos el uso de ellas para sincronizar a los subgrupos respecto de un cambio de POI de destino y para chequear en los POI de *Coordinación*, cuando un grupo cumple con los criterios propuestos según la estrategia de coordinación activa para el juego en curso.

Cuando se inicia el juego, dentro del proceso de inicialización del mismo, se obtiene el primer POI en el camino del subgrupo que se acaba de logear. Una vez obtenido éste, se crea un servicio llamado *PoolServiceDestinos*, que a intervalos regulares de tiempo, se encarga de verificar si el próximo destino para un subgrupo ha cambiado. De esta manera, cuando se detecta que el POI al cual debe dirigirse un subgrupo cambia, se notifica mediante un mensaje en la pantalla del dispositivo. En la Figura B3, vemos la creación del servicio *PoolServiceDestinos*. El método `completeInitDestinos()` es invocado tras haberse obtenido el primer POI en el camino del subgrupo. Si el primer POI del recorrido pudo recuperarse correctamente (en cuyo caso resultado=1), se crean los servicios *PoolServiceDestinos* y *PoolServiceEstadoCoordinado*.

```

public void completeInitDestinos(SoapObject result) {
    SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
    Integer resultado = Integer.parseInt(res.toString());

    if(resultado==1){
        ((JuegoColaborativo) getApplication()).startPoolServiceDestinos();
        ((JuegoColaborativo) getApplication()).startPoolServiceEstadoCoordinado();
    }
}

```

Figura B3- Creación de los servicios *PoolServiceDestinos* y *PoolServiceEstadoCoordinado*

*PoolServiceDestinos*, mediante un método del *Web Service*, consulta al servidor respecto del próximo POI asignado para el subgrupo y con ésta respuesta determina si su destino inmediato ha cambiado o no y también si el juego ha finalizado. En la Figura B4, vemos la composición de *PoolServiceDestinos* y destacamos el método `doServiceWork()` que realiza la invocación al *Web Service*.

```

/**
 * Created by Fernando on 2016/04/16.
 */
public class PoolServiceDestinos extends Service {

    public static final long UPDATE_INTERVAL = 5000;
    public static final long DELAY_INTERVAL = 0;

    private Timer timer = new Timer();

    public IBinder onBind(Intent intent) { return null; }

    @Override
    public void onCreate() {
        super.onCreate();
        _startService();
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        _shutdownService();
    }

    private void _startService() {
        TimerTask asynchronousTask;
        final Handler handler = new Handler();
        asynchronousTask = (TimerTask) () -> {
            handler.post(() -> {
                try {
                    doServiceWork();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            });
        };
        timer.scheduleAtFixedRate(asynchronousTask, DELAY_INTERVAL, UPDATE_INTERVAL);
    }

    private void _shutdownService() {
        if (timer != null) {
            timer.cancel();
        }
    }

    private void doServiceWork() {
        ((JuegoColaborativo) getApplication()).chequearCambioPOIDestino();
    }
}

```

```

private void doServiceWork() {
    ((JuegoColaborativo) getApplication()).chequearCambioPOIDestino();
}

```

Figura B4- Estructura del servicio *PoolServiceDestinos* e invocación del método `doServiceWork()`

El método `chequearCambioPOIDestino()` del controlador (Figura B4) es quien

en definitiva invoca al *Web Service* para conocer el próximo destino asignado para el subgrupo y pasa la respuesta obtenida del servidor mediante el método `completeChequearCambioPOIDestino()`. En la Figura B5, vemos la invocación al método que devuelve el destino inmediato asignado para el subgrupo.

```
/**
 * Método que es llamado por el PoolServiceDestinos para ver si tengo un destino nuevo asignado
 */
public void chequearCambioPOIDestino() {
    ArrayList<NameValuePair> nameValuePair = new ArrayList<>(1);
    nameValuePair.add(new BasicNameValuePair("idSubgrupo", Integer.toString(getSubgrupo().getId())));

    //El siguiente metodo de WS retorna el valor destino que tiene apuntado el subgrupo en la tabla subgrupo_destino
    WSTask esperarEstadoTask = new WSTask();
    esperarEstadoTask.setReferer(this);
    esperarEstadoTask.setMethodName(SoapManager.METHOD_CHECK_CAMBIO_DESTINO);
    esperarEstadoTask.setParameters(nameValuePair);
    esperarEstadoTask.executeTask("completeChequearCambioPoiDestino", "errorChequearCambioPoiDestino");
}
```

Figura B5- Se invoca al *Web Service* que retorna el próximo destino para el subgrupo.

Recibida la respuesta del servidor, el cliente verifica con la información proporcionada si su destino ha cambiado. Si encuentra que el destino inmediato devuelto por el servidor es distinto al actual, realiza una validación adicional para saber si se ha llegado al fin de juego. Todas estas validaciones son realizadas en el método `completeChequearCambioPoiDestino()` como se pueden apreciar en la Figura B6.

```

public void completeChequearCambioPoiDestino(SoapObject result) {
    SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
    Integer resultado = Integer.parseInt(res.toString());

    if(resultado != getSubgrupo().getDestinoProximo().getId()){

        //Fin de juego
        if(resultado.equals(-1)){
            getSubgrupo().setEstado(getSubgrupo().ESTADO_FINAL);
            CharSequence text = "Fin del Juego para tu grupo!";
            int duration = Toast.LENGTH_LONG;
            Toast toast = Toast.makeText(getApplicationContext(), text, duration);
            toast.show();

            this.finJuego();

        } else {
            getSubgrupo().setEstado(getSubgrupo().ESTADO JUGANDO);
            CharSequence text = "Cambio el POI al cual debes dirigirte!";
            int duration = Toast.LENGTH_LONG;
            Toast toast = Toast.makeText(getApplicationContext(), text, duration);
            toast.show();

            //Con el id poi recupero el objeto de la coleccion de pois asociadas a mi
            Poi poiDestino = getPoiDestinoFromPoisAVisitarCollection(resultado);
            getSubgrupo().setDestinoProximo(poiDestino);
            //Redibujar el mapa para este subgrupo y mandar un mensaje de cambio
            this.getCurrentActivity().startActivity(new Intent(this, MapActivity.class));
        }
    }
}

```

Figura B6- El cliente termina de verificar si hay un cambio de destino y/o fin de juego.

Por otra parte, tenemos dos barreras que se usan en conjunto para llevar a cabo la coordinación de un grupo en un POI de *Coordinación*. Cuando un subgrupo llega a un POI de *Coordinación*, lo primero que hace es cambiar su estado a `ESPERANDO_COORDINACION` para luego proceder a la creación del servicio *PoolServiceEstados* que es el responsable de invocar en el servidor al *Web Service* asociado a la estrategia de coordinación activa para el juego en curso. En la Figura B7 se puede apreciar la creación de dicho servicio. En la Figura B8 se puede apreciar la definición del servicio *PoolServiceEstados*.

```

public void completeSetEstadoEsperando(SoapObject result) {
    try{
        SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
        Integer resultado = Integer.parseInt(res.toString());

        //El PoolServiceEstados que se inicia mas adelante debe chequear la barrera siempre para el poi actual
        if(resultado != -1) {
            getSubgrupo().setEstado(getSubgrupo().ESTADO_ESPERANDO_COORDINACION);
            this.getCurrentActivity().startService(new Intent(getCurrentActivity(), PoolServiceEstados.class));
            //Enciendo el pool para saber cuando se termina de coordinar el grupo
            this.startPoolServiceEstadoCoordinado();
        }
    } catch (Exception e){
        Log.e("ERROR", e.getMessage());
    }
}

```

Figura B7- Se crea el servicio *PoolServiceEstados* cuando un subgrupo llega a un POI De *Coordinación*

```

/ **
 * Created by Fernando on 15/02/14.
 */
public class PoolServiceEstados extends Service {

    public static final long UPDATE_INTERVAL = 5000;
    public static final long DELAY_INTERVAL = 0;

    private Timer timer = new Timer();

    public IBinder onBind(Intent intent) { return null; }

    @Override
    public void onCreate() {
        super.onCreate();
        _startService();
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        _shutdownService();
    }

    private void _startService() {
        TimerTask asynchronousTask;
        final Handler handler = new Handler();
        asynchronousTask = (TimerTask) () -> {
            handler.post() -> {
                try {
                    doServiceWork();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        };
        timer.scheduleAtFixedRate(asynchronousTask, DELAY_INTERVAL, UPDATE_INTERVAL);
    }

    private void _shutdownService() {
        if (timer != null) {
            timer.cancel();
        }
    }

    private void doServiceWork() {
        String metodoCoordinacion = ((JuegoColaborativo) getApplication()).getEstrategiaDeCoordinacion().coordinar();
        ((JuegoColaborativo) getApplication()).esperarCoordinacionSubgrupos(metodoCoordinacion);
    }
}

```

Figura B8- Definición del servicio *PoolServiceEstados*

*PoolServiceEstados* tras su creación, invoca en el servidor al método que determina si un grupo se encuentra coordinado para un POI de *Coordinación*, de acuerdo a la estrategia de coordinación activa. Esto lo realiza en el método `doServiceWork()` en dos pasos; en primer lugar obtiene la estrategia de coordinación activa para el juego en curso, y luego invoca al método correspondiente en el servidor. Ambos procedimientos se reflejan en la Figura B9.

```

private void doServiceWork() {
    String metodoCoordinacion = ((JuegoColaborativo) getApplication()).getEstrategiaDeCoordinacion().coordinar();
    ((JuegoColaborativo) getApplication()).esperarCoordinacionSubgrupos(metodoCoordinacion);
}
}

* Método que es llamado por el PoolServiceEstados que devuelve 1 si todos los subgrupos están coordinados, 0 de lo contrario
*/
public void esperarCoordinacionSubgrupos(String metodo){
    ArrayList<NameValuePair> nameValuePair = new ArrayList<>();
    nameValuePair.add(new BasicNameValuePair("idEstado", Integer.toString(getSubgrupo().getEstado())));
    nameValuePair.add(new BasicNameValuePair("idGrupo", Integer.toString(getSubgrupo().getGrupo().getId())));

    WSTask esperarEstadoTask = new WSTask();
    esperarEstadoTask.setReferer(this);
    esperarEstadoTask.setMethodName(metodo);
    esperarEstadoTask.setParameters(nameValuePair);
    esperarEstadoTask.executeTask("completeEsperarCoordinacionSubgrupos", "errorEsperarCoordinacionSubgrupos");
}
}

```

Figura B9- Se obtiene la estrategia de coordinación activa y se verifica si el grupo está coordinado.

Recibida la respuesta del servidor, la misma se evalúa en el método `completeEsperarCoordinacionSubgrupos()` y en caso de ser afirmativa (es decir, el grupo se encuentra coordinado en su totalidad para el POI actual), el cliente procede a darle continuidad al juego o a finalizarlo, de acuerdo al estado en el que se encontraba. En la Figura B10 se puede apreciar que si el subgrupo estaba en `ESTADO_ESPERANDO_COORDINACION` le da continuidad al juego y si estaba en `ESTADO_FINAL`, se da por concluido el mismo.

```

public void completeEsperarCoordinacionSubgrupos(SoapObject result) {
    //chequeo si el valor del resultado es positivo para levantar la barrera
    SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
    int resultadoCoordinacion = Integer.parseInt(res.toString());

    if (resultadoCoordinacion == 1){
        //ahora dependiendo del estado esperado, es el metodo que debo llamar despues de ejecutar la tarea
        if (getSubgrupo().getEstado() == getSubgrupo().ESTADO_ESPERANDO_COORDINACION){
            this.continuarJuego();
        }

        if (getSubgrupo().getEstado() == getSubgrupo().ESTADO_FINAL){
            this.finJuego();
        }
    }
}
}

```

Figura B10- Obtenida la respuesta del servidor se verifica si el grupo está coordinado o no

Una vez que se reconoce al grupo como coordinado, en el método `continuarJuego()` simplemente se cambia el estado a todos los subgrupos del grupo a `ESTADO_COORDINADO`. Este cambio de estado se puede apreciar en la Figura B11. Este cambio de estado influye en el servicio `PoolServiceEstado`, cuya creación se pudo ver en la Figura B3.

```

public void continuarJuego() {
    this.removeProximityAlertPOIDeCoordinacion();
    //Levanta la barrera
    this.getCurrentActivity().stopService(new Intent(new Intent(getCurrentActivity(), PoolServiceEstados.class)));

    ArrayList<NameValuePair> nameValuePairs = new ArrayList<>(2);
    nameValuePairs.add(new BasicNameValuePair("idGrupo", Integer.toString(getSubgrupo().getGrupo().getId())));
    nameValuePairs.add(new BasicNameValuePair("idEstado", Integer.toString(getSubgrupo().ESTADO_COORDINADO)));

    //Cambia el estado de los subgrupos
    WSTask setJugandoTask = new WSTask();
    setJugandoTask.setReferer(this);
    setJugandoTask.setMethodName(SoapManager.METHOD_CAMBIAR_ESTADO_GRUPO);
    setJugandoTask.setParameters(nameValuePairs);
    setJugandoTask.executeTask("completeUpdateGrupoCoordinado", "errorUpdateGrupoCoordinado");
}

public void completeUpdateGrupoCoordinado(SoapObject result) {
    SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
    int resultadoCambioEstado = Integer.parseInt(res.toString());

    if (resultadoCambioEstado != -1) {
        for (Subgrupo subgrupo : getSubgrupo().getGrupo().getSubgrupos()) {
            subgrupo.setEstado(getSubgrupo().ESTADO_COORDINADO);
        }
    }
}

```

Figura B11- Cambio de estado de los subgrupos

La definición del servicio *PoolServiceEstadoCoordinado* se puede apreciar en la Figura B12. Este servicio, mediante una invocación al *Web Service* verifica si todos los subgrupos del grupo se encuentran en `ESTADO_COORDINADO`. Para ello, el método `doServiceWork()` invoca al mensaje `chequearGrupoEnEstadoCoordinado()` como se puede apreciar al final de la Figura B12. Si la respuesta a este chequeo es positiva, (es decir, todos los subgrupos del grupo están en `ESTADO_COORDINADO`) se levanta la barrera, y se notifica a todos los subgrupos del grupo que pueden continuar presentándoseles la pieza asociada al POI de *Coordinación* para que den la respuesta correspondiente, de acuerdo a la consigna del juego. Esto se puede apreciar en la Figura B13.

```

Created by Fernando on 2016/05/03.
public class PoolServiceEstadoCoordinado extends Service {

    public static final long UPDATE_INTERVAL = 5000;
    public static final long DELAY_INTERVAL = 0;

    private Timer timer = new Timer();

    public IBinder onBind(Intent intent) {return null;}

    @Override
    public void onCreate() {
        super.onCreate();
        _startService();
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        _shutdownService();
    }

    private void _startService() {
        TimerTask asynchronousTask;
        final Handler handler = new Handler();
        asynchronousTask = (TimerTask) () -> {
            handler.post(() -> {
                try {
                    doServiceWork();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            });
        };
        timer.scheduleAtFixedRate(asynchronousTask, DELAY_INTERVAL, UPDATE_INTERVAL);
    }

    private void _shutdownService() {
        if (timer != null) {
            timer.cancel();
        }
    }

    private void doServiceWork() {
        ((JuegoColaborativo) getApplication()).chequearGrupoEnEstadoCoordinado();
    }
}

```

```

private void doServiceWork() {
    ((JuegoColaborativo) getApplication()).chequearGrupoEnEstadoCoordinado();
}

```

Figura B12- Definición del servicio *PoolServiceEstadoCoordinado*

Como se menciono anteriormente, se verifica si todos los subgrupos del grupo se encuentran en ESTADO\_COORDINADO. Si la respuesta a este chequeo es positiva, (es decir, todos los subgrupos del grupo están en ESTADO\_COORDINADO) se levanta la barrera, y se notifica a todos los subgrupos del grupo que pueden continuar presentándoseles la pieza asociada al POI de *Coordinación* para que den la respuesta correspondiente, de acuerdo a la consigna del juego. Esto se puede apreciar en la Figura B13.



```

/**
 * Metodo que es llamado por el PoolServiceEstadoCoordinado para chequear si el estado de todos los subgrupos de un grupo
 * es coordinado (4)
 */
public void chequearGrupoEnEstadoCoordinado(){
    ArrayList<NameValuePair> nameValuePair = new ArrayList<>(2);
    nameValuePair.add(new BasicNameValuePair("idGrupo", Integer.toString(getSubgrupo().getGrupo().getId())));
    nameValuePair.add(new BasicNameValuePair("idEstado", Integer.toString(getSubgrupo().ESTADO_COORDINADO)));

    WSTask checkGrupoCoordinadoTask = new WSTask();
    checkGrupoCoordinadoTask.setReferer(this);
    checkGrupoCoordinadoTask.setMethodName(SoapManager.METHOD_ESPERAR_ESTADO_SUBGRUPOS);
    checkGrupoCoordinadoTask.setParameters(nameValuePair);
    checkGrupoCoordinadoTask.executeTask("completeChequearGrupoCoordinado", "errorChequearGrupoCoordinado");
}

public void completeChequearGrupoCoordinado(SoapObject result) {
    SoapPrimitive res = (SoapPrimitive) result.getProperty("valorInteger");
    Integer resultado = Integer.parseInt(res.toString());

    if (resultado.equals(1)) {
        //Levanta la barrera
        this.getCurrentActivity().stopService(new Intent(new Intent(getCurrentActivity(), PoolServiceEstadoCoordinado.class)));
        this.showDialog("El grupo se encuentra coordinado! Responde la consigna para continuar.", "JuegoColaborativo");

        //Se presenta la pieza asociada al POI para que el subgrupo responda
        Poi poi = getSubgrupo().getDestinoProximo();
        this.mostrarInfoPieza(poi.getIdPieza());
    }
}

```

Figura B13- Se levanta la barrera para el grupo y se presenta la pieza asociada

- **Utilización de mapas**

En nuestra extensión utilizamos las mismas herramientas que ese usaron en [Apezteguía and Rapetti, 2014] para la visualización de mapas.

En lo que es el módulo de *Administración Web*, los mapas colaboran para la configuración de los POI y de los caminos secuenciales que deben seguir los subgrupos. El plugin utilizado para tal fin es un plugin de *jQuery*<sup>5</sup> para *GoogleMaps*<sup>6</sup> denominado *Gmap3*<sup>7</sup>. Por el lado del cliente, se utilizan también las librerías provistas por *GoogleMaps* para *Android* para el uso de mapas en este tipo de aplicaciones.

- **Utilización de sensores de proximidad**

La utilización de sensores de proximidad sigue siendo tan importante en nuestra extensión como en [Apezteguía and Rapetti, 2014]. La forma de simular la detección de un subgrupo (por parte de un POI) no varía pero cuenta con algunas adaptaciones para la

<sup>5</sup> Página de jQuery: <https://jquery.com> (Ultimo acceso: 1/8/2016)

<sup>6</sup> Página de GoogleMaps: <https://maps.google.com> (Ultimo acceso: 1/8/2016)

<sup>7</sup> Página de Gmap3: <https://gmap3.net> (Ultimo acceso: 1/8/2016)

resolución de nuestro problema particular.

En primer lugar, al momento de agregarse el alerta de proximidad, se especifica para qué tipo de POI es el sensor, tal como se puede apreciar en la Figura B14.

```
//Tomo el poi destino del subgrupo y lo muestro en el mapa
Poi proximoDestino = subgrupo.getDestinoProximo();
this.getGoogleMap().addMarker(
    new MarkerOptions().position(new LatLng(proximoDestino.getCoordenadas().getLatitud(), proximoDestino.getCoordenadas().getLongitud()))
        .title(proximoDestino.getReferencia())
        .icon(BitmapDescriptorFactory.fromResource(R.drawable.start_flag));

if (proximoDestino.isDeCoordinacion()) {
    addProximityAlert(proximoDestino, PROX_ALERT_POI_COORDINACION, 0);
} else {
    addProximityAlert(proximoDestino, PROX_ALERT_POI_COMUN, 0);
}
```

Figura B14 – Al momento de agregar el sensor al POI se discrimina el tipo de POI

Luego, en el método `onReceive()` de la clase `ProximityIntentReceiver` (que es la encargada de tomar una acción cuando detecta un dispositivo dentro del radio del sensor) decide su accionar en base al tipo de POI en que se encuentra localizado dicho sensor. Esto lo podemos apreciar en la Figura B15, donde se invoca al método `enviarJugandoEnPoiComun()` en el caso de tratarse de un POI *Común* o al método `enviarJugandoEnPoiDeCoordinacion()` si se trata de un POI de Coordinación.

```
@Override
public void onReceive(Context context, Intent intent) {
    String key = LocationManager.KEY_PROXIMITY_ENTERING;
    int id = intent.getIntExtra("id", -1);
    Boolean entering = intent.getBooleanExtra(key, true);
    if (entering) {
        if (intent.getAction() == MapActivity.PROX_ALERT_POI_COMUN) {
            this.getApplication().enviarJugandoEnPoiComun();
        }
        if (intent.getAction() == MapActivity.PROX_ALERT_POI_COORDINACION) {
            this.getApplication().enviarJugandoEnPoiDeCoordinacion();
        }
    } else {
        Log.e("location", "exiting");
    }
}
```

Figura B15 – El sensor toma diferentes acciones de acuerdo al tipo de POI donde esté localizado