UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Título: Creación dinámica de funcionalidad en Aplicaciones Móviles basadas en Posicionamiento.
Autores: Juan Ignacio Tonelli.
Director: Dra. Cecilia Challiol.
Codirector: Dra. Silvia Gordillo.
Carrera: Licenciatura en Sistemas

Resumen

Las Aplicaciones Móviles basadas en Posicionamiento son aquellas que se focalizan en dar soporte a experiencias de la vida real del usuario mediante un dispositivo móvil. En este contexto, la posición del usuario es de suma importancia para definir el comportamiento de la aplicación; es por esto que es necesario utilizar algún mecanismo que permita obtener dicha posición, para luego poder actuar de manera acorde. En la actualidad, los dispositivos móviles (teléfonos celulares, tabletas, etcétera) poseen, en su mayoría, sensores que facilitan esta tarea; ejemplos de esto son los sensores GPS, la triangulación por WiFi, o la lectura de códigos QR.
En este trabajo se propone un enfoque para la generación dinámica de la funcionalidad presente en las Aplicaciones Móviles basadas en Posicionamiento. Dicho enfoque intentar solucionar la problemática actual de la carencia de un modelo aplicable de manera genérica debido a las variaciones que afectan la funcionalidad de cada caso particular, como consecuencia de la gran variedad de dominios abarcados por este tipo de aplicaciones. Para esto, se realizó un diseño de solución orientado a objetos que posee las características de escalabilidad y reusabilidad necesarias para ser aplicado de manera genérica en el campo de este tipo de aplicaciones.

Palabras Claves

Aplicaciones Móviles basadas en Posicionamiento, Funcionalidad dinámica, Diseño orientado a objetos, Escalabilidad, Reusabilidad.

Conclusiones

Como resultado de este trabajo de investigación aplicada se obtuvo un modelo de solución orientado a objetos, escalable y genérico para la definición y creación dinámica de la funcionalidad de las Aplicaciones Móviles basadas en Posicionamiento. Tanto el modelo de solución como los conceptos definidos fueron aplicados de manera práctica en prototipos, para así comprobar que cumplen con las características mencionadas.

Trabajos Realizados

- *Se realizó un relevamiento del estado del arte y la problemática actual en el campo de las Aplicaciones Móviles basadas en Posicionamiento.*
- *Se diseñó un modelo orientado a objetos genérico y escalable para dar solución a la problemática planteada.*
- *A modo de prueba empírica, se implementaron dos prototipos de Aplicación Móvil basada en Posicionamiento con funcionalidades diferentes, para poner a prueba el modelo propuesto.*

Trabajos Futuros

- *Definir templates o moldes de funcionalidad, o parte de ella, para su utilización como punto de partida en la definición de una Aplicación Móvil basada en Posicionamiento.*
- *Integrar el modelo y los conceptos definidos en esta tesis con una herramienta visual que permita la creación de funcionalidad de Aplicaciones Móviles basadas en Posicionamiento a usuarios no expertos.*
- *Implementación de prototipos de Aplicación Móvil basada en Posicionamiento en dominios específicos que puedan ampliar el conjunto de reglas provistas por el modelo.*

Fecha de la presentación: OCTUBRE 2015

Índice

1. Introducción 2
1.1 Motivación 2
1.2 Objetivo 4
1.3 Estructura de la tesis 4
2. Estado del arte 6
2.1 Aplicaciones Móviles basadas en Posicionamiento 6
2.2 Taxonomía de las Aplicaciones Móviles basadas en Posicionamiento 8
2.3 Modelos de solución existentes 12
3. Modelo propuesto 16
3.1 Conceptos principales 16
3.2 Problemática a resolver 19
3.3 Características del modelo propuesto 24
3.4 Descripción del modelo propuesto 25
4. Prototipos desarrollados 34
4.1 Generalidades de ambos prototipos 34
4.2 Tecnologías utilizadas 40
4.3 Prototipo 1 – Funcionalidad lineal 41
4.4 Prototipo 2 – Funcionalidad de grafo 48
5. Ejemplo de uso 54
5.1 Prototipo 1 – Funcionalidad lineal 54
5.2 Prototipo 2 – Funcionalidad de grafo 60
6. Publicaciones realizadas 69
7. Conclusiones y Trabajos Futuros 71
Referencias bibliográficas 73
Anexo I: Publicaciones 75

1. Introducción

1.1 Motivación

Las denominadas *Aplicaciones Móviles basadas en Posicionamiento* son aquellas que basan su comportamiento en la posición física del usuario, la cual se obtiene mediante algún mecanismo de sensado[1] como por ejemplo GPS, NFC (*Near Field Communication*), triangulación por Wi-Fi, lectura de códigos 2D, etcétera. En este tipo de aplicaciones, los contenidos brindados se asocian a posiciones físicas reales, por lo que los usuarios podrán acceder a determinado contenido sólo si se encuentran en la posición física adecuada. Teniendo en cuenta esta característica, al realizar un relevamiento de las aplicaciones existentes, se puede encontrar que la distribución y restricciones que se aplican a los contenidos varían en cada caso concreto; por ejemplo, en una aplicación utilizada en un museo, como la presentada en [Callaway et al., 2012], tiene sentido guiar a los visitantes en un recorrido específico, indicándoles a dónde dirigirse según un orden preestablecido. Sin embargo, en otras aplicaciones podría ser necesario una funcionalidad diferente, es decir, los contenidos podrían presentar otras restricciones de recorrido; como ejemplo de esto podrían tomarse aplicaciones de turismo (como por ejemplo [Metro Paris Subway]), en donde el contenido se brinda según la cercanía a la posición del usuario, pero no existe una relación de orden.

Existen actualmente trabajos en los que se intenta clasificar a las Aplicaciones Móviles basadas en Posicionamiento, e incluso intentando asociar características de funcionalidad a cada categoría; por ejemplo, en [Millard et al., 2013], los autores postulan que los dominios abarcados por este tipo de aplicaciones se pueden reducir a:

- Guías Turísticas Sensibles a la Posición (*Location Aware Tour Guides*): aplicaciones de uso turístico. Su funcionalidad suele presentar estructuras de contenido abiertas, es decir, sin un orden preestablecido.

- Herramientas Educativas Sensibles a la Posición (*Location Aware Educational Tools*): aplicaciones que dan soporte a un proceso de aprendizaje. Generalmente presentan una funcionalidad en la que los contenidos son accedidos en un orden secuencial prestablecido.

- Juegos Sensibles a la Posición (*Location Aware Games*): aplicaciones con fines de entretenimiento. En este caso, la variación de funcionalidad es tan grande que no suele asociárselas a una en particular.

- Ficción Sensible a la Posición (*Location Aware Fiction*): aplicaciones utilizadas para la puesta en escena de una obra de ficción. De manera similar a los Juegos Sensibles a la Posición, los posibles tipos de funcionalidad varían según los estilos narrativos

[1] El término "*sensado"* se refiere a la acción de obtener un dato desde un sensor. Si bien dicho término no está oficialmente aceptado por la RAE, es ampliamente utilizado en el área de Computación Móvil.

utilizados.

Existen también trabajos en los que se clasifica a las aplicaciones exclusivamente por su funcionalidad o estructura de contenidos. En [Kjeldskov et al., 2007], por ejemplo, se asocian los tipos de funcionalidad presentes en las *guías móviles* con las llamadas *metáforas*, obteniendo la siguiente clasificación:

- Búsqueda del tesoro *(Treasure hunts)*: aplicaciones que presentan una funcionalidad lineal.

- Rompecabezas *(Jig-saw puzzles)*: aplicaciones en las que la funcionalidad no exige un orden en la manera en que los contenidos son visualizados, pero sí en cuanto a su totalidad.

- Dominó: aplicaciones que presentan una funcionalidad en forma de grafo con bifurcaciones, es decir, en donde existe un orden, pero, a diferencia de la *Búsqueda del tesoro*, pueden existir varios caminos para llegar al final.

- Palabras Cruzadas *(Scrabble)*: similar a la metáfora de *Rompecabezas*, pero en este caso la funcionalidad no exige la visualización de todos los contenidos.

- Recolectando mariposas *(Collecting butterflies)*: aplicaciones en las que la funcionalidad no posee ningún tipo de restricción, es decir, los contenidos son totalmente independientes entre sí.

De manera similar, en [Millard et al., 2013] se propone una clasificación de la funcionalidad presente en las llamadas *narrativas móviles*, las cuales son aplicaciones que sirven como soporte para procesos narrativos. En este caso, los tipos definidos son:

- Cañón *(Canyon)*: aplicaciones en las que la funcionalidad establece que los nodos (contenidos) se suceden de forma secuencial.

- Delta: aplicaciones en las que la "historia" posee bifurcaciones; es decir, la funcionalidad establece que los nodos (contenido y posición) poseen uno o varios posibles siguientes.

- Planicie *(Plain)*: aplicaciones en las que la funcionalidad no establece ninguna restricción en particular sobre los nodos, por lo que pueden ser accedidos en cualquier orden.

Como puede observarse por los ejemplos citados, si bien existen autores que han realizado una taxonomía sobre las características de la funcionalidad detectada en las aplicaciones, siempre se limitan a un dominio concreto (en el caso de [Kjeldskov et al., 2007], a las *guías móviles*, en el caso de [Millard et al., 2013], a las *narrativas móviles*, etcétera).

La motivación de este trabajo surge entonces como la necesidad detectada de un enfoque

más general que los adoptados actualmente, sin limitar el análisis de la funcionalidad a un dominio de aplicación específico. La dificultad de esta tarea subyace en que la gran cantidad de dominios abarcados por las Aplicaciones Móviles basadas en Posicionamiento tiene como consecuencia una gran variedad en cuanto al funcionamiento de este tipo de aplicaciones; esta es la razón por la cual, en la actualidad, los desarrolladores utilizan soluciones ad hoc o específicas para el dominio o situación particular. Es deseable entonces plantear una solución a la problemática actual de que los desarrolladores, a la hora de implementar una Aplicación Móvil basada en Posicionamiento, deben realizar el análisis y desarrollo de manera específica para cada caso particular, sin contar con un enfoque o modelo de solución existente que agilice o facilite la tarea de definir la funcionalidad de dichas aplicaciones.

1.2 Objetivo

El objetivo de este trabajo es desarrollar un modelo de solución para la creación dinámica de la funcionalidad de Aplicaciones Móviles basadas en Posicionamiento de forma genérica y escalable, de manera que sea independiente de un dominio o situación específicos. El modelo planteado deberá tener las características beneficiosas que describen las Leyes de Evolución del Software descriptas en [Lehman, 1996], como *cambio continuo*, *crecimiento continuo*, *complejidad incremental*, etcétera.

Para lograr este objetivo, será necesario realizar una investigación y relevamiento de los enfoques existentes que estén relacionados con la temática, para luego realizar un análisis de las ventajas y falencias de cada uno, y finalmente plantear un enfoque propio con las características mencionadas de generalidad y escalabilidad. Para obtener pruebas empíricas de que estas características se cumplen, será necesario realizar pruebas y desarrollar prototipos que se basen en él.

Como resultado de este trabajo, se espera obtener un modelo de solución orientado a objetos que permita crear y modificar dinámicamente la funcionalidad de cualquier Aplicación Móvil basada en Posicionamiento, independientemente de su dominio particular.

1.3 Estructura de la tesis

El presente trabajo se encuentra estructurado de la siguiente manera:

- **Capítulo 1 (este capítulo).** Donde se motiva el tema de este trabajo y se plantean los objetivos.

- **Capítulo 2.** Donde se analiza el *estado del arte*, mostrando los resultados del relevamiento de los enfoques existentes y la taxonomía de las Aplicaciones Móviles basadas en Posicionamiento.

- **Capítulo 3.** Donde se introducen los conceptos de la solución y el *modelo propuesto* y sus características, se realiza la caracterización del problema, y se detalla su desarrollo mediante diagramas de clase y de secuencia.

- **Capítulo 4.** Donde se describen los prototipos desarrollados utilizando el modelo propuesto, las tecnologías utilizadas, y sus características.

- **Capítulo 5**. Donde se muestran *ejemplos de uso*, esto es, capturas de pantalla de los prototipos puestos en marcha, y siendo utilizados desde la perspectiva de un usuario.

- **Capítulo 7.** Donde se describen las publicaciones realizadas, relacionadas con la temática de este trabajo.

- **Capítulo 7.** Donde se resumen las conclusiones del trabajo y se describen los posibles trabajos futuros.

2. Estado del arte

En este capítulo se hará un análisis de los enfoques existentes para la temática de Aplicaciones Móviles basadas en Posicionamiento desde el punto de vista teórico, así como también los modelos de solución concretos que existen.

2.1 Aplicaciones Móviles basadas en Posicionamiento

Antes de comenzar con el análisis del estado del arte, es necesario definir el concepto de *Aplicación Móvil basada en Posicionamiento* tal como lo interpretará el autor de este trabajo, dado que no existe una versión unificada de este concepto:

> Una ***Aplicación Móvil basada en Posicionamiento*** es un producto de software especialmente diseñado para dar soporte a una experiencia de la vida real del usuario mediante su dispositivo móvil.

Esta definición se asemeja a la interpretación dada en [Millard et al., 2013], donde se definen los *sistemas de hipertexto basados en posicionamiento* como *"[...] sistemas que combinan contexto físico con contenido virtual, y permite a los autores utilizar el espacio físico como un área de trabajo digital, o (desde una perspectiva diferente), aumentar el espacio físico con objetos virtuales"*[2]. Tanto por la definición del autor de este trabajo como la provista en [Millard et al., 2013], queda claro que este tipo de aplicaciones pertenece a la esfera de la Computación Móvil, definida en [Roy et al., 2003] como:

> *"[...] la computación móvil puede ser definida como aquella que ocurre mientras un usuario puede o está moviéndose, utilizando un dispositivo (generalmente portátil) que es independiente de la posición y se puede comunicar mediante un canal inalámbrico."*[3]

En cualquier Aplicación Móvil basada en Posicionamiento existen los llamados *contenidos*, que no son más que la información que se le brinda al usuario en el contexto de la experiencia. Esta información puede ser brindada en formato de texto plano, imágenes, videos o cualquier otro archivo multimedia soportado por los dispositivos móviles. La particularidad de los contenidos de este tipo de aplicaciones es que no suelen ser accedidos de manera directa por el usuario en cualquier momento, sino que están

[2] Traducción del autor de esta tesina, del original: *"Location-based hypertext systems combine physical context with virtual content, and allow authors to use a physical space as their digital canvas, or (from the opposite perspective) to augment a physical space with virtual objects."*

[3] Traducción del autor de esta tesina, del original: *"[...] mobile computing can be defined as computing that occurs whilst a user is able to move or is moving around, using a device (often handheld) directed by the user that is location independent and can communicate through a wireless channel."*

asociados a una posición física del mundo real (es por esto que estas aplicaciones se denominan "*basadas en posicionamiento"*), por lo que los usuarios podrán accederlos sólo cuando se encuentren en las posiciones adecuadas. En el contexto de este trabajo, esta relación entre una posición y el contenido que se brinda en ella se denominará *punto de interés*.

Cabe destacar que si bien existen varios aspectos de contexto posibles para las aplicaciones *sensibles al contexto* (grupo en el que se encuentran incluidas las Aplicaciones Móviles basadas en Posicionamiento), los relevamientos realizados en [Emmanouilidis et al., 2013] muestran que el dato contextual más utilizado en este tipo de aplicaciones es la *posición* del usuario. En la Figura 2.1 se muestra el gráfico resultante del análisis realizado en dicho trabajo, donde se grafica que, dentro de las categorías de información contextual (círculo *Context Categories*), los datos inherentes al usuario de la aplicación son los más utilizados (elipse *User*), y dentro de ellos, la posición física (rectángulo *Location*).

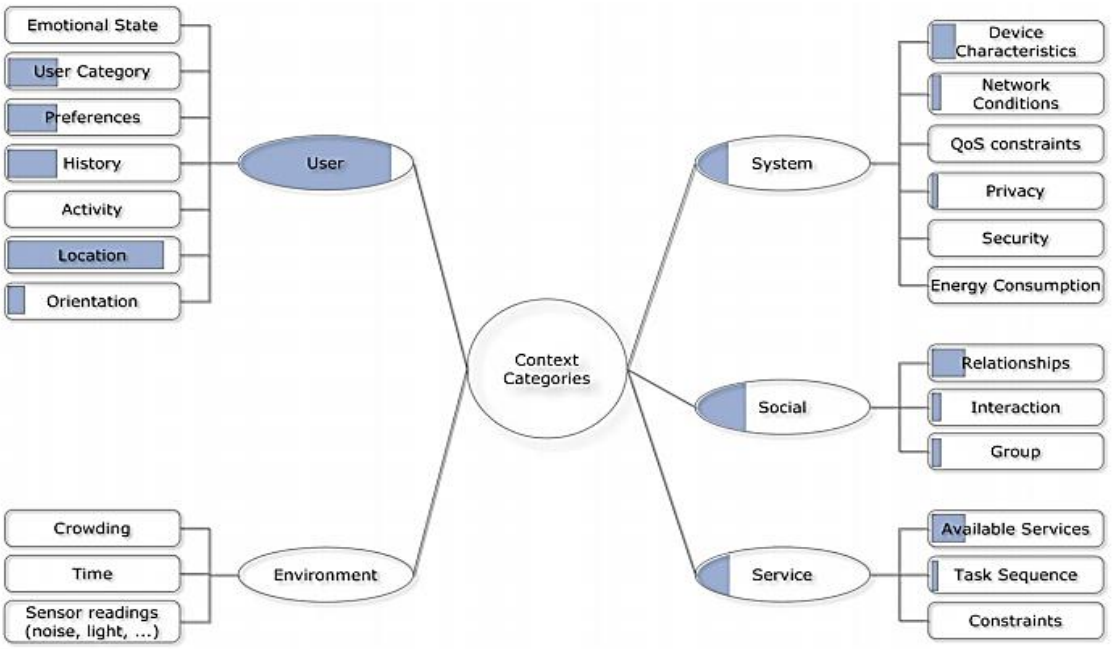

Figura 2.1 – Relevamiento realizado en [Emmanouilidis et al., 2013] sobre los aspectos de contexto más utilizados.

Para obtener la posición de los usuarios se utilizan uno o varios *mecanismos de sensado*. Estos mecanismos generalmente hacen uso de los sensores con los que cuentan los dispositivos móviles actuales para determinar la posición del dispositivo, y por asunción, la del usuario que lo está utilizando. Ejemplos de mecanismos de sensado son los sensores GPS, NFC (Near Field Communcation), triangulación por antenas, etcétera. También es interesante mencionar el caso de los Códigos QR (del acrónimo en inglés *QR Code, Quick Response Code*, es decir, *Código de Respuesta Rápida*), los cuales, según la reseña realizada en [Kato et al., 2010], forman parte de una tecnología de códigos de barras bidimensionales que sobrepasaron su función original de manejo de datos a nivel

industrial para convertirse luego en una forma de representación de datos multifacética, permitiendo, entre otras posibilidades, utilizarlo como un mecanismo de sensado; esto implica que un código QR se asocia a una posición física del mundo real, y se asume que cuando un usuario lee ese código QR, se encontrará en la posición asociada al mismo.

2.2 Taxonomía de las Aplicaciones Móviles basadas en Posicionamiento

Dada la gran cantidad de dominios abarcados por la definición del concepto de *Aplicación Móvil basada en Posicionamiento*, varios autores han realizado taxonomías teniendo en cuenta diferentes aspectos. Por ejemplo, en [Millard et al., 2013], los autores postulan que los dominios abarcados por este tipo de aplicaciones se pueden reducir a cuatro categorías:

- Guías Turísticas Sensibles a la Posición (*Location Aware Tour Guides*): el objetivo en este tipo de aplicaciones es guiar e informar al usuario en un lugar turístico. Suelen tener una estructura abierta (es decir, sus contenidos no tienen un orden preestablecido), de manera de no restringir la libertad de acción del usuario, asumiendo que el orden en que accede a los contenidos no es de importancia.

- Herramientas Educativas Sensibles a la Posición (*Location Aware Educational Tools*): en este tipo de aplicaciones, el objetivo principal es dar soporte a un proceso de aprendizaje. Si bien la estructura de este tipo de aplicaciones puede variar, por lo general los contenidos suelen accederse en un orden lineal prestablecido, de manera de tener un control más detallado del progreso de los participantes.

- Juegos Sensibles a la Posición (*Location Aware Games*): este tipo de aplicaciones se utiliza con fines de entretenimiento. La forma en que se estructuran sus contenidos varía considerablemente, por lo que no existe un patrón predominante.

- Ficción Sensible a la Posición (*Location Aware Fiction*): en este tipo de aplicaciones se utiliza la narrativa para lograr la inmersión de los participantes en una obra de ficción. Nuevamente, las estructuras utilizadas en este tipo de aplicaciones son muy variables, tanto como lo son los estilos de narrativa a los que dan soporte.

En [Emmanouilidis et al., 2013], la taxonomía propuesta se basa en la división de la arquitectura de las aplicaciones en *tiers* o *niveles*. Cada "nivel" encapsula un aspecto genérico de la aplicación; así, los tres niveles generales son: Nivel de Aplicación (*Applicacion Tier*), que define las características del software visible por los usuarios, el Nivel Intermedio (*Middle Tier*), donde se definen los protocolos de comunicación utilizados para transmitir datos de la aplicación, y Nivel de Datos (*Data Tier*), que define la forma en que la información se persiste. La taxonomía de las aplicaciones entonces se basa en esta clasificación por niveles, y la misma consiste en diferentes aspectos que forman parte de alguno de los tres niveles generales, entre los cuales se encuentran:

- Recuperación de datos (*Data retrieval*): clasifica las aplicaciones dependiendo de

la forma en que la información llega a los usuarios. Las sub-clasificaciones en este apartado son:

  - *Remota:* donde la información está almacenada fuera del dispositivo del usuario. A su vez, la recuperación remota puede ser *distribuida*, si la información se recupera desde varios servidores, o *centralizada*, si se recupera la información desde un único servidor.

  - *Local:* la información necesaria para la aplicación se encuentra almacenada en el dispositivo del usuario, por lo que su recuperación no requiere consultas a repositorios externos.

- Técnicas de localización (*Localization techniques and environment type*): clasifica las aplicaciones dependiendo del mecanismo de sensado utilizado, es decir, la técnica utilizada para obtener la posición física del dispositivo del usuario. En este apartado se pueden identificar dos tipos de técnicas de localización:

  - *Directa:* los métodos de posicionamiento *directo* producen una coordenada absoluta que identifica a la posición del usuario. Ejemplos de estos mecanismos son GPS y triangulación por antenas.

  - *Indirecta:* los métodos de posicionamiento *indirecto* permiten inferir la posición del usuario mediante elementos para los cuales el sistema posee una posición física asociada. Por ejemplo, los códigos QR son un mecanismo de posicionamiento indirecto, ya que si bien la tecnología de códigos de barra no permite explicitar la naturaleza de la información que representa, puede utilizarse como mecanismo de sensado asociando una posición a cada código QR, de manera que cuando el usuario interactúe con uno de ellos, el sistema asuma que se encuentra en la posición asociada.

- Características del dispositivo del usuario (*Client-side device characteristics and software*): clasifica las aplicaciones dependiendo de los requisitos necesarios en los dispositivos de los usuarios para utilizarla. En este sentido se pueden considerar numerosas variables, como el sistema operativo, los sensores disponibles, el soporte para determinadas librerías (por ejemplo, Java), etcétera.

- Implementación de sensibilidad al contexto (*Context aware implementations*): clasifica las aplicaciones dependiendo del soporte brindado con respecto a la sensibilidad al contexto. Esta categoría excede a las Aplicaciones Móviles basadas en Posicionamiento, ya que éstas pertenecen a la rama de las aplicaciones en donde la sensibilidad *a la posición del usuario* es la característica predominante (aunque no excluye otros posibles aspectos del contexto).

- Interfaz móvil del usuario (*Mobile user iterfaces*): clasifica las aplicaciones dependiendo del diseño y las características de la interfaz de usuario. Existen varias características que se tienen en cuenta para evaluar la facilidad de uso y claridad de una interfaz; los autores del trabajo postulan una lista de algunas de las más comunes (según los relevamientos utilizados), entre la que se pueden encontrar: uso de gestos de la pantalla táctil, barras de comando, listas, menús, pestañas de navegación, mapas interactivos, hipervínculos, reconocimiento de voz, instrucciones de audio o video, etcétera.

Finalmente, un enfoque diferente se presenta en [Kjeldskov et al., 2007], donde las aplicaciones se clasifican según la estructura utilizada para ordenar los contenidos de la misma; los autores identifican un conjunto de *metáforas*, las cuales se corresponden con determinadas estructuras de ordenamiento para los contenidos. Las mismas son:

- Búsqueda del tesoro *(Treasure hunts)*: historias lineales que se caracterizan por estar compuestas de una serie de partes o capítulos, cada uno asociado a una posición física y con un orden específico; esto implica que los usuarios deben acceder a los contenidos en el orden preestablecido para poder progresar en la experiencia.

- Rompecabezas *(Jig-saw puzzles)*: historias no lineales compuestas de un conjunto de elementos asociados a posiciones físicas, que si bien están relacionados semánticamente y en conjunto dan cohesión a la experiencia, los usuarios pueden accederlos en cualquier orden, siempre y cuando los accedan en su totalidad.

- Dominó: historias en las que cada punto o capítulo puede tener uno o varios sucesores, a diferencia de la metáfora de *búsqueda del tesoro*, donde sólo existe un único sucesor.

- Palabras Cruzadas *(Scrabble)*: variante de la metáfora del *rompecabezas* en la que no es necesario acceder a todas las piezas (contenidos asociados a posiciones físicas) para lograr coherencia en la experiencia del usuario.

- Recolectando mariposas *(Collecting butterflies)*: con esta metáfora, cada contenido es independiente del resto, tanto semántica como estructuralmente; esto significa que los usuarios de estas experiencias pueden acceder a cualquier subconjunto de los contenidos, en cualquier orden.

Cada una de estas *metáforas* representa una abstracción de una estructura de recorrido de puntos de interés y las restricciones que a este se aplican. Así, se puede interpretar a la metáfora de *búsqueda del tesoro* como a una estructura lineal de puntos de interés, donde cada uno posee sólo un sucesor; a la metáfora de *rompecabezas* como a un conjunto no ordenado de puntos de interés, en donde el usuario debe visitarlos en su totalidad para completar la experiencia; a la metáfora de *dominó* como a un grafo de puntos de interés, donde cada uno puede tener uno o varios sucesores; a la metáfora de *palabras cruzadas*

como a un conjunto sin orden de puntos de interés, donde los usuarios sólo necesitan visitar un subconjunto de todos los elementos; y finalmente a la metáfora de *recolectando mariposas* como a un conjunto no ordenado de puntos de interés, en donde cada elemento es independiente semántica y estructuralmente del resto. En la Figura 2.2 se muestra la representación gráfica de las metáforas para guías móviles, según la interpretación realizada en [Lliteras et al., 2012]:

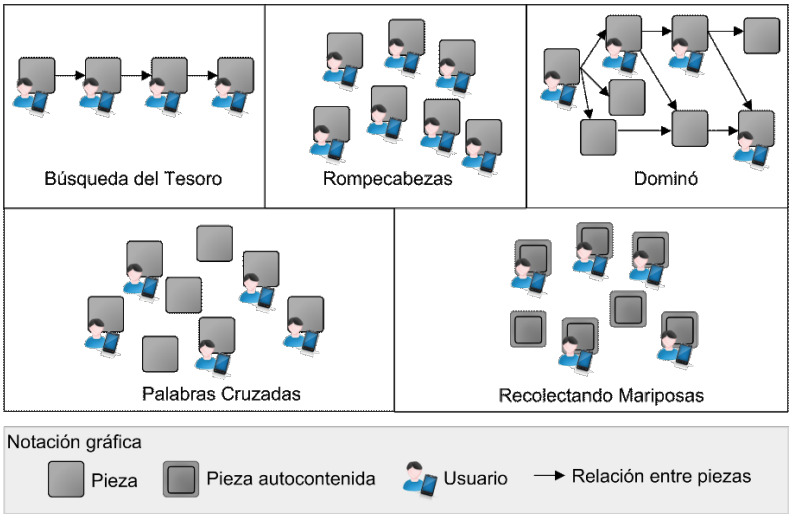

Figura 2.2 – Interpretación gráfica de las metáforas para guías móviles, presentada en [Lliteras et al., 2012].

Si bien las mencionadas metáforas se aplican específicamente a un tipo particular de aplicación (las guías móviles), esta forma de abstracción se puede trasladar a cualquier Aplicación Móvil basada en Posicionamiento. Así, una clasificación similar se utiliza en [Millard et al., 2013], en este caso en el contexto de las llamadas *narrativas móviles*, las cuales son aplicaciones que sirven como soporte para procesos narrativos. En este caso, los tipos definidos son:

- Cañón *(Canyon)*: narrativa lineal representada como una secuencia ordenada de nodos, donde cada nodo es un elemento narrativo asociado a una posición física.
- Delta: narrativa con bifurcaciones; es decir, nodos (elemento narrativo y posición) que poseen uno o varios posibles nodos subsiguientes.
- Planicie *(Plain)*: una colección de nodos que son accesibles en cualquier orden.

En la Figura 2.3 se presenta la interpretación gráfica de los tipos mencionados:

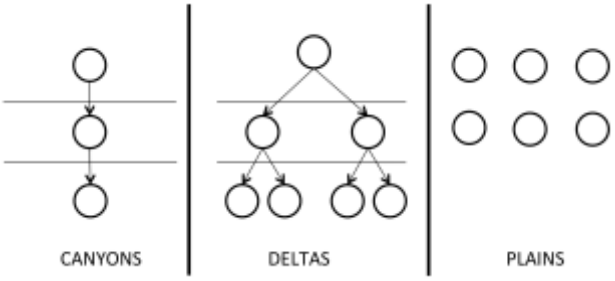

Figura 2.3 – Interpretación gráfica de los tipos de narrativa descriptos en [Millard et al., 2013].

Para este trabajo, serán de especial interés aquellas clasificaciones que se basen en la forma de estructurar los contenidos, como son las propuestas en [Millard et al., 2013] y [Kjeldskov et al., 2007], ya que, como se detallará más adelante, dicha estructura impacta sobre el funcionamiento de las aplicaciones.

2.3 Modelos de solución existentes

Si bien actualmente no existe un modelo de solución propuesto para la representación tanto de los datos como de la funcionalidad de las Aplicaciones Móviles basadas en Posicionamiento, algunos autores han postulado soluciones para situaciones específicas. Por ejemplo, en [Bouvin et al., 2003], los autores describen la arquitectura de un framework para la generación de hipermedia sensible al contexto, HyCon (contracción de *Hyper-Context*). Para ello, presentan un modelo de solución (ver Figura 2.4) en el que se representan solamente los *datos* utilizados en la aplicación, es decir, la funcionalidad de la misma no es parte del modelo de solución, y se encuentra codificada de manera ad hoc.

El modelo de datos de la Figura 2.4 representa *objetos de hipermedia* y *objetos físicos*, además de las *estructuras hipermedia*, en este caso acopladas al estándar XLink, el cual permite relacionar diferentes archivos XML. Esto se debe a que todos los objetos de este modelo poseen explícitamente la capacidad de ser convertidos a XML, ya que el framework utiliza dicho formato para manipularlos internamente.

La clase *AbstractObject* se utiliza para generalizar a los objetos de hipermedia y de contexto. Sus características comunes son un identificador global único, un conjunto de metadatos sobre el objeto en sí (como el creador y la fecha de creación del objeto), y un conjunto de "propiedades" (pares de clave-valor) que los autores describen como un punto en donde se puede dar extensibilidad y flexibilidad al modelo.
La clase *Context* se utiliza para agrupar objetos que pertenecen al mismo contexto. Los autores no lo mencionan, pero esta clase modela el patrón de diseño *Composite* ([Gamma et al., 1995]).
La clase *Profile* representa objetos que contienen información de preferencias sobre el usuario o el dispositivo que está utilizando.
La clase *Location* representa objetos asociados a un espacio físico; los autores describen

a las instancias de esta clase como "puntos de interés", es decir, la conjunción entre el contenido que se muestra al usuario y la posición asociada a dicho contenido.
Las clases *Link, Arc, Locator* y *Resource* sólo existen para respetar el estándar XLink ([DeRose et al., 2001]).

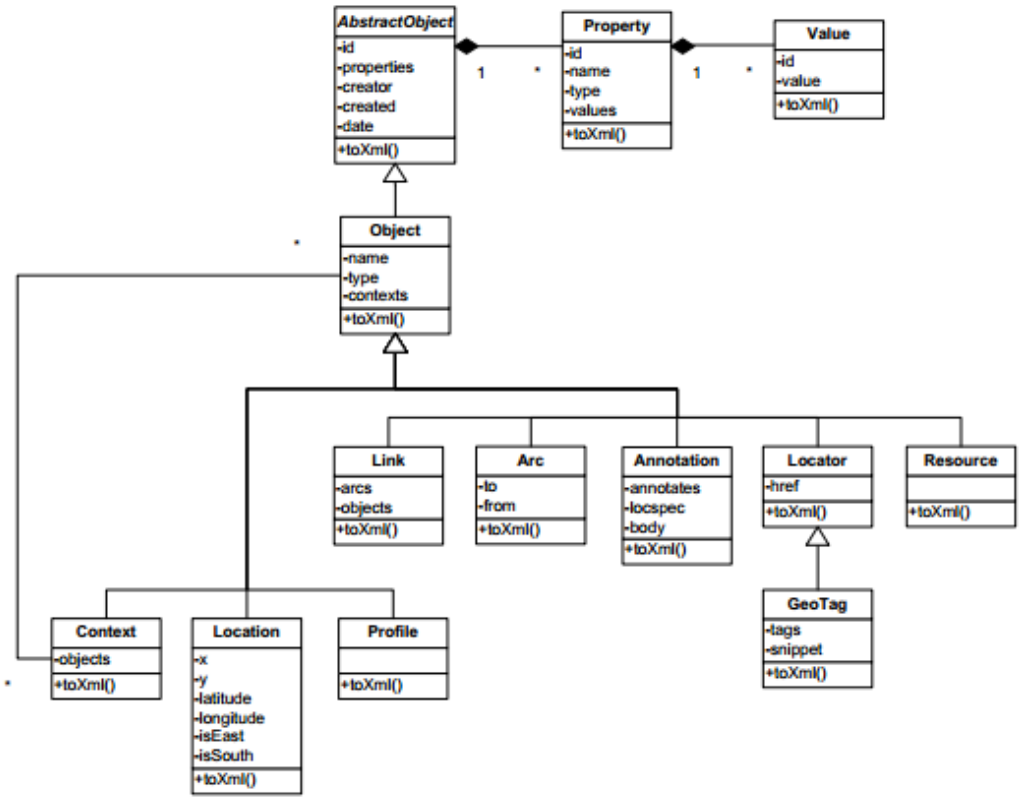

Figura 2.4 – Modelo de datos del framework HyCon, propuesto en [Bouvin et al., 2003].

Como puede observarse de la Figura 2.4 y la descripción de las clases brindadas por los autores del trabajo, el modelo sólo se utiliza como representación de datos, pero no define comportamiento más allá de la transformación a formato XML, que está relacionada con los pormenores de la implementación y no con el dominio de la temática.

Otra solución propuesta es la herramienta Castor (acrónimo de *Context Aware STORytelling*), introducida en [Pitarello et al., 2012], la cual fue desarrollada para asistir en la generación de *historias interactivas*, teniendo en cuenta características particulares de los usuarios que utilizarían la herramienta, en este caso, niños menores de diez años. Castor permite a los usuarios generar una experiencia exclusivamente *in situ*, es decir, en el lugar donde se llevará a cabo una vez puesta en marcha; esto implica que el o los usuarios que están creando la experiencia deben realizar el recorrido que luego será el que se brinde a los participantes del producto final.
Los autores de Castor utilizan como base un modelo conceptual introducido en [Pitarello et al., 2009], desarrollado a partir de las definiciones del reconocido filólogo italiano Cesare Segre; el concepto principal que extraen de él es el de *historia* (*story*), el cual se

define como la conjunción de fragmentos, cada uno de los cuales tiene un orden cronológico y situacional, y en conjunto forman el hilo argumental. A estos fragmentos, los autores de Castor llaman *escenas* (*scenes*); dichas escenas encapsulan una situación dentro de la historia, conteniendo así el espacio físico donde ésta transcurre, los personajes involucrados, los eventos, el tiempo y el contenido. En la Figura 2.5 se muestra la representación gráfica del concepto de *escena* propuesto en [Pitarello et al., 2009].

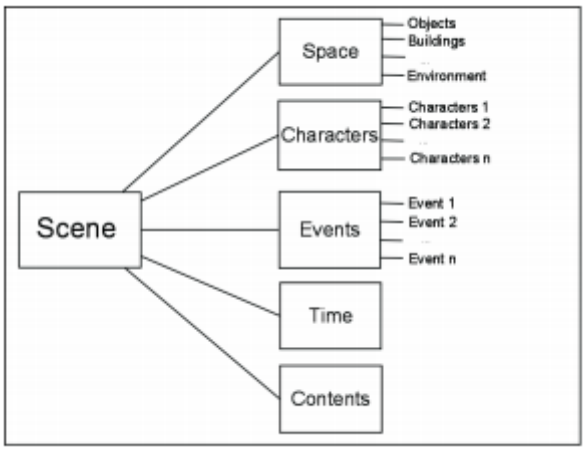

Figura 2.5 – Modelo conceptual de una *escena*, según la interpretación de [Pitarello et al., 2009].

Una *historia* queda entonces definida como la sucesión de *escenas* que la componen. Cabe destacar que los autores de Castor contemplan la posibilidad de que una historia pueda desenvolverse de manera diferente dependiendo de las situaciones presentadas a los usuarios, existiendo así varias alternativas para una misma historia; una representación visual de una historia se muestra en la Figura 2.6.

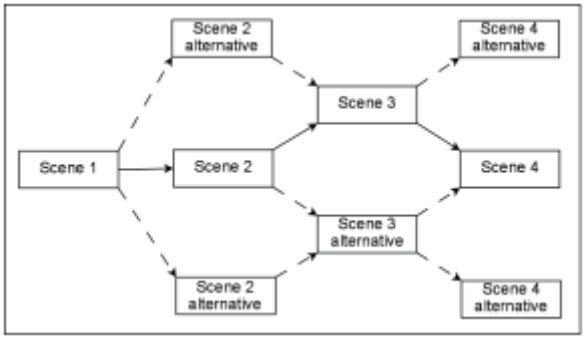

Figura 2.6 – Modelo conceptual de una *historia*, según la interpretación de [Pitarello et al., 2009].

En el modelo conceptual de esta herramienta, una *escena* es un agrupamiento en donde están acoplados conceptos disjuntos, como son los datos de contexto (entre ellos, de posicionamiento) y los contenidos propios de la experiencia en sí. Otra consideración a tener en cuenta es que, al ser una herramienta destinada exclusivamente a la generación de *historias interactivas*, sólo contempla estructuras narrativas *lineales*, es decir, sucesiones (con posibles bifurcaciones) con un orden prestablecido.

Por último, en [Millard et al., 2013] se propone un modelo de clases UML para representar narrativas de hipertexto basadas en posicionamiento. La clase principal de este modelo es *Story*, la cual representa la *historia* en su totalidad; cada historia contiene un conjunto de *capítulos*, representados por la clase *Chapter*. La clase *TimedChapter* representa un tipo especial de capítulo que incluye un evento de tiempo que permite realizar acciones controladas por un reloj. Cada capítulo está compuesto por *elementos de capítulo*, representados con la clase *ChapterElement*; los elementos pueden ser de clase *Node* o *Stack*, dependiendo de su naturaleza. La clase *Node* representa un contenido de la historia encapsulado; además, los nodos pueden tener embebidos *elementos contextuales* (clase *ContextualElement*), los cuales son consultas semánticas que se resuelven en tiempo de ejecución para determinar un dato de contexto (por ejemplo, un *elemento contextual* podría ser "*la ciudad más cercana"*, y cuando dicha consulta se resuelva en tiempo de ejecución, se reemplazará por el valor obtenido, por ejemplo, *"Chivilcoy"*). A su vez, un nodo puede estar asociado inflexiblemente a una posición geográfica (*PinnedNode*) o puede estar disponible libre de dicha asociación (*UnPinnedNode*). La clase *Stack* se utiliza para crear experiencias secuencias a partir de nodos sin relación con posiciones geográficas (*UnPinnedNode*). En la Figura 2.7 se muestra el diagrama de clases descripto.

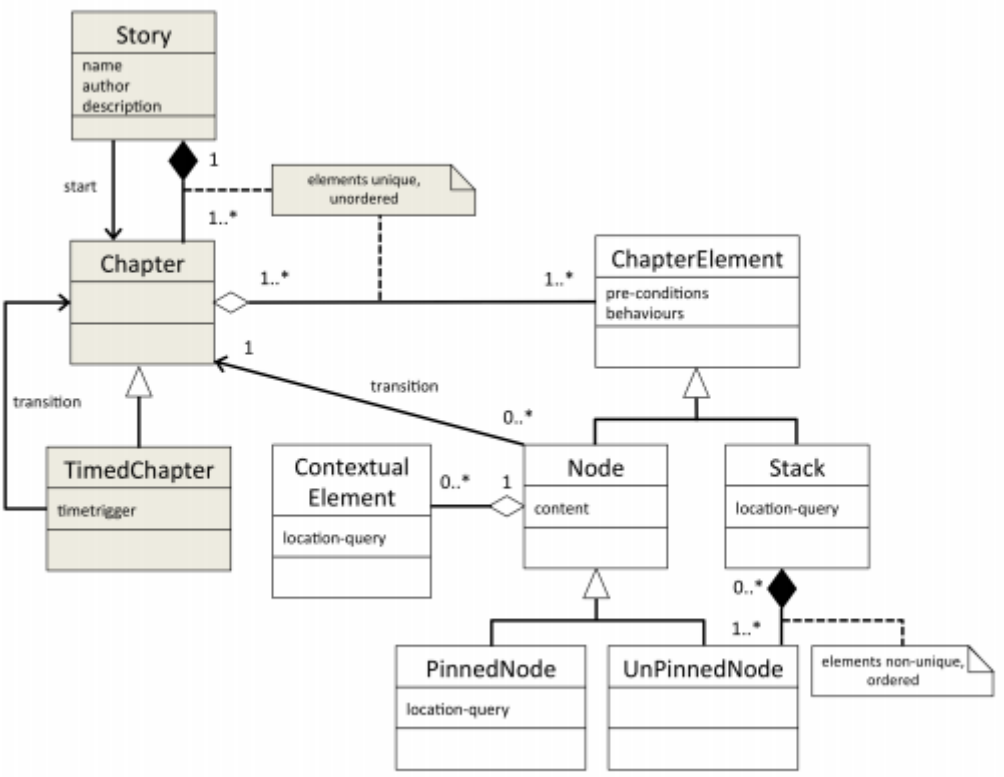

Figura 2.7 – Modelo de clases propuesto en [Millard et al., 2013].

3. Modelo propuesto

A lo largo de este capítulo se introducirán los conceptos y el modelo de objetos planteados para darle solución a la problemática planteada a partir del estado del arte realizado.

3.1 Conceptos principales

En esta sección se describirán los conceptos generales de las Aplicaciones Móviles basadas en Posicionamiento, en particular, aquellos que son necesarios para contextualizar adecuadamente la problemática a resolver.

Del análisis de las taxonomías de Aplicaciones Móviles basadas en Posicionamiento mencionadas en el Capítulo 2, desarrolladas en [Kjeldskov et al., 2007] y [Millard et al., 2013], además de los relevamientos analizados a partir de [Emmanouilidis et al., 2013], surge la necesidad de dar soporte a los distintos tipos de estructuras utilizadas para organizar los contenidos de las aplicaciones; para esto, se introducirá el concepto de *funcionalidad*, según se entiende en este trabajo:

> *La **funcionalidad** de una Aplicación Móvil basada en Posicionamiento define la estructura y las restricciones aplicadas a la forma en que los usuarios de la misma acceden a sus contenidos.*

Esta forma de definir la *funcionalidad* de las Aplicaciones Móviles basadas en Posicionamiento tiene como consecuencia la necesidad de contar con información sobre los usuarios de las mismas, necesaria para darles soporte adecuadamente. Por ejemplo, para guiar a un usuario durante el recorrido de una experiencia, es necesario conocer, además de su posición actual (brindada por el mecanismo de sensado), información sobre su lugar de destino. Cabe destacar que algunos de los estados detectados sólo tienen sentido o serán asignados a los usuarios de experiencias con determinado tipo de funcionalidad; por ejemplo, el estado *Seleccionando destino* sólo será utilizado en experiencias con un tipo de funcionalidad en la que los puntos de interés posean más de un posible siguiente. Algunos de los estados más representativos que se tendrán en cuenta en este trabajo son:

- *Experiencia iniciada*: estado en el que se encuentran los usuarios cuando comienzan la experiencia pero aún no están caminando hacia una posición o visualizando un contenido concreto. Este estado inicial puede omitirse según la funcionalidad adoptada en la experiencia, ya que podría asumirse que la posición en la que el usuario se une a la experiencia coincide con la del primer contenido, o simplemente se asume que una vez unido, pasará directamente a caminar hacia la primera posición.

- *Caminando hacia posición* P: los usuarios que tienen un destino concreto y se

encuentran caminando actualmente tendrán asignado este estado; al conocer la posición *P* a la que se está dirigiendo el usuario, se puede asistirlo para llegar a destino.

- *Visualizando contenido* C*:* estado en el que el usuario se encuentra visualizando un contenido concreto *C*, pudiendo ser este un texto, imagen, audio, etcétera. Es útil ya que con esta información puede calcularse los posibles destinos del usuario (si es que existiese alguno).

- *Seleccionando destino:* en las experiencias con tipos de funcionalidad que admitan que un punto de interés posea varios destinos posibles (como son la metáfora *Dominó* en [Kjeldskov et al., 2007] o los *Deltas* en [Millard et al., 2013]), este estado representa aquel en el que un usuario está seleccionando cuál será su siguiente destino, del conjunto de posibles.

- *Experiencia finalizada:* estado que indica que el usuario ha alcanzado la condición de fin de experiencia, la cual dependerá del tipo de funcionalidad de la experiencia (por ejemplo, con una funcionalidad *lineal*, un usuario finalizará la experiencia luego de visualizar el último contenido del recorrido).

Para ejemplificar una aplicación que utiliza estos estados de usuario, se asumirá una experiencia que consta de tres puntos de interés, compuestos cada uno por un contenido y una posición (se denotarán *Contenido 1, Contenido 2,* y *Contenido 3,* respectivamente; y *Posición 1, Posición 2*, y *Posición 3* a las posiciones asociadas a esos contenidos); además, dicha experiencia tendrá una funcionalidad *lineal*, es decir, los contenidos deben ser accedidos en orden, el cual está dado por los *elementos*. Dichos elementos definen la relación entre contenidos, manteniendo de esta manera la cohesión de los mismos, ya que no poseen otra información más que la que es de interés al usuario; esta separación entre los contenidos en sí y las relaciones estructurales que existen entre ellos disminuye el acoplamiento y permite reutilizar contenidos en distintas experiencias. En la Figura 3.1 se grafica la distribución del ejemplo de experiencia descripto.

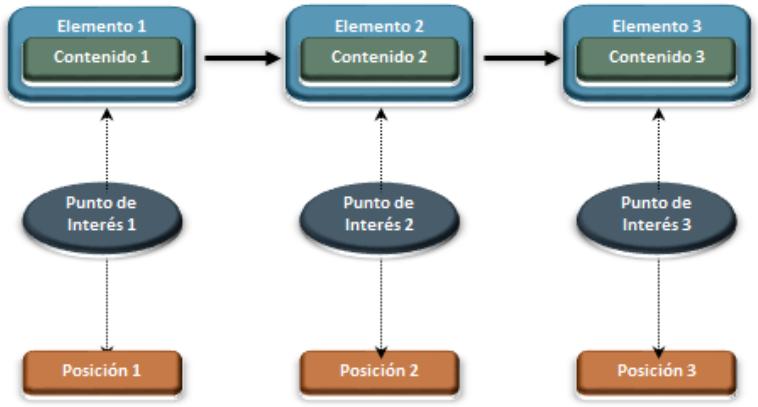

Figura 3.1| – Distribución de los puntos de interés en una aplicación de ejemplo.

El usuario, al comenzar la experiencia, se encontrará en el estado *Caminando hacia la Posición 1*. Una vez que un mecanismo de sensado (por ejemplo, el sensor GPS del dispositivo) indique que el usuario ha llegado a la posición destino, el mismo pasará al estado *Visualizando Contenido 1*. Cuando el usuario finalice de visualizar el Contenido 1, pasará al estado *Caminando hacia la Posición 2*, así hasta que el usuario finalice de visualizar el último contenido (Contenido 3), momento en el cual se le asignará el estado *Experiencia finalizada*. En la Figura 3.2 se muestra un Diagrama de Transición de Estados (DTE) representando los estados de los usuarios en la funcionalidad descripta anteriormente.

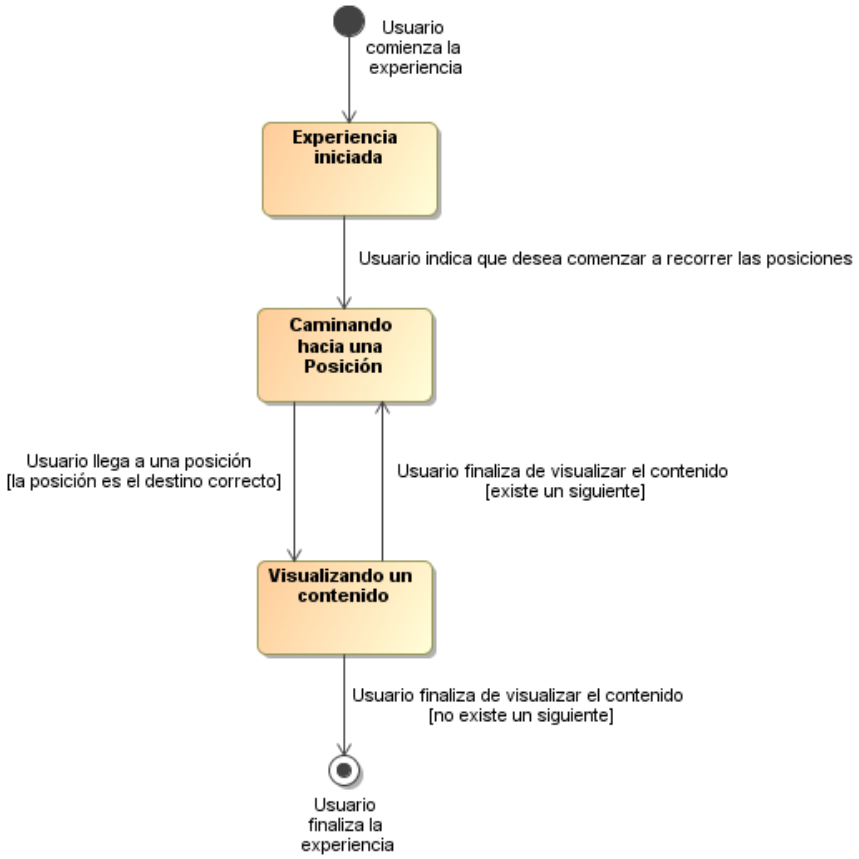

Figura 3.2 – Diagrama de Transición de Estados (DTE) de la funcionalidad descripta.

Adicionalmente a los estados de usuario, es necesario conocer las *acciones* que estos pueden tomar en el contexto de la experiencia. En el ejemplo de la aplicación anterior, la manera de detectar que un usuario finalizó de visualizar un contenido es, justamente, mediante la acción *Finalizar visualización de Contenido* C*,* la cual consiste en una notificación por parte del usuario (puede ser implícita o explícita, según la implementación requerida), que indica que el usuario ha terminado de visualizar el contenido *C*. En la Figura 3.2, las acciones de los usuarios quedarían representadas en las transiciones. De manera similar a los estados, existen acciones que sólo serán utilizadas en experiencias con un tipo de funcionalidad concreta. Por ejemplo, la acción *Seleccionar*

destino D, la cual se lleva a cabo cuando un usuario selecciona su próximo destino de un conjunto de posiciones posibles, sólo tiene sentido con una funcionalidad de tipo *grafo* (con bifurcaciones), mientras que en una funcionalidad *lineal*, donde cada punto de interés sólo posee un siguiente, el usuario nunca realizará dicha acción.

3.2 Problemática a resolver

La gran cantidad de dominios abarcados por las Aplicaciones Móviles basadas en Posicionamiento tiene como consecuencia la variabilidad en la funcionalidad de cada una de ellas; por ejemplo, en determinados casos es necesario utilizar una funcionalidad en la que los contenidos son brindados a los usuarios en un orden preestablecido, mientras que en otros casos los contenidos están disponibles para que los usuarios los consuman, sin controlar el orden en que lo hacen. La funcionalidad necesaria en cada caso debe ser evaluada concretamente, ya que cada aplicación requerirá un tipo que se adapte a sus necesidades específicas.

Actualmente esta problemática no cuenta con una solución genérica; la funcionalidad de las experiencias se desarrolla de manera ad hoc para cada caso. Por ejemplo, en la herramienta de generación de Aplicaciones Móviles basadas en Posicionamiento HyCon, introducida en [Bouvin et al., 2003] y extendida luego en [Hansen et al., 2008], la funcionalidad utilizada en una experiencia en particular es implementada manualmente por los desarrolladores, en una representación interna (en formato XML) de la herramienta. Otro caso, como el de la herramienta Castor ([Pittarello et al., 2012]), permite a usuarios no expertos construir experiencias in situ, las cuales pueden presentar distintos tipos de funcionalidad, dependiendo de cómo los usuarios compongan las llamadas *etapas de historia* (*story stages*), las cuales definen el contenido y la posición asociada, así como la conexión con otras *etapas* según su tipo. El problema de la herramienta Castor es que dichas *etapas de historia* están acopladas cada una a una posición GPS; esto tiene dos consecuencias significativas: primero, los contenidos no pueden ser reutilizados en distintas experiencias, ya que están asociados invariablemente a una posición GPS y a la relación que definen con otras etapas, y segundo, el mecanismo de sensado GPS está implementado como una parte inherente a la herramienta, por lo que no es posible cambiarlo.

Cabe destacar que tanto HyCon como Castor resuelven satisfactoriamente los problemas específicos para los que fueron desarrollados; el objetivo de este análisis es notar que no existe un modelo de solución genérico para las Aplicaciones Móviles basadas en Posicionamiento, que provea desacoplamiento de los conceptos de *contenidos*, *posiciones y mecanismos de sensado* de manera tal que sean reutilizables sin importar el dominio particular. Más aún, la *funcionalidad* de las aplicaciones no es contemplada como una parte variable de las experiencias, sino como una característica estática; este enfoque es la razón de la falta de una solución dinámica y escalable para este aspecto de las Aplicaciones Móviles basadas en Posicionamiento.

Para lograr la característica de dinamismo mencionada, en este trabajo la representación

de las posibles acciones y estados de los usuarios son de suma importancia, ya que con esta información se da forma a la funcionalidad de la experiencia: esto significa que las consecuencias de cada acción tomada por un usuario se definen mediante una correspondencia entre el par (*estado actual del usuario, acción tomada*) y el estado resultante de la interacción. Por ejemplo, con una funcionalidad *lineal* (cada punto de interés tiene un único sucesor), se podría afirmar que el par (estado: *Visualizando un contenido*, acción: *Finalizar visualización de contenido*) daría como consecuencia que el usuario involucrado pase al estado *Caminando hacia la posición* P (siendo *P* la posición correspondiente al siguiente punto de interés).

Esta es la razón por la cual en este trabajo se tiene al concepto de *regla de transición* como el más importante para la representación de la funcionalidad dinámica de una Aplicación Móvil basada en Posicionamiento:

> Una ***regla de transición*** se define como una directiva que, dada la información de entrada *acción realizada por un usuario* y *estado actual de ese usuario*, dará como resultado la consecuencia de esa interacción. La consecuencia en sí describe el estado al cual pasará el usuario involucrado.

En otras palabras, las reglas de transición de una aplicación definen, en conjunto, la *funcionalidad* de la misma. Por esta razón, se puede definir el concepto de *funcionalidad* a partir del concepto de *regla de transición*:

> *La **funcionalidad** de una Aplicación Móvil basada en Posicionamiento es el conjunto de reglas de transición que, en conjunto, definen las restricciones aplicadas a la forma en que los usuarios de la misma acceden a sus contenidos.*

En las Figuras 3.3 se muestra la representación gráfica del concepto de regla de transición.

Figura 3.3 – Representación de una regla de transición genérica.

A manera de ejemplo, se listarán a continuación las reglas de transición correspondientes a la funcionalidad descripta en la Figura 3.2:

- **Regla 1:**
  - Aplicada a:
    - Estado actual: *-indefinido-.*
    - Acción realizada: *Unirse a la experiencia.*
  - Estado resultante: *Experiencia iniciada.*

- **Regla 2:**
  - Aplicada a:
    - Estado actual: *Experiencia iniciada.*
    - Acción realizada: *Comenzar experiencia.*
  - Estado resultante: *Caminando hacia una posición.*

- **Regla 3:**
  - Aplicada a:
    - Estado actual: *Caminando hacia una posición.*
    - Acción realizada: *Llegar a una posición.*
  - Estado resultante: *Visualizando un contenido.*

- **Regla 4:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*
  - Estado resultante:
    - Caso 1: Si no existe un siguiente → *Experiencia finalizada.*
    - Caso 2: Si existe un siguiente → *Caminando hacia una posición.*

Como puede observarse, existen reglas (como la Regla 4, en el ejemplo) las cuales pueden producir distintos estados resultantes para un mismo par (*estado actual, acción realizada*); esto es así ya que algunas transiciones requieren la evaluación de condiciones del contexto de la experiencia (en el ejemplo, verificar si el contenido visualizado es el último del recorrido o posee un sucesor). Este aspecto de las reglas se describirá oportunamente en la Sección 3.3.

Como ejemplo del dinamismo que aporta el concepto de regla de transición, considérese la experiencia descripta en la sección anterior, donde la Figura 3.2 describe las posibles transiciones de estado. Nótese que en la transición del estado *Caminando hacia posición* hacia *Visualizando contenido*, se asume que el usuario ha llegado a la posición indicada. ¿Pero qué sucede cuando el usuario llega a una posición que no era la indicada? Simplemente se podría definir una nueva regla de transición para contemplar este caso, definiendo que dicha interacción resultará en el estado *Caminando hacia posición*, para reflejar el comportamiento en el que la aplicación le podría indicar al usuario que ha llegado a una posición que no era el destino original, y guiarlo según corresponda. En la Figura 3.4 se muestra el DTE modificado con la incorporación de esta nueva regla a la funcionalidad:

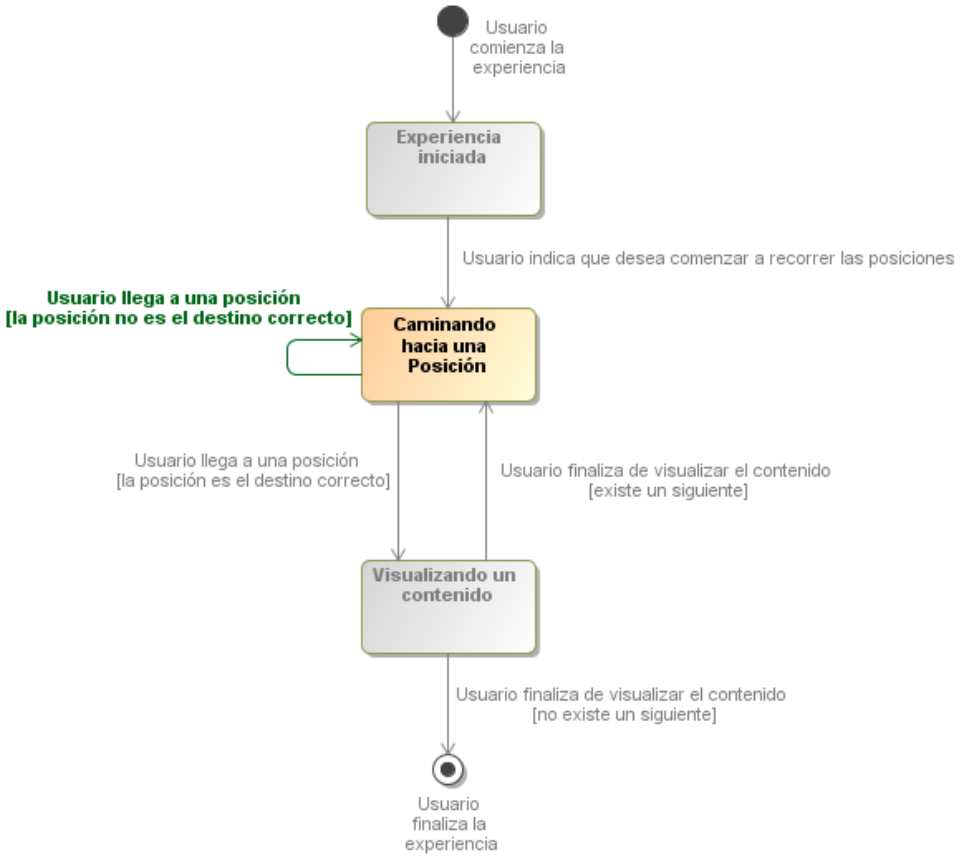

Figura 3.4 – Diagrama de Transición de Estados (DTE) de la funcionalidad modificada con la nueva regla.

Este ejemplo grafica la simplicidad con la que se podría modificar la funcionalidad de una aplicación mediante la modificación de una de las reglas de transición:

- **Regla 3 *(modificada)*:**
  - Aplicada a:
    - Estado actual: *Caminando hacia una posición.*
    - Acción realizada: *Llegar a una posición.*
  - Estado resultante:
    - Caso 1: Si la posición es el destino correcto → *Visualizando un contenido.*
    - Caso 2: Si la posición no es el destino correcto → *Caminando hacia una posición.*

El dinamismo brindado por este enfoque se aplica también en casos más críticos, como en la modificación de una funcionalidad de manera tal que signifique un cambio de *tipo* (según las taxonomías analizadas anteriormente, introducidas en [Kjeldskov et al., 2007] y [Millard et al., 2013]). Por ejemplo, si quisiera transformarse la funcionalidad descripta en la Figura 3.4 de manera que ya no sea *lineal*, sino que presente una estructura de *grafo*, es decir, que cada punto de interés pueda tener más de un posible sucesor, y que además

el usuario pueda elegir su destino ante cada *bifurcación*, bastaría con agregar y modificar las reglas de transición necesarias como se detalla a continuación:

- **Regla 2 *(modificada)*:**
  - Aplicada a:
    - Estado actual: *Experiencia iniciada.*
    - Acción realizada: *Comenzar recorrido.*
  - Estado resultante:
    - Caso 1: Si hay más de un posible destino → *Seleccionando próximo destino.*
    - Caso 2: Si no hay más de un posible destino → *Caminando hacia una posición.*
- **Regla 4 *(modificada)*:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*
  - Estado resultante:
    - Caso 1: Si no existe un siguiente → *Experiencia finalizada.*
    - Caso 2: Si existen varios siguientes → *Seleccionando próximo destino.*
    - Caso 3: Si existe sólo un siguiente → *Caminando hacia una posición.*
- **Regla 5 *(agregada)*:**
  - Aplicada a:
    - Estado actual: *Seleccionando próximo destino.*
    - Acción realizada: *Seleccionar próximo destino.*
  - Estado resultante: *Caminando hacia una posición.*

Estos cambios en el conjunto de reglas de transición bastarían para que la funcionalidad de la experiencia pase a ser del tipo *grafo*, descripta a continuación en la Figura 3.5:

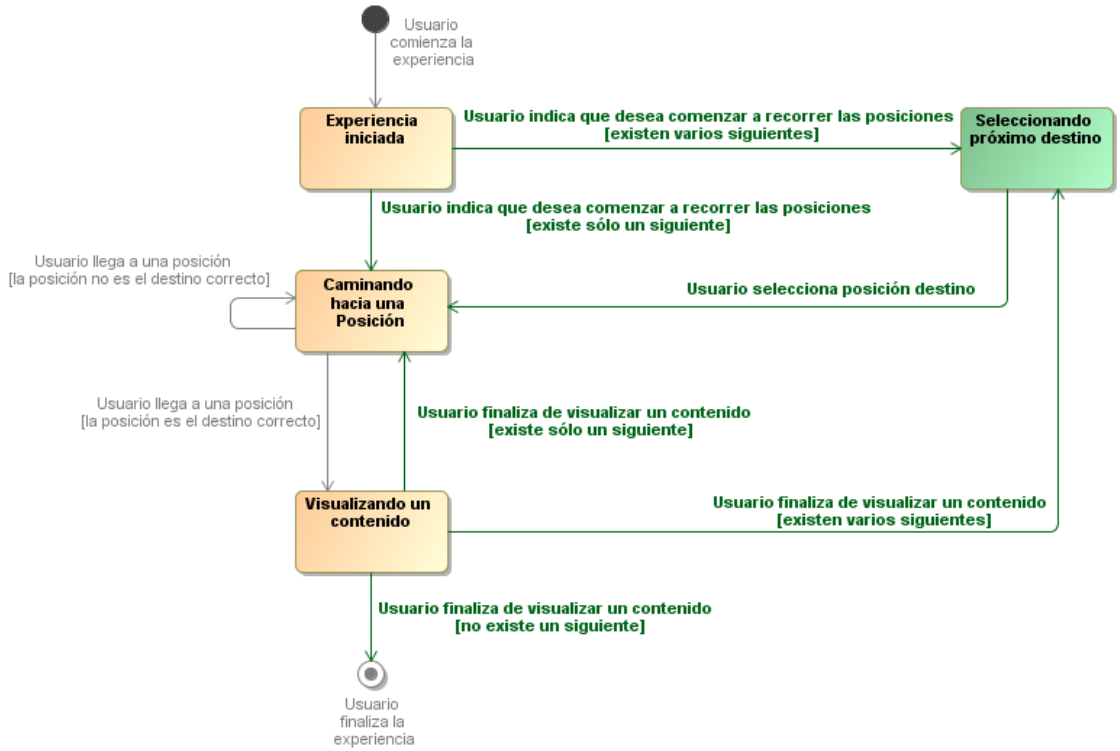

Figura 3.5 – Diagrama de Transición de Estados (DTE) de la funcionalidad modificada con la nueva regla.

3.3 Características del modelo propuesto

El modelo que se presenta en este trabajo propone resolver la problemática planteada a través de un diseño de solución genérico, escalable y dinámico (según las *Leyes de evolución del software* descriptas en [Lehman, 1996], como *cambio continuo*, *crecimiento continuo*, y *disminución de calidad*, entre otras) para la representación de funcionalidad de las Aplicaciones Móviles basadas en Posicionamiento; "genérico" por la necesidad de poder representar cualquier tipo de funcionalidad, "escalable" para que la aparición o detección de nuevos tipos de funcionalidad sea trasladable fácilmente al modelo, y "dinámico" en el sentido de que la representación de la funcionalidad sea intercambiable o modificable en tiempo de ejecución.

En cuanto a los *contenidos* de las aplicaciones, es necesario representarlos de una manera genérica, que permita incorporar nuevos tipos según las aplicaciones lo requieran. Por razones prácticas y para la puesta en marcha de los prototipos desarrollados, se creó tipo de contenido concreto, el *texto plano*, el cual simplemente representa un contenido expresado en texto estático. Es importante destacar que el concepto de *contenido* no involucra otra responsabilidad más que la de representar la información que se le brindará al usuario; es por esto que las relaciones entre los contenidos de una aplicación están dadas por una entidad desacoplada de los mismos; estos son los *elementos*, los cuales pueden poseer (o no, dependiendo de cómo se estructure la aplicación) relaciones con otros elementos. Esta separación encuentra su justificación en temas que sobrepasan el

alcance de este trabajo, por lo que basta mencionar que de esta manera se logra un completo desacoplamiento entre la capa *física* y la capa *digital* de cada aplicación; este concepto es introducido en [Alconada et al., 2015a], en donde la capa de contenidos queda desacoplada de la capa física, y son los *puntos de interés* los que los relacionan. En la Figura 3.6 se grafica dicha separación:

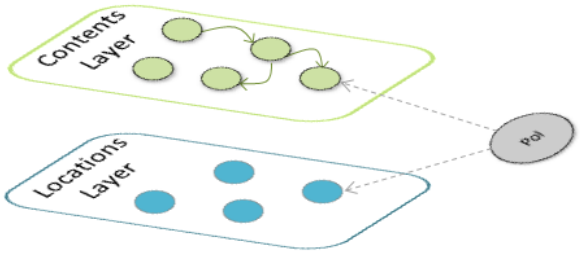

Figura 3.6 – Separación conceptual entre la capa de contenidos y la capa de posiciones físicas presentada en [Alconada et al., 2015a].

Si bien el enfoque de solución adoptado para las reglas de transición puede encontrar algunas similitudes con los lineamientos seguidos en [Arsanjani, 2001], el cual describe un conjunto de variantes del patrón de comportamiento *Rule Object*, el modelo de solución propuesto difiere en cuanto al comportamiento y el objetivo del patrón en su forma tradicional, ya que en este caso, las reglas de transición actúan conjuntamente para representar un tipo de funcionalidad de una Aplicación Móvil basada en Posicionamiento, y no son reglas de negocio "volátiles" o independientes unas de otras. Además, cada regla posee una complejidad adicional, dada por el valor de c*ondiciones de contexto* que posee en sus ternas de evaluación; estas condiciones fueron asumidas como un valor simple para los ejemplos presentados anteriormente, pero internamente poseen una lógica de evaluación de predicados que requiere de una solución más compleja.

3.4 Descripción del modelo propuesto

En esta sección se describirán los conceptos principales del modelo propuesto, que son aquellos que ofrecen una solución a la problemática descripta: representar la funcionalidad de las Aplicaciones Móviles basadas en Posicionamiento, de manera genérica, escalable y dinámica.

En cuanto al modelo de solución, es claro que debe dar soporte a la representación de la funcionalidad de una Aplicación Móvil basada en Posicionamiento, independientemente de sus particularidades. Más aún, el modelo debe ser lo suficientemente escalable para contemplar nuevos tipos de funcionalidad que puedan surgir.

En primer lugar se describirán los conceptos básicos que forman parte de la representación de las Aplicaciones Móviles basadas en Posicionamiento, para luego introducir los conceptos que dan solución específicamente al tema que abarca este trabajo, la generación dinámica de funcionalidad. Varios de estos conceptos, como el de *contenido*, *elemento* y *punto de interés*, ya fueron introducidos en [Alconada et al., 2015b]

El concepto base de toda aplicación de este tipo es el de *contenido*, el cual representa una pieza de información que puede ser texto plano, imágenes, video, o cualquier otro formato multimedia, que se pretende brindar al usuario. La clase *Content,* graficada en el diagrama de clases UML de la Figura 3.7, representa justamente este concepto; es la información que se les brinda a los usuarios de la experiencia. En dicho diagrama se presenta a un contenido genérico mediante la clase *Content*, y el tipo concreto de contenido queda definido por las subclases. A modo de ejemplo se incluyeron algunos subtipos concretos como *PlainTextContent* (texto plano), *Question* (pregunta), *Image* (imagen), pero la idea subyacente de este modelo es que ante la necesidad de un nuevo tipo de contenido, baste con extender la superclase *Content*.

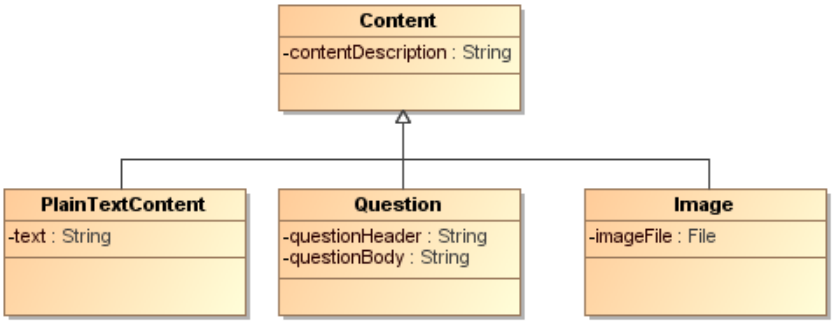

Figura 3.7 – Clase *Content* con algunos tipos concretos de contenido de ejemplo.

Nótese que en el diagrama de la Figura 3.7, los *contenidos* poseen sólo la información que se le brinda al usuario, y no incluyen por ejemplo información sobre el posicionamiento o la relación de orden con los demás contenidos; esta forma de modelar disminuye el acoplamiento que se produce entre las distintas clases de la solución, es decir, cuánto dependen unas de otras para ser utilizables.
Como consecuencia de este bajo acoplamiento, se introdujo el concepto de *elemento*, representados por la clase *Element*, cada uno de los cuales tienen la responsabilidad de definir una relación de orden (en sentido abstracto, no en cuanto a posicionamiento) existente entre los contenidos. De esta manera, la clase *Content* sólo posee la información brindada al usuario, y la clase *Element* es la que define el orden, si es que lo hubiere, de dichos contenidos. En el diagrama de la Figura 3.8 se introduce la clase *Element*, la cual posee un conjunto de *relaciones* (clase *Relationship*) con otros elementos; en [Alconada et al., 2015b], estas *relaciones* son denominadas *nodos* (“*nodes”*), por su similitud con el concepto de nodo que se tiene en la teoría de grafos. Para el contexto de esta tesina basta comprender que dicha entidad representa una relación entre dos elementos.

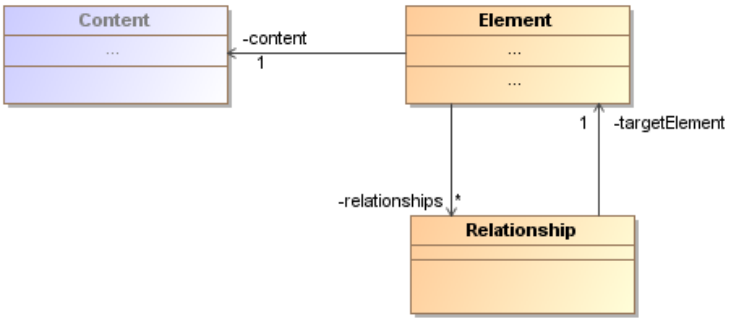

Figura 3.8 – Clase *Element* con su contenido asociado y las relaciones con otros elementos.

Nuevamente, cabe destacar que la relación que definen los *elementos* es puramente "digital", es decir, establecen el orden de los contenidos (por ejemplo, el *Contenido 1* debe ser visualizado antes del *Contenido 2*) sin decir nada sobre la información posicional de los mismos.

Para poder asociar los contenidos a posiciones físicas del mundo real es necesario representar una *posición*. Para esto se introduce la interfaz *Position*; se optó por utilizar una interfaz en lugar de una clase ya que la representación de una posición física puede ser muy diferente dependiendo del mecanismo de sensado utilizado, por lo que, para mantener un modelo de solución genérico y extensible, se define sólo el comportamiento que debe ser capaz de llevar a cabo cualquier objeto que se considere una *posición*. A modo de ejemplo, en el modelo propuesto se incluyó la clase QRCode, la cual representa un código QR, que en este caso se utilizará como una posición. En el diagrama de la Figura 3.9 se grafica la interfaz *Position* y la clase *QRCode* descriptas.

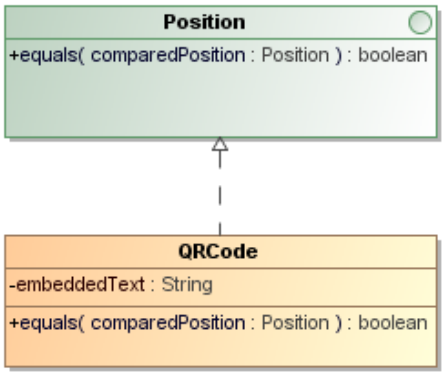

Figura 3.9 – Interfaz *Position* y clase QRCode, que implementa dicha interfaz.

Si bien la interfaz *Position* es la que representa a las posiciones físicas del mundo real, la relación entre dichas posiciones y los elementos (o, de manera equivalente, entre posiciones y contenidos) se define mediante el concepto ya mencionado de *punto de interés*. El mismo se representa mediante la clase *PointOfInterest*, y su principal función es la de asociar un elemento (contenido) a una posición física. En la Figura 3.10 se muestra el diagrama en el que dicho concepto "une" a los elementos con la posición a la

que están asociados. Esta separación es un aspecto importante del modelo propuesto, ya que la "capa digital", es decir, los contenidos y la relación lógica entre ellos, queda desacoplada de la "capa física", las posiciones; los *puntos de interés* son lo que vinculan ambas capas.

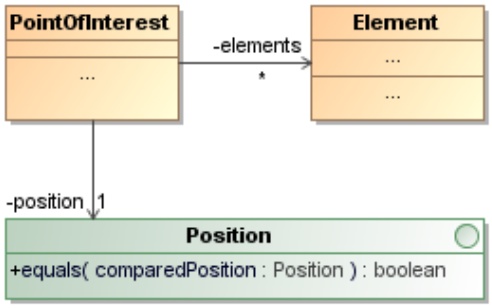

Figura 3.10 – Los puntos de interés (clase *PointOfInterest*) relacionan la parte digital (*Element*) con la de posicionamiento físico (*Position*).

Finalmente, los usuarios de las Aplicaciones Móviles basadas en Posicionamiento se representan mediante la clase *User*, la cual posee un *estado actual* (clase *UserState*) que representa qué está haciendo el usuario en un momento dado. A su vez, cada estado de usuario tiene asociado un *tipo de estado* (clase *UserStateType*). Esta solución combina los patrones de diseño *State* ([Gamma et al., 1995]) y *TypeObject* ([Johnson et al., 1996]); en primer lugar, el uso del patrón *State* se justifica con el hecho de que el estado del usuario podría cambiar su comportamiento. En segundo lugar, el patrón *TypeObject* es necesario ya que, como se explicará más adelante, los *tipos* de estado (sin importar la información a nivel de instancia) son necesarios para el funcionamiento de las reglas de transición. En la Figura 3.11 se muestra el diagrama de clases con los conceptos de *usuario*, *estado de usuario*, y *tipo de estado de usuario*.

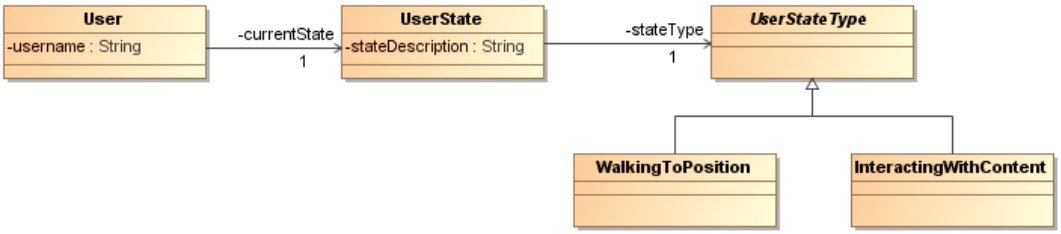

Figura 3.11 – Cada usuario (clase *User*) posee un estado actual asignado (clase *UserState*); a su vez, cada estado tiene un tipo (clase *UserStateType*).

De los usuarios que participan de la experiencia interesa también conocer las acciones que estos pueden realizar en el contexto de la aplicación; por esta razón se modeló una jerarquía de acciones posibles. Esta solución se asemeja al patrón *Command* ([Gamma et al., 1995]), el cual encapsula acciones y los datos necesarios para ejecutarlas. La diferencia con el patrón *Command* en su estado puro es que la jerarquía de acciones que pueden realizar los usuarios es utilizada principalmente como dato por las reglas de transición más que como abstracciones de comportamiento; es por esto que también para

este caso se utilizó el patrón de diseño *TypeObject* ([Johnson et al., 1996]), para desacoplar una acción de su tipo concreto. En la Figura 3.12 se muestra el diagrama de clases de la jerarquía de acciones de usuario; las clases concretas en la jerarquía de *UserActionType* son las acciones básicas que podría realizar un usuario, pero se debe tener en cuenta que esta jerarquía es extensible según se requiera representar nuevas acciones.

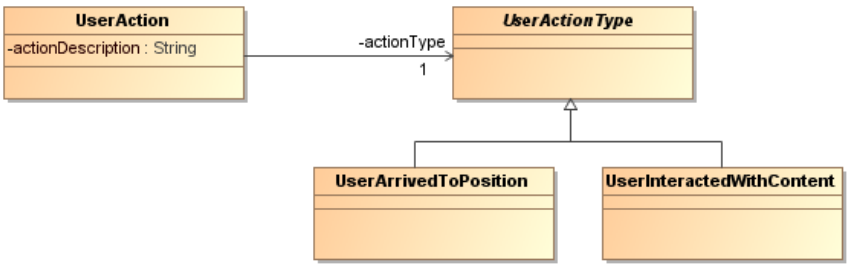

Figura 3.12 – Una acción de un usuario en una experiencia se representa mediante la clase *UserAction*, la cual posee a su vez un tipo asignado (clase *UserActionType*) que la distingue concretamente.

Hasta aquí se han descripto los conceptos necesarios para la representación de las Aplicaciones Móviles basadas en Posicionamiento en forma general; a partir de ahora se focalizará la descripción de los conceptos relacionados directamente con la solución propuesta para la representación dinámica y escalable de la funcionalidad de las mismas.

En la Sección 3.2 se introdujo el concepto de *funcionalidad* de una Aplicación Móvil basada en posicionamiento, concluyendo que podía entenderse como las restricciones resultantes del conjunto de *reglas de transición* que la componen. Una *regla de transición* define una relación de correspondencia entre una terna compuesta por *{estado actual del usuario, acción que realizó el usuario, condiciones de contexto}* y un estado resultante; de esta manera, al definir una regla de transición, se está implementando una pequeña parte de la funcionalidad de la aplicación. Los tres campos que componen la terna de una regla se describen como:

- Tipo de Estado actual del usuario: es el tipo de estado del usuario al momento de iniciar la transición. Este campo es una asociación en la clase que representa a una regla de transición (clase *TransitionRule*). Es aquí donde puede apreciarse la necesidad de utilizar el patrón de diseño TypeObject ([Johnson et al., 1996]) para representar los posibles tipos de estados de usuario.

- Acción realizada por el usuario: al igual que el estado del usuario, el tipo de acción que realizó para dar comienzo a una transición, está representado como una asociación en la clase *TransitionRule*.

- Condiciones de contexto: a diferencia de los otros dos campos, no son un valor simple que se recibe como dato de entrada, sino que representan determinadas condiciones del contexto mediante la evaluación de predicados. Para esto, cada

regla de transición tiene un conjunto de *aserciones de interacción* (clase *InteractionAssertion*). Cada aserción representa una situación de la forma: *siempre que el usuario haya realizado la acción* A (dada por el campo de la terna en la regla) *partiendo desde el Estado actual* E (dato también provisto por la regla), *su estado resultante será* A', *siempre y cuando se cumplan las condiciones que hacen a esta aserción verdadera*.

Es importante destacar que para una terna de evaluación, sólo puede existir una única regla aplicable; análogamente, al evaluar los predicados de una regla en particular, sólo una aserción será aplicable a dicha regla. Una aserción es *aplicable* si y sólo si todas las condiciones que la componen son verdaderas.

A modo de ejemplo, una regla de transición podría definirse como:

- **Regla:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*
  - Estado resultante: *Caminando hacia una posición.*

En este caso, la regla estaría definiendo una parte de una funcionalidad *lineal*, ya que se indica que cuando un usuario finaliza de consultar un contenido, directamente se le indica a dónde se debe dirigir, asumiendo que sólo existe un único camino posible. En la Figura 3.13 se muestra una primera aproximación (no completa aún) del diagrama de clases para la representación de las reglas de transición.

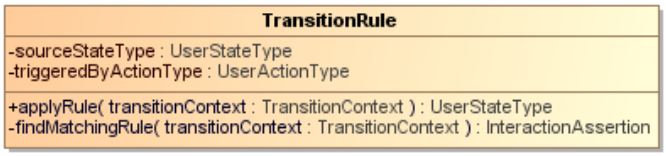

Figura 3.13 – Primera aproximación de la representación de una regla de transición.

El componente de *condiciones de contexto* no es un dato de entrada como lo son el *estado actual del usuario* o la *acción realizada*, sino que es una evaluación interna del estado actual del contexto de la experiencia; esto es así ya que diferentes condiciones de contexto podrían alterar el estado resultante para un mismo par de *estado actual* y *acción realizada*. Para representar la información de contexto que se consultará internamente en la evaluación de una regla, se introduce la entidad de *contexto de transición* (clase *TransitionContext*), la cual encapsula la información relevante al contexto en el momento de realizar la evaluación de la regla de transición. A modo de ejemplo, la misma regla presentada anteriormente podría contemplar el caso en que el usuario marcó como finalizado un contenido que no corresponde con el que estaba visualizando (según indica su *estado actual*), lo que significaría que la misma regla podría resultar en dos estados diferentes, dependiendo de las condiciones de contexto. La regla modificada quedaría definida de la siguiente manera:

- **Regla *(modificada)*:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*

  - Estado resultante:
    - Caso 1: Si el contenido finalizado coincide con el que estaba visualizando → *Caminando hacia una posición.*

    - Caso 2: Si el contenido finalizado no coincide con el que estaba visualizando → *Visualizando un contenido.*

Cabe destacar que el estado resultante del Caso 2 fue elegido arbitrariamente a modo ejemplo, para representar el hecho de que si el usuario marca como finalizado un contenido que no era el que estaba visualizando, se le mostrará nuevamente el contenido original; el estado resultante en cada caso de la regla podría ser cualquiera que el creador de la experiencia desee. En la Figura 3.14 se muestra el diagrama de clases que representa una regla de transición, esta vez junto con las aserciones (clase *InteractionAssertion*) y las condiciones (*InteractionCondition*) que la componen, así como también la clase *TransitionContext*, que encapsula el contexto de evaluación que podría ser necesario consultar para completar la evaluación.

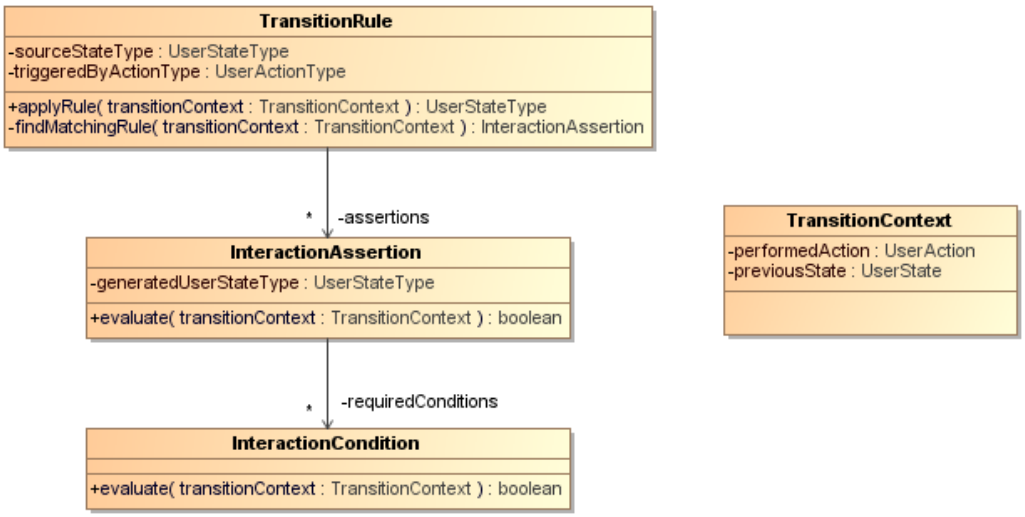

Figura 3.14 – Diagrama completo de las partes con conforman una regla de transición.

Como se mencionó anteriormente, cada Aplicación Móvil basada en Posicionamiento posee un *conjunto* de reglas, por lo que es necesario contar con un manejador de reglas de transición, que tenga la función de, además de contener a todas las reglas de la experiencia, tener la capacidad de seleccionar la regla aplicable en cada caso. Esta es la responsabilidad de la clase *TransitionRuleManager*, presentada en el diagrama de la Figura 3.15.

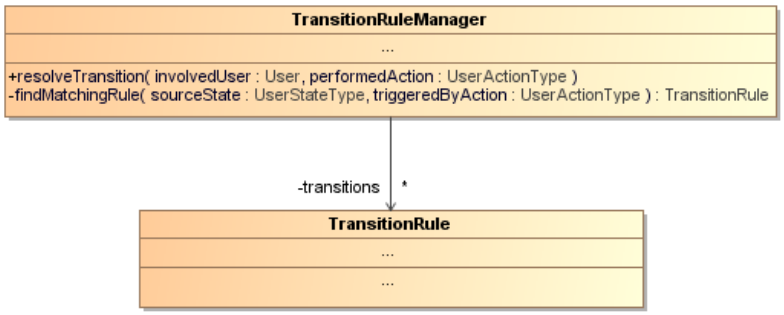

Figura 3.15 – Administrador de reglas de una aplicación; contiene todas las reglas aplicables y sabe buscar cuál de ellas se debe aplicar en cada caso.

Para resumir el funcionamiento de las reglas se puede decir que, cuando un usuario de la aplicación realiza una acción, la consecuencia es "calculada" con una regla de transición que toma como entrada la acción realizada y el estado actual del usuario, y luego de verificar las condiciones de contexto pertinentes, genera un estado resultante de la aplicación de la regla correspondiente.

Finalmente, la clase *Experience* representa una experiencia en particular, la cual se relaciona con los distintos módulos que la componen, como son los participantes (a través de la clase *UserManager*), los puntos de interés (a través de la clase *PoIManager*) y las reglas de transición (a través de la clase *TranstionRuleManager*). El diagrama de la Figura 3.16 muestra dicha disposición.

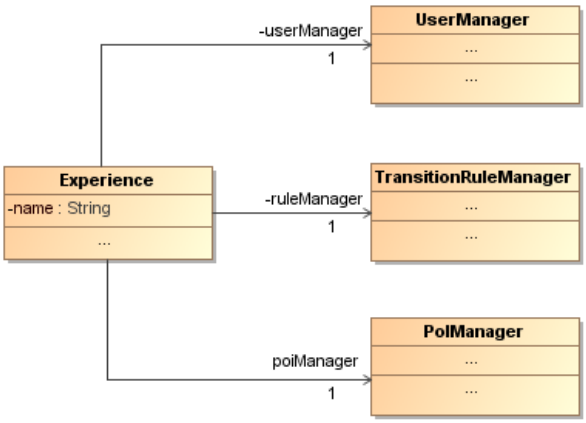

Figura 3.16 – Diagrama general de la experiencia, relacionada con el resto de los modelos a través de *managers*.

Para explicitar el comportamiento de este modelo al momento de reaccionar ante un evento de un participante dentro de la experiencia se introducirán los métodos más importantes para su comprensión. El primero de ellos es el método *resolveTransition(User, UserAction)* de la clase *Experience*; el mismo indica a la

experiencia que un participante (indicado por el parámetro *User*) ha realizado una acción relevante (indicado por el parámetro *UserAction*). Como la consecuencia toda acción relevante realizada por un participante depende de la funcionalidad de la experiencia, se delega esta responsabilidad a la clase *TransitionRuleManager*, la cual, como se dijo anteriormente, contiene todas las reglas aplicables que definen a la funcionalidad de la experiencia; para delegar esta responsabilidad, la clase *Experience* prepara el contexto de transición con los datos necesarios, y luego invoca al método *resolveTransition(TransitionContext)* de la clase *TransitionRuleManager*. Este *manager* de reglas se encarga, en primer lugar, de encontrar una regla de transición aplicable al estado actual del participante involucrado y la acción realizada (recordar que una regla se identifica por los datos de entrada *estado actual del participante* y *acción realizada*); esta búsqueda se realiza mediante el método interno *findMatchingRuleFor(UserState, UserAction)*, dando como resultado la regla aplicable al caso concreto. El siguiente paso es delegar la resolución de la transición a la regla que aplica al caso concreto; esto significa invocar al método *resolveTransition(TransitionContext)* de la regla. Internamente, la regla verificará las *condiciones de contexto,* las cuales, como se describió oportunamente, forman la tercera componente que define el estado resultante para la situación concreta a la que se aplica la regla. Una vez resulta la regla, esta dará como resultado el nuevo estado del participante que ejecutó la acción. En el diagrama de secuencia de la Figura 3.17 se grafica esta interacción.

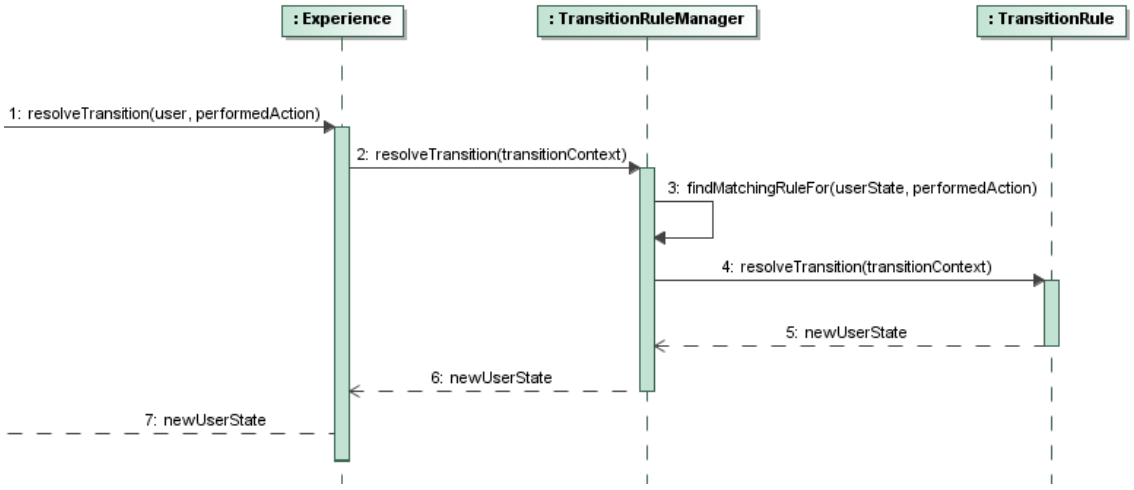

Figura 3.17 – Diagrama de secuencia que describe la interacción entre los objetos involucrados cuando debe resolverse la interacción de un participante en una experiencia.

Hasta aquí, se han descripto los conceptos del modelo propuesto para la creación dinámica de funcionalidad para las Aplicaciones Móviles basadas en Posicionamiento. En los capítulos siguientes se mostrarán los resultados de la aplicación de estos conceptos en las pruebas y prototipos realizados.

4. Prototipos desarrollados

En este capítulo se describirán los prototipos de Aplicaciones Móviles basadas en Posicionamiento que fueron implementados para realizar pruebas empíricas del modelo propuesto. El primero de ellos presenta una funcionalidad *lineal*, es decir, los contenidos de la experiencia están ordenados secuencialmente, y los participantes deben realizar el recorrido en el orden adecuado; el segundo prototipo presenta las modificaciones necesarias a las reglas de transición del primer prototipo, de modo que el mismo presente una funcionalidad de *grafo*, es decir, en ciertos puntos de la experiencia existirán casos en los que hay más de un camino posible por el que los participantes pueden continuar, y será su elección por dónde hacerlo.

El desarrollo de los prototipos se dividió en dos etapas: la primera consistió en la definición de la funcionalidad de la experiencia. En este caso, se utilizó el formato XML (*eXtensive Markup Language*) para representar las reglas de transición introducidas en el Capítulo 3. La segunda etapa consiste en la implementación de una aplicación Web para poner a prueba la funcionalidad definida. En las secciones subsiguientes se describirán los detalles de los prototipos desarrollados.

4.1 Generalidades de ambos prototipos

Ambos prototipos utilizan el mismo conjunto de contenidos y puntos de interés, pero relacionados de manera diferente. Cabe destacar que, al ser sólo *prototipos*, los valores concretos de los contenidos en sí carecen de relevancia para esta tesis. A continuación se describen los valores de cada contenido, según aparecen en ambos prototipos:

Contenido 1:
- Título: *Content 1.*
- Tipo: contenido textual.
- Texto: *Este es el contenido 1.*

Contenido 2:
- Título: *Content 2.*
- Tipo: contenido textual.
- Texto: *Este es el contenido 2.*

Contenido 3:
- Título: *Content 3.*
- Tipo: contenido textual.
- Texto: *Este es el contenido 3.*

Contenido 4:
- Título: *Content 4.*
- Tipo: contenido textual.
- Texto: *Este es el contenido 4.*

Como ejemplo de espacio físico para los puntos de interés se utilizó un plano del *Museo de Ciencias Naturales de La Plata*[4]. En la Figura 4.1.1 se muestra el plano general con las cuatro posiciones donde se mostrarán los cuatro contenidos mencionados anteriormente. Notar que, por simplicidad, las posiciones marcadas con los códigos QR referencian a salas del museo en general, y no un punto específico dentro de ellas.

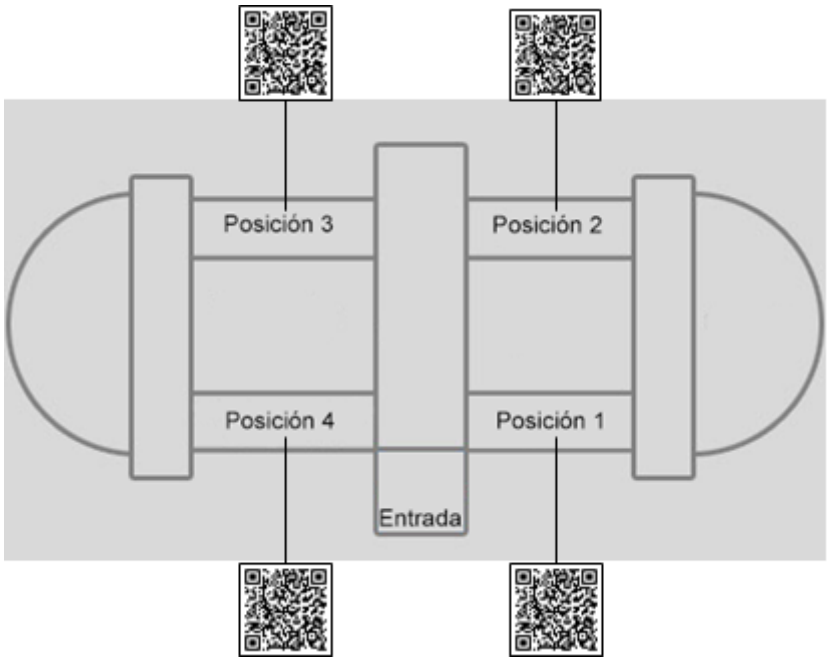

Figura 4.1.1 – Plano general del espacio físico utilizado para ambos prototipos.

Cabe destacar que, por cuestiones de simplicidad, las guías (instrucciones de asistencia móvil) para llegar a cada posición son imágenes estáticas que indican cómo llegar a una sala determinada del museo, pero no especifican un punto específico de cada sala; en la práctica, el formato de la guía podría ser tan complejo y detallado como sea necesario, por ejemplo, mediante el uso de herramientas existentes para visualización de mapas, como Google Maps[5] u OpenStreetMap[6], con características de zoom, cálculo dinámico de caminos, etcétera.

Ambos prototipos comparten el siguiente flujo de guías a medida que los usuarios avanzan en la experiencia. Las Figuras 4.1.2, 4.1.3, 4.1.4 muestras las guías correspondientes para dirigirse a las posiciones en el orden adecuado.

[4] http://www.museo.fcnym.unlp.edu.ar/
[5] http://www.google.com/maps
[6] http://www.openstreetmap.org

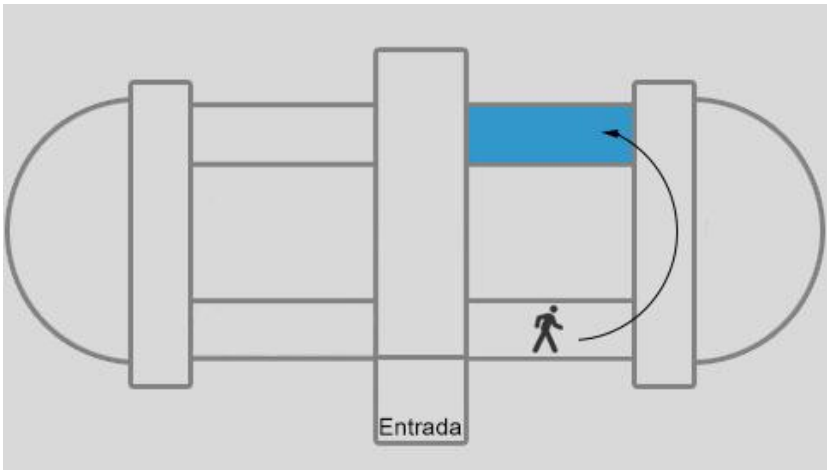

Figura 4.1.2 – Guía para dirigirse desde la posición 1 a la posición 2.

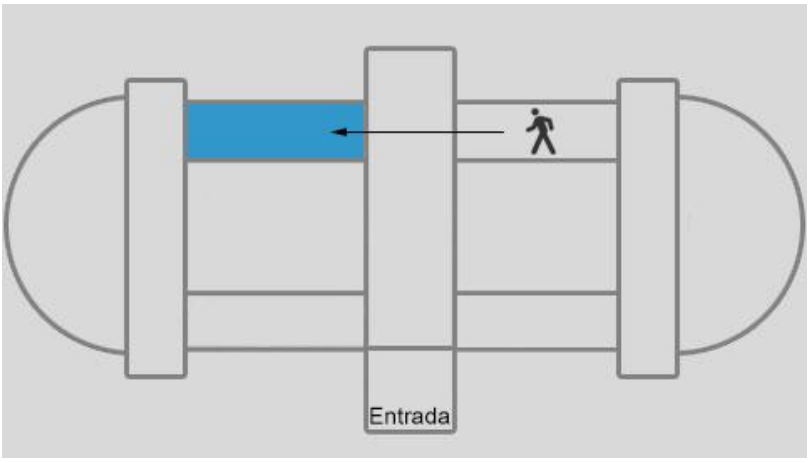

Figura 4.1.3 – Guía para dirigirse desde la posición 2 a la posición 3.

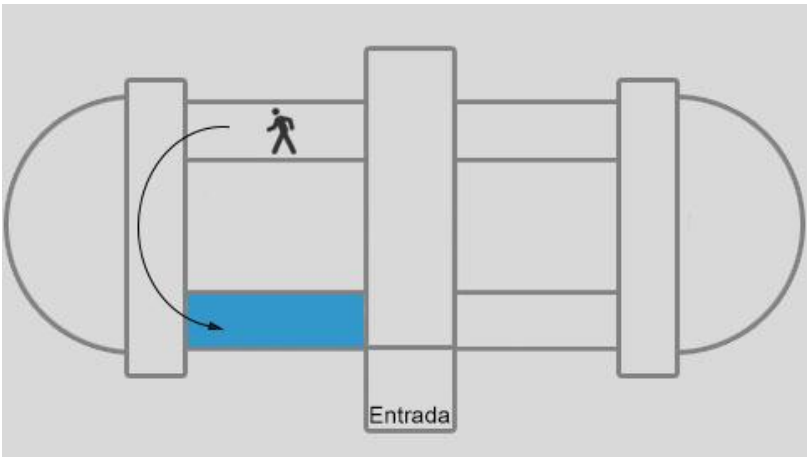

Figura 4.1.4 – Guía para dirigirse desde la posición 3 a la posición 4.

Para contemplar los casos en los que el usuario llega a alguna posición que no coincide con el destino señalado en la guía, se optó por utilizar guías genéricas para indicarle la posición del destino correcto, mostrando directamente la sala destino en el mapa. Nuevamente, esta fue una decisión tomada para este prototipo en particular; en la práctica, podría optarse por generar todas las combinaciones posibles de guías, de manera que las mismas siempre posean un punto de partida definido. Estas guías son utilizadas únicamente en los casos en el que el usuario se equivoca al llegar un destino incorrecto. Las Figuras 4.1.5, 4.1.6, 4.1.7 y 4.1.8 muestran las guías genéricas para llegar a todas las posiciones de la experiencia.

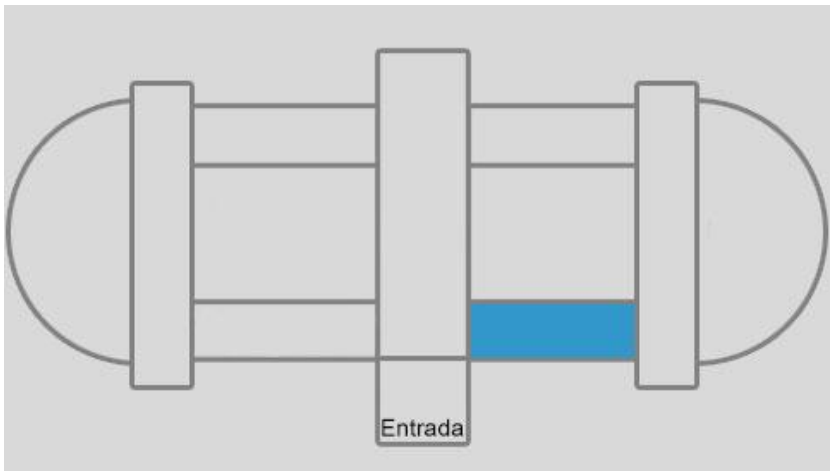

Figura 4.1.5 – Guía para dirigirse a la posición 1.

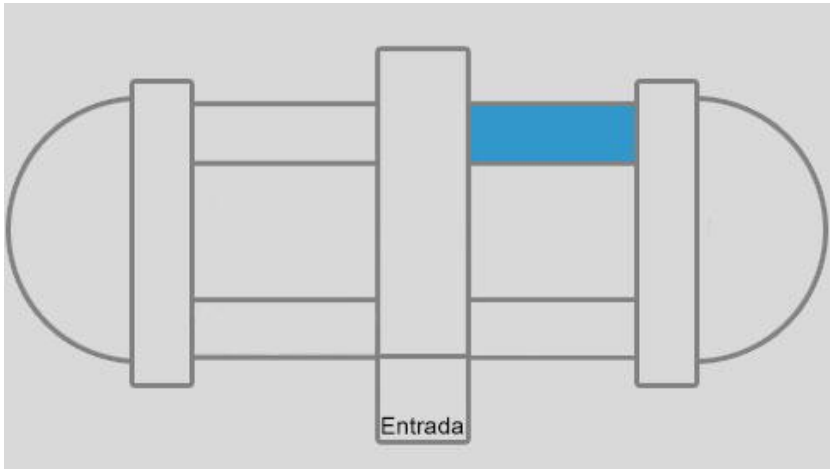

Figura 4.1.6 – Guía para dirigirse a la posición 2.

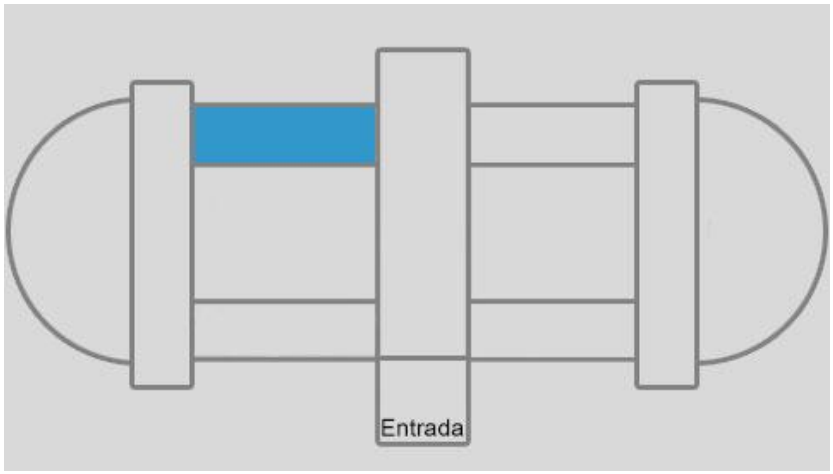

Figura 4.1.7 – Guía para dirigirse a la posición 3.

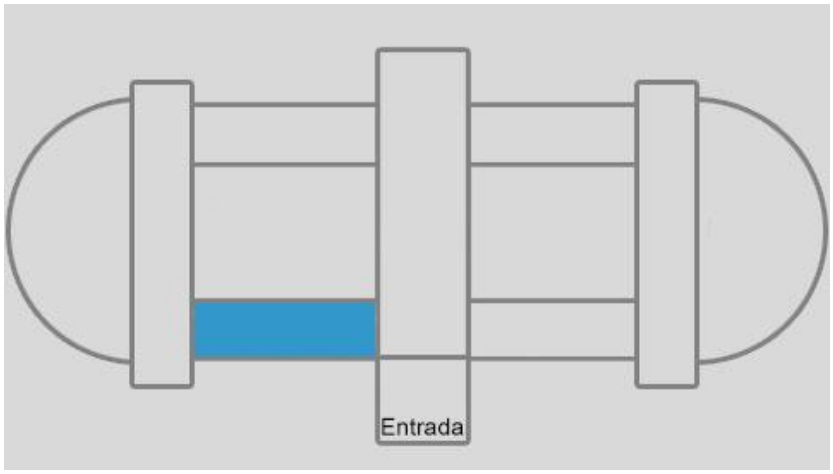

Figura 4.1.8 – Guía para dirigirse a la posición 1.

Por último, para la definición de las reglas de transición de cada funcionalidad, fue necesario hacer uso de conjunto de estados de usuario, acciones de usuario, y condiciones de contexto ya provistas por el modelo de solución presentado en esta tesis. Este conjunto predefinido surge del estudio de los tipos de funcionalidad existentes (como en las metáforas definidas en [Kjeldskov et al., 2007]), por los que abarca los casos más comunes. Aun así, desde el punto de vista del desarrollador, es posible extender el modelo propuesto para definir nuevos estados, acciones o condiciones, según sea necesario. A continuación se lista el conjunto de estados, acciones y condiciones, soportados y provistos (como clases concretas) actualmente por el modelo:

- **Estados de usuario/participante:**
  - Recientemente unido a la experiencia.
  - Visualizando un contenido.
  - Caminando hacia una posición.
  - Seleccionando próximo destino.
  - Experiencia finalizada.

- **Acciones de usuario/participante:**
  - Comenzar experiencia.
  - Finalizar visualización de un contenido.
  - Llegar a una posición.
  - Seleccionar próximo destino.

- **Condiciones de contexto:**
  - Existe al menos un siguiente (*con respecto a un contenido*).
  - Existe exactamente un siguiente (*con respecto a un contenido*).
  - Existen varios siguientes (*con respecto a un contenido*).
  - No existe un siguiente (*con respecto a un contenido)*.
  - La posición a la que se llegó coincide con el destino señalado.
  - La posición a la que se llegó no coincide con el destino señalado.
  - La visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente.
  - La visualización marcada como finalizada no coincide con el contenido que se estaba visualizando anteriormente.

Para instanciar las reglas de transición necesarias para implementar la funcionalidad de una experiencia, se utilizó el formato XML. A continuación se describirán las etiquetas utilizadas en dicha representación, para ambos prototipos.

- ***<stateTransitionRules>*:** contiene todas las reglas que definen la funcionalidad de una Aplicación Móvil basada en Posicionamiento.

- ***<stateTransitionRule>***: contiene la definición para una única regla de transición. Dentro de ella, se encuentran todos los componentes que la definen, como el estado origen del usuario y la acción realizada por el mismo.

- ***<sourceState>*:** especifica el estado origen del usuario con el que se relaciona la regla. Junto con la acción realizada, identifica una regla.

- ***<triggeredByAction>*:** representa la acción realizada por el usuario. Junto con el estado origen, identifica una regla

- ***<assertions>*:** contiene las diferentes aserciones o "predicados" que puede tomar esta regla de transición; cada aserción define una serie de condiciones que deben cumplirse para que sea aplicable, y el estado resultante en caso de que lo sea.

- ***<conditions>*:** especifica las condiciones (cero, una, o varias) necesarias para que la aserción que la contiene sea considerada "verdadera". Cada condición se especifica mediante la sub-etiqueta ***<condition>***.

- ***<resultingState>*:** especifica el estado resultante para una aserción, en caso de que esta sea verdadera. Recordar que, para una regla en particular, sólo una de sus

aserciones puede ser verdadera; esto significa que, de todas las aserciones que componen una regla, sólo una de ellas tendrá todas sus condiciones *verdaderas* al momento de evaluar la regla, y es justamente aquella que definirá el estado resultante del usuario.

4.2 Tecnologías utilizadas

Para el desarrollo de ambos prototipos se utilizó el lenguaje Java para la implementación del modelo de objetos; para la aplicación Web que hace uso del modelo se utilizó el framework Spring[7], junto con el módulo SpringMVC[8], de manera de implementar una arquitectura de servicios, como la descripta en [Fowler et al., 2002]; con esta arquitectura, si bien el modelo está implementado en Java[9], la funcionalidad del mismo puede ser invocada mediante requerimientos HTTP. Para las páginas HTML en sí, se utilizó la tecnología JSP[10] (Java Server Pages), la cual permite embeber código Java en las páginas HTML, las cuales son compiladas antes de desplegar la aplicación; esto permite la manipulación de objetos Java dentro de las páginas Web. Finalmente, como mecanismo de sensado de los prototipos, se utilizaron Códigos QR asociados a los puntos de interés de la experiencia. En la Figura 4.2.1 se muestra la arquitectura general de los prototipos implementados.

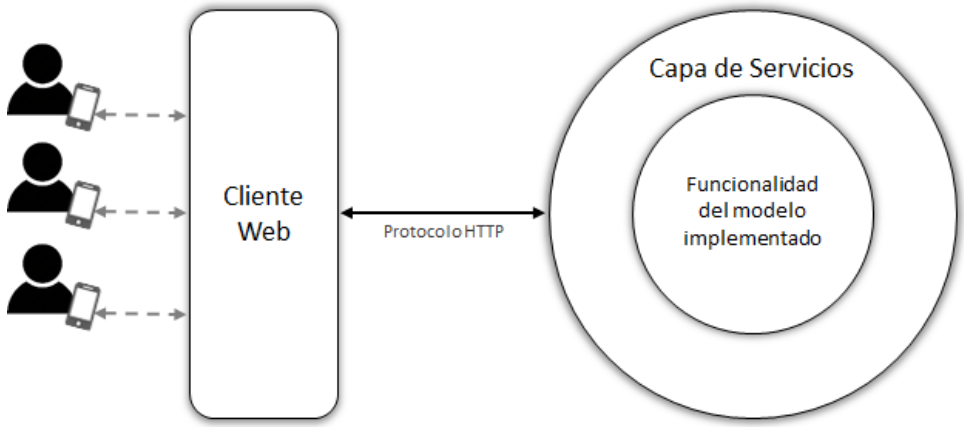

Figura 4.2.1 – Arquitectura general de los prototipos implementados.

Los requerimientos del dispositivo para utilizar ambos prototipos son:

- Acceso a Internet (vía WiFi, 3G, 4G, etcétera).
- Navegador Web (Firefox, Chrome, Opera, el navegador nativo del sistema operativo del dispositivo).
- *(Opcional)* Cámara fotográfica (para la captura de códigos QR).
- *(Opcional)* Aplicación de lectura de códigos QR.

[7] http://projects.spring.io/spring-framework/
[8] http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/mvc.html
[9] http://www.java.com
[10] http://www.oracle.com/technetwork/java/javaee/jsp/index.html

Los prototipos implementados permiten simular la lectura de códigos QR mediante una página HTML, para facilitar las pruebas en computadoras de escritorio o dispositivos sin los requerimientos opcionales. En la Figura 4.2.2 se muestra la sección en común para ambos prototipos de la página principal, en donde se muestran las opciones para simular la lectura de códigos QR, tanto para comenzar la experiencia como para simular la llegada a una posición en particular. En la funcionalidad de ambos prototipos, la lectura del código del primer contenido coincide con la acción de *Comenzar experiencia.*

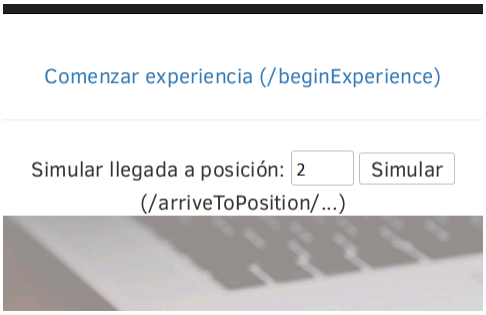

Figura 4.2.2 – Página principal de ambos prototipos, desde donde se pueden simular las lecturas de códigos QR necesarias.

4.3 Prototipo 1 – Funcionalidad lineal

Como prototipo inicial, se implementó una Aplicación Móvil basada en Posicionamiento con funcionalidad *lineal*. Esta funcionalidad implica una secuencia preestablecida de contenidos que los usuarios deben visitar para finalizar la experiencia. En este caso, el prototipo posee cuatro contenidos de texto (como se mencionó en la Sección 4.1), asociados con los correspondientes puntos de interés y posiciones (códigos QR). En la Figura 4.3.1 se muestra la estructura de la experiencia del Prototipo 1.

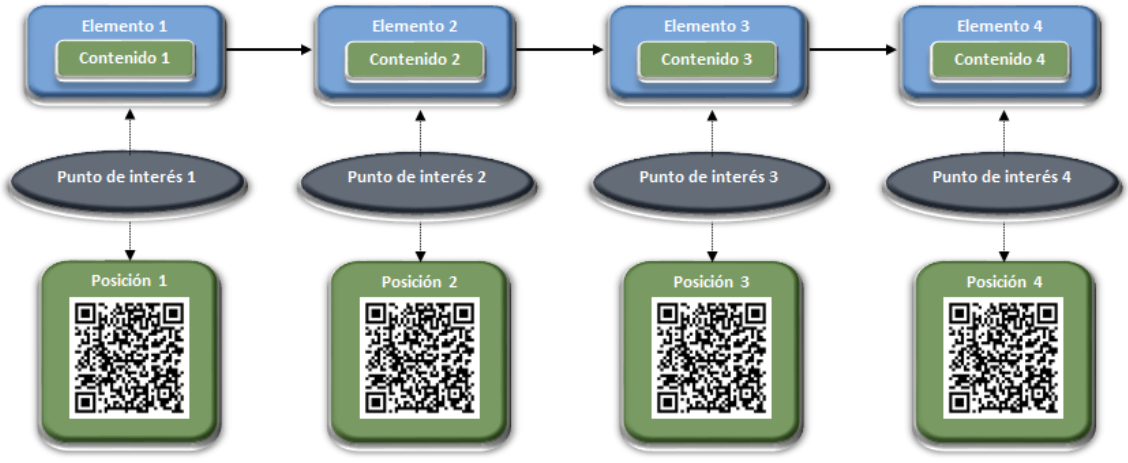

Figura 4.3.1 – Estructura del prototipo con funcionalidad lineal.

Para establecer el tipo de funcionalidad del Prototipo 1, bastó con instanciar las *reglas de transición* (concepto introducido en el Capítulo 3, Sección 2) de manera de reflejar la funcionalidad deseada, con sus respectivas acciones y estados de usuario, y las posibles condiciones de contexto. A continuación, en la Figura 4.3.2 se muestra un Diagrama de Transición de Estados que representa la funcionalidad del Prototipo 1.

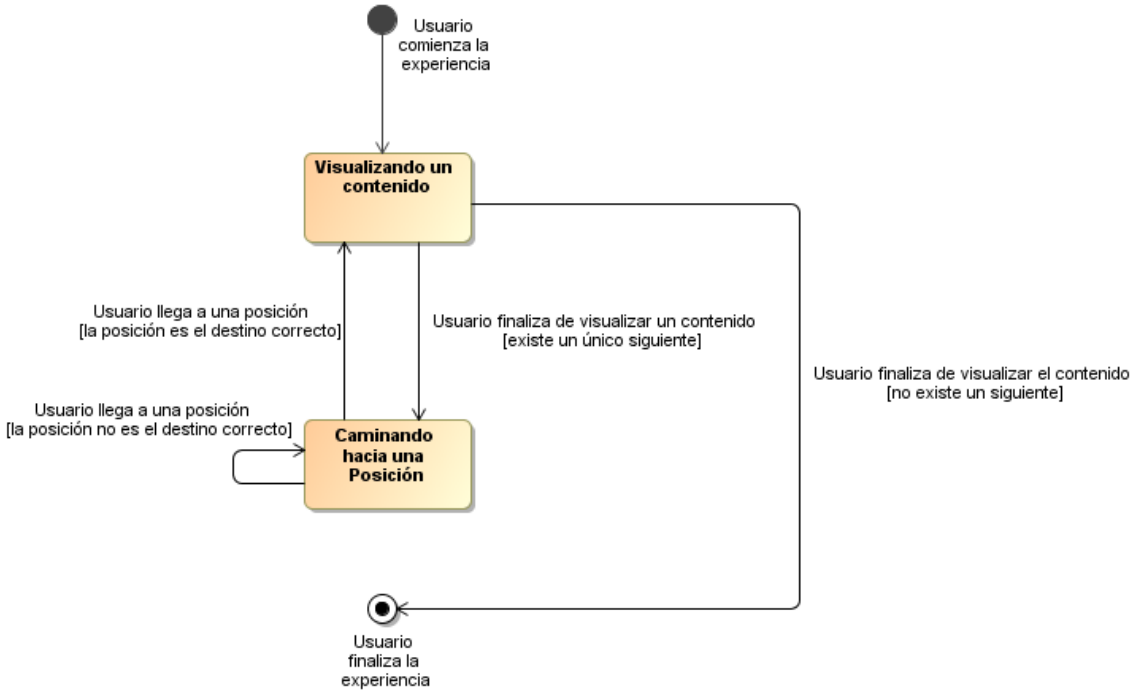

Figura 4.3.2 – Diagrama de Transición de Estados de la funcionalidad del prototipo lineal.

A continuación se detallará cómo se representan las reglas de transición de la funcionalidad lineal del Prototipo 1.

- **Regla 1:**
  - Aplicada a:
    - Estado actual: *Recientemente unido a la experiencia.*
    - Acción realizada: *Comenzar experiencia.*
  - Estado resultante: *Visualizando un contenido.*

  Notar que para esta regla sólo hay definida una única aserción, y la misma no posee condiciones, por lo que existe sólo un estado resultante posible; esto es así ya que cuando una aserción no posee condiciones, se asume verdadera. En el Código 4.3.1 se muestra la representación XML de la Regla 1.

```
<!-- Regla 1 -->
<stateTransitionRule>
   <sourceState>RecentlyJoined</sourceState>

<triggeredByAction>ParticipantBeganExperience</triggeredByAction
>

   <assertions>
      <assert>
         <resultingState>InteractingWithContent</resultingState>
            <conditions>conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.3.1 – Especificación en formato XML de la regla de transición 1.

- **Regla 2:**
  - Aplicada a:
    - Estado actual: *Caminando hacia una posición.*
    - Acción realizada: *Llegar a una posición.*

  - Estado resultante:
    - Caso 1: Si [la posición a la que se llegó coincide con el destino señalado] → *Visualizando un contenido.*

    - Caso 2: Si [la posición a la que se llegó no coincide con el destino señalado] → *Caminando hacia una posición.*

En este caso, la regla especifica más de un posible estado resultante dependiendo de qué aserción sea verdadera al momento de la evaluación. En el primer caso, si el destino es el correcto (es decir, coincide con la posición que se le había indicado al usuario), pasará al estado *Visualizando un contenido*; sin embargo, si el usuario se dirige a una posición que no es la que se le había indicado, la aplicación le indicará nuevamente a dónde debe dirigirse, y pasará a estar nuevamente en el estado *Caminando hacia una posición*. En el Código 4.3.2 se muestra la representación XML de la Regla 2.

```
<!-- Regla 2 -->
<stateTransitionRule>
   <sourceState>WalkingToPosition</sourceState>

<triggeredByAction>ParticipantArrivedToPosition</triggeredByActi
on>

   <assertions>
      <!-- Aserción para destino correcto -->
      <assert>
         <resultingState>InteractingWithContent</resultingState>
         <conditions>
            <condition>DestinationMatch</condition>
```

```
         </conditions>
      </assert>

      <!-- Aserción para destino incorrecto -->
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>
            <condition>DestinationMismatch</condition>
         </conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.3.2 – Especificación en formato XML de la regla de transición 2.

- **Regla 3:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*

  - Estado resultante:
    - Caso 1: Si [la visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente] y [existe exactamente un siguiente] → *Caminando hacia una posición.*

    - Caso 2: Si [la visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente] y [no existe un siguiente] → *Experiencia finalizada.*

    - Caso 3: Si [la visualización marcada como finalizada no coincide con el contenido que se estaba visualizando anteriormente] → *Caminando hacia una posición.*

  Esta regla es la que define principalmente la propiedad *lineal* de la funcionalidad, ya sus dos primeros casos contemplan las situaciones en que existe un siguiente (en tal caso, el usuario pasa a estar *Caminando hacia una posición*) y en la que no existe un siguiente (donde el usuario pasará al estado *Experiencia finalizada*). El Caso 3 fue incluido como consecuencia de las pruebas realizadas: si bien no define la propiedad *lineal* de la aplicación, contempla una situación anómala en la que el usuario marca como finalizado un contenido que no es el que se encontraba visualizando anteriormente; en tal caso, se procede a estar *Caminando hacia una posición*, más precisamente, hacia aquella que contiene el contenido que estaba visualizando originalmente. Se tomó esta decisión para asegurar la correcta contextualización del contenido que se estaba visualizando originalmente (si se hubiera optado por mostrar el contenido directamente –pasando al estado *Visualizando un contenido* en lugar de *Caminando hacia una posición*-, no podría asegurarse que el mismo está siendo visualizado en la posición asociada). El Código 4.3.3 muestra la especificación XML de la Regla 3.

```
<!-- Regla 3 -->
<stateTransitionRule>
   <sourceState>InteractingWithContent</sourceState>
   <triggeredByAction>ParticipantInteractedWithContent
   </triggeredByAction>

   <assertions>
      <!-- Aserción para contenido correcto y existen siguientes
-->
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>
            <condition>ContentMatch</condition>
            <condition>IsInnerElement</condition>
         </conditions>
      </assert>

      <!-- Aserción para contenido correcto y sin siguientes -->
      <assert>
         <resultingState>FinishedExperience</resultingState>
         <conditions>
            <condition>ContentMatch</condition>
            <condition>IsLastElement</condition>
         </conditions>
      </assert>

      <!-- Aserción para contenido incorrecto -->
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>
            <condition>ContentMismatch</condition>
         </conditions>
      </assert>

   </assertions>
</stateTransitionRule>
```

Código 4.3.3 – Especificación en formato XML de la regla de transición 3

- **Regla 4:**
  - Aplicada a:
    - Estado actual: *Caminando hacia una posición.*
    - Acción realizada: *Finalizar visualización de un contenido.*
  - Estado resultante: *Caminando hacia una posición.*

Esta regla representa, al igual que el Caso 3 de la Regla 3, contempla una situación anómala, descubierta durante la etapa de pruebas. Por la naturaleza Web del prototipo desarrollado, se identificó el caso en el cual un usuario que se encontraba caminando hacia una posición podría intentar acceder (mediante la manipulación de la URL en el navegador) a un contenido en cualquier momento, e intentar marcarlo como visto. Para contemplar esta situación, se introdujo esta regla, la cual establece que si el usuario está caminando hacia una posición y, aun así,

realiza la acción de *Finalizar visualización de un contenido*, pasará a estar nuevamente *Caminando hacia una posición*, es decir, aquella a la que debía dirigirse originalmente. En el Código 4.3.4 se muestra la representación XML de la Regla 4.

```
<!-- Regla 4 -->
<stateTransitionRule>
   <sourceState>WalkingToPosition</sourceState>
   <triggeredByAction>ParticipantInteractedWithContent
   </triggeredByAction>

   <assertions>
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.3.4 – Especificación en formato XML de la regla de transición 4.

- **Regla 5:**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Llegar a una posición.*
  - Estado resultante: *Visualizando un contenido.*

De manera similar a Regla 4, esta regla contempla otra situación anómala, en la cual el usuario se encuentra *Visualizando un contenido*, pero realiza la acción de *Llegar a una posición* sin haber finalizado la visualización del contenido explícitamente; en tal caso, se le muestra nuevamente el contenido que estaba visualizando anteriormente, para que pueda marcar su visualización como finalizada, sin necesidad de caminar hacia el punto anterior. En el Código 4.3.5 se muestra la representación XML de la Regla 5.

```
<!-- Regla 5 -->
<stateTransitionRule>
   <sourceState>InteractingWithContent</sourceState>

<triggeredByAction>ParticipantArrivedToPosition</triggeredByActi
on>

   <assertions>
      <assert>
         <resultingState>InteractingWithContent</resultingState>
         <conditions>conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.3.5 – Especificación en formato XML de la regla de transición 5.

Notar que las reglas o casos que contemplan situaciones anómalas también amplían la funcionalidad de la experiencia, dándole así más robustez. Las cinco reglas mencionadas anteriormente dan forma a la funcionalidad lineal, tal como se la describe con el Diagrama de Transición de Estados de la Figura 4.3.2. Cabe recordar que cada una de estas reglas se encuentra contenido dentro de una etiqueta <*stateTransitionRules*>, especificando así que deben evaluarse en conjunto para dar forma a la funcionalidad de una Aplicación Móvil basada en Posicionamiento. El Código 4.3.6 ejemplifica el uso de la etiqueta general para encapsular un conjunto de reglas.

```
<!-- Reglas de transición para el prototipo lineal -->
<stateTransitionRules>

   Código 4.1
   Código 4.2
   Código 4.3
   Código 4.4
   Código 4.5

</stateTransitionRules>
```

Código 4.3.6 – Especificación en formato XML del conjunto de reglas.

Una vez detalladas las reglas de transición que definen a la funcionalidad lineal, se puede mostrar la relación entre dichas reglas y el Diagrama de Transición de Estados de la Figura 4.3.2, el cual representa la funcionalidad lineal. En la Figura 4.3.3 se muestra el mismo diagrama, esta vez detallando la relación que existe entre los estados y transiciones con las reglas definidas.

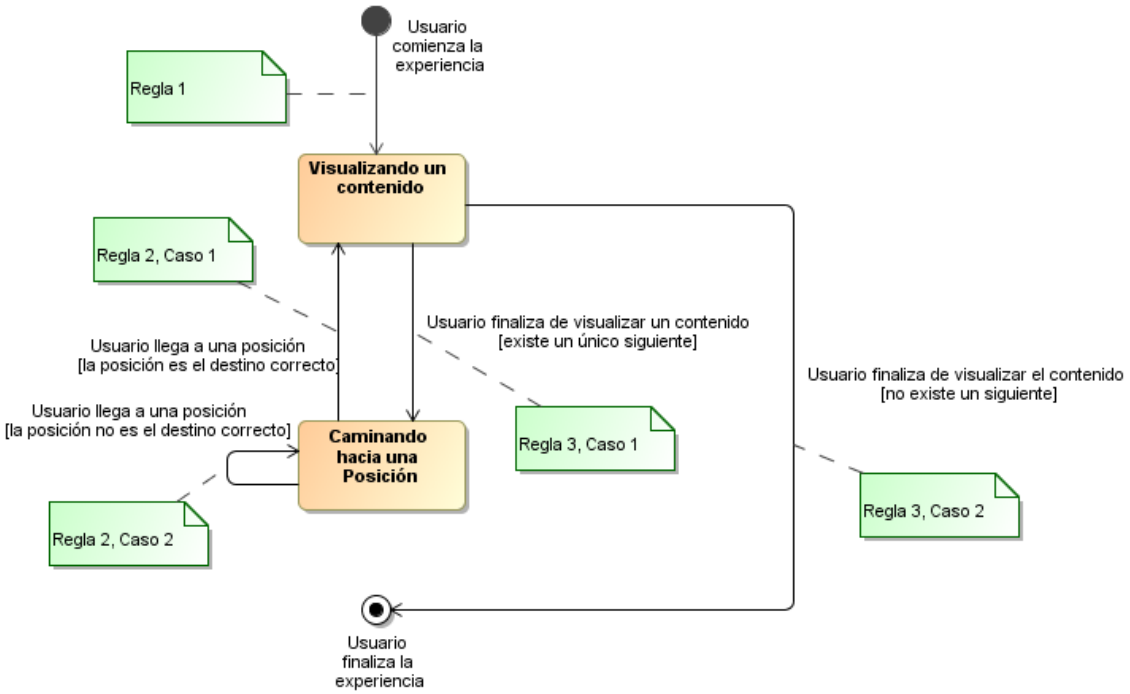

Figura 4.3.3 – Relación entre el DTE de la funcionalidad lineal y las reglas de transición definidas.

Cabe destacar que las reglas (o casos dentro de reglas) que describen situaciones anómalas ajenas a la propiedad lineal de la funcionalidad, como son la Regla 4, la Regla 5, y el Caso 3 de la Regla 3, no aparecen en el Diagrama de Transición de Estados de la figura anterior, ya que no aportarían una descripción semántica de la funcionalidad.

4.4 Prototipo 2 – Funcionalidad de grafo

Como se mostró en el Capítulo 3, basta con modificar el conjunto de reglas de transición de una Aplicación Móvil basada en Posicionamiento para obtener la funcionalidad deseada. A modo de ejemplo, se realizó un segundo prototipo que, mediante un cambio en la definición de las reglas de transición utilizadas en el Prototipo 1, presenta una funcionalidad de *grafo*, en la que existen puntos de interés con más de un posible destino, y es decisión de cada usuario qué camino tomar en cada bifurcación. En este caso, la estructura de la experiencia queda definida según se grafica en la Figura 4.4.1, en la cual se puede apreciar que tanto los contenidos como las posiciones son las mismas que en el Prototipo 1, variando solamente las instancias de los elementos, los cuales definen la estructura de grafo del Prototipo 2.

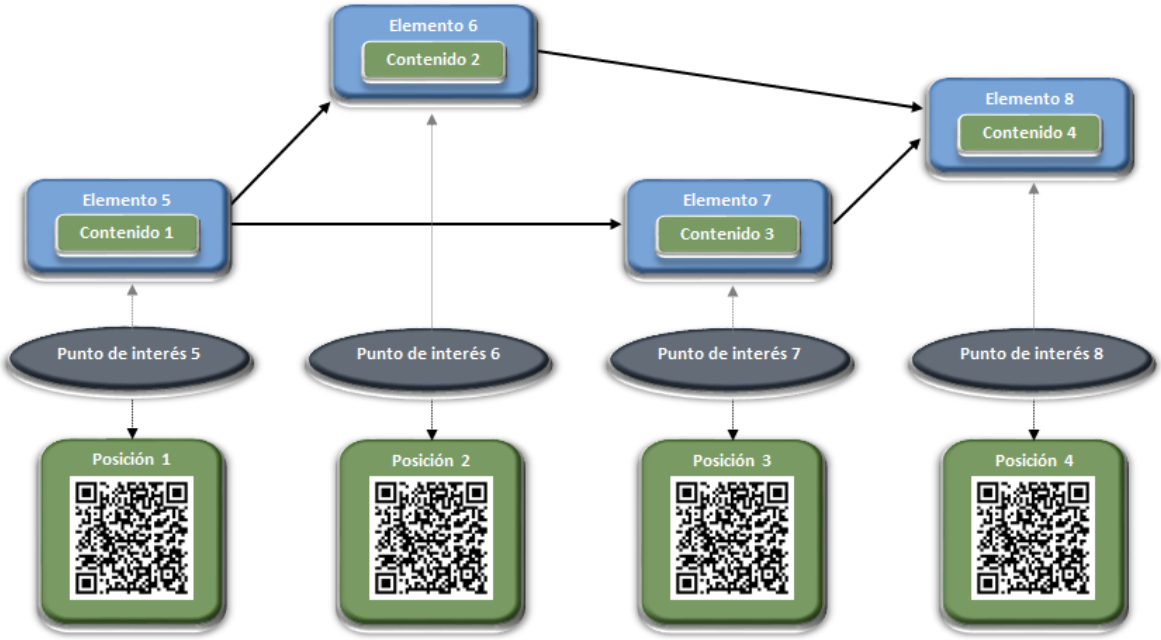

Figura 4.4.1 – Estructura de contenidos y puntos de interés del prototipo con funcionalidad de grafo.

Con esta nueva estructura de elementos, cuando el usuario finalice de visualizar el Contenido 1, podría dirigirse hacia el Contenido 2 o el Contenido 3.

Esta reestructuración de los contenidos de la experiencia requirió la introducción de una nueva guía, la cual indica cómo llegar desde la Posición 2 a la Posición 4. La Figura 4.4.2 muestra esta guía.

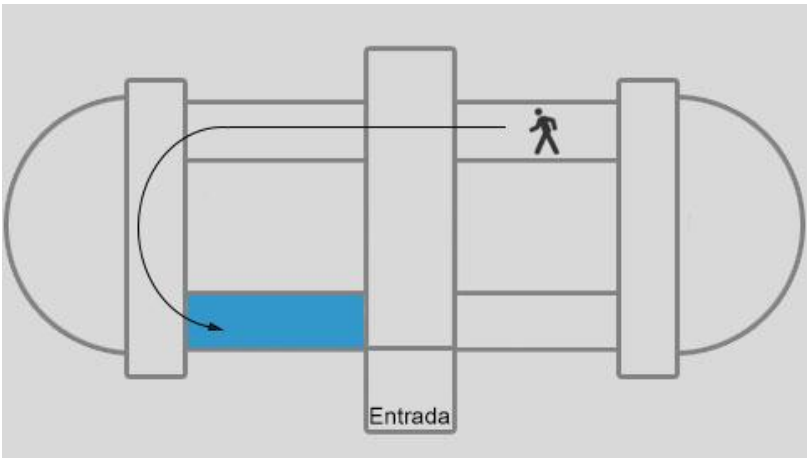

Figura 4.4.2 – Guía para dirigirse desde la posición 2 a la posición 4 (exclusiva para la estructura de grafo del Prototipo 2).

En la Figura 4.4.3 se muestra el Diagrama de Transición de Estados para la funcionalidad de grafo utilizada para el Prototipo 2.

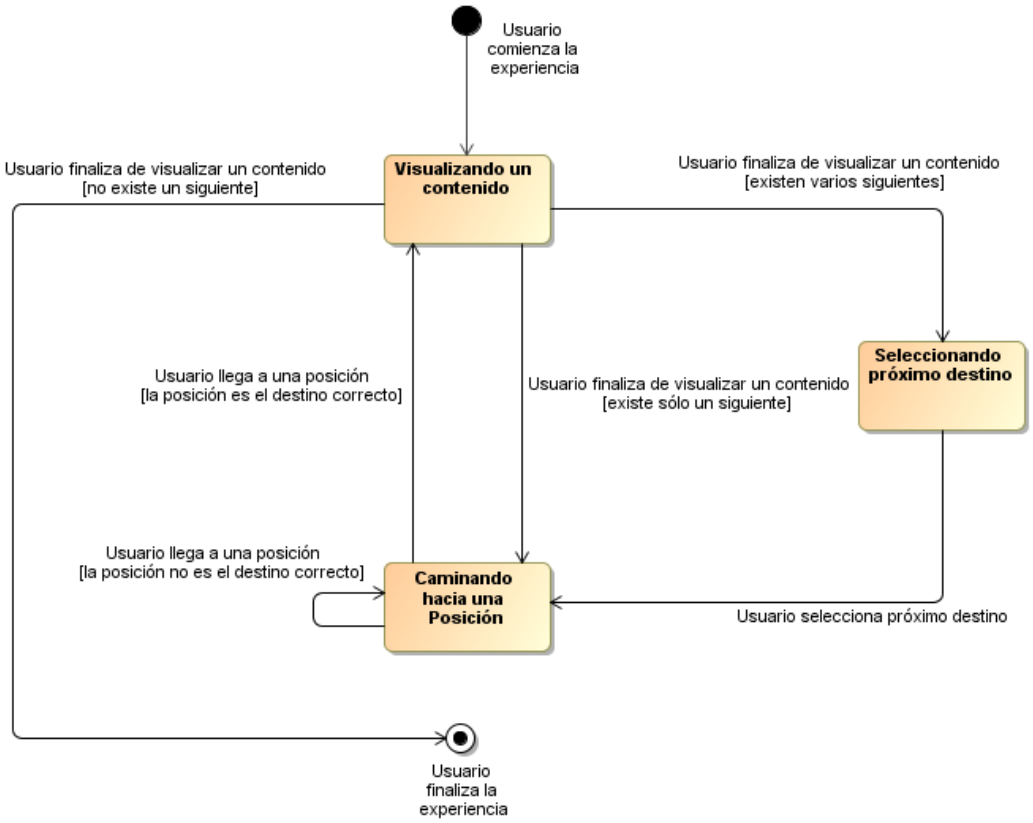

Figura 4.4.3 – Diagrama de Transición de Estados de la funcionalidad del prototipo de grafo.

Notar que, en la caso de la funcionalidad descripta en la Figura 4.4.3, fue necesario hacer uso de estados y acciones de usuario, además de condiciones de contexto, que no fueron

necesarias para describir la funcionalidad lineal; tal es el caso del estado *Seleccionando próximo destino*, la acción *Seleccionar próximo destino,* y la condición *Existen varios siguientes*.

Para lograr la funcionalidad de grafo, se realizaron las modificaciones necesarias al conjunto de reglas del Prototipo 1. Las Regla 1 (Código 4.3.1), Regla 2 (Código 4.3.2), Regla 4 (Código 4.3.4) y Regla 5 (Código 4.3.5) se mantuvieron sin modificaciones; en cuanto a la Regla 3, que indicaba que cuando un usuario finalizaba de visualizar un contenido pasaba a caminar hacia el siguiente destino (si es que hubiera un siguiente), fue modificada para que, cuando existe más de un posible destino siguiente, el usuario pase al estado *Seleccionando próximo destino*. También fue introducida una nueva regla, la Regla 6, la cual contempla la transición entre los estados *Seleccionando próximo destino* y *Caminando hacia una posición*, teniendo en cuenta la nueva acción *Seleccionar próximo destino*.

Las reglas de transición para la funcionalidad de grafo entonces quedan definidas de la misma manera que en la funcionalidad lineal, con las siguientes alteraciones:

- **Regla 3 (modificada):**
  - Aplicada a:
    - Estado actual: *Visualizando un contenido.*
    - Acción realizada: *Finalizar visualización de un contenido.*

  - Estado resultante:
    - Caso 1: Si [la visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente] y [existe exactamente un siguiente] → *Caminando hacia una posición.*

    - Caso 2: Si [la visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente] y [no existe un siguiente] → *Experiencia finalizada.*

    - Caso 3: Si [la visualización marcada como finalizada no coincide con el contenido que se estaba visualizando anteriormente] → *Caminando hacia una posición.*

    - Caso 4 (**nuevo**): Si [la visualización marcada como finalizada coincide con el contenido que se estaba visualizando anteriormente] y [existen varios siguientes] → *Seleccionando próximo destino.*

  Para modificar la Regla 3 y dar la propiedad de *grafo* a la funcionalidad, fue necesario introducir el Caso 4, en el que se contempla la posibilidad de que exista más de un posible destino luego de finalizar la visualización de un contenido. En tal caso, el usuario pasará al estado *Seleccionando próximo destino.* En el Código 4.4.1 se muestra la representación XML de la Regla 3 modificada para la

especificación de la funcionalidad de grafo.

```
<!-- Regla 3 (modificada para grafo) -->
<stateTransitionRule>
   <sourceState>InteractingWithContent</sourceState>
   <triggeredByAction>ParticipantInteractedWithContent
   </triggeredByAction>

   <assertions>
      <!-- Aserción para contenido correcto y un único destino -
->
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>
            <condition>ContentMatch</condition>
            <condition>IsInnerElement</condition>
            <condition>OnlyOnePossibleDestination</condition>
         </conditions>
      </assert>

      <!-- Aserción para contenido correcto y sin siguientes -->
      <assert>
         <resultingState>FinishedExperience</resultingState>
         <conditions>
            <condition>ContentMatch</condition>
            <condition>IsLastElement</condition>
         </conditions>
      </assert>

      <!-- Aserción para contenido incorrecto -->
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>
            <condition>ContentMismatch</condition>
         </conditions>
      </assert>

      <!-- Aserción para contenido correcto y varios destinos -->
      <assert>
         <resultingState>SelectingNextPosition</resultingState>
         <conditions>
            <condition>ContentMatch</condition>
            <condition>IsInnerElement</condition>
            <condition>MultiplePossibleDestinations</condition>
         </conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.4.1 – Especificación en formato XML de la regla de transición 1.

- **Regla 6 (nueva):**
  - Aplicada a:
    - Estado actual: *Seleccionando próximo destino.*
    - Acción realizada: *Seleccionar próximo destino.*
  - Estado resultante: *Caminando hacia una posición.*

  Esta nueva regla se aplica en la situación en la que el usuario ha confirmado cuál será su próximo destino, luego de haberlo seleccionado de un conjunto de posibles. En el Código 4.4.2 se muestra la representación XML de la Regla 6.

```
<!-- Regla 6 (nueva para grafo) -->
<stateTransitionRule>
   <sourceState>SelectingNextPosition</sourceState>
   <triggeredByAction>ParticipantSelectedDestination
   </triggeredByAction>

   <assertions>
      <assert>
         <resultingState>WalkingToPosition</resultingState>
         <conditions>conditions>
      </assert>
   </assertions>
</stateTransitionRule>
```

Código 4.4.2 – Especificación en formato XML de la regla de transición 1.

La relación entre las reglas y el Diagrama de Transición de Estados de la funcionalidad de grafo entonces queda establecida según la Figura 4.4.4. Notar que, nuevamente, las reglas y casos que contemplan situaciones anómalas que no definen la semántica de la funcionalidad de grafo, no aparecen en el diagrama por cuestiones de legibilidad.

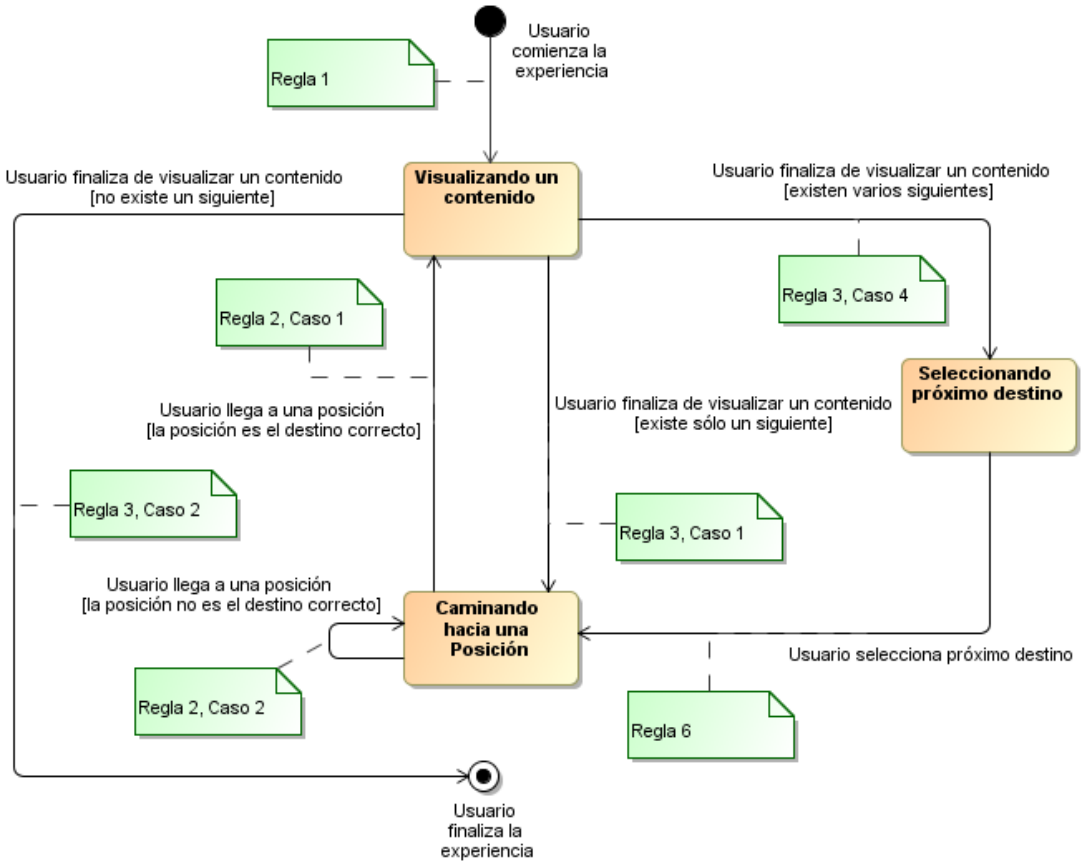

Figura 4.4.4 – Relación entre el DTE de la funcionalidad de grafo y las reglas de transición definidas.

De esta manera, puede observarse el bajo costo y esfuerzo que significa implementar dinámicamente la funcionalidad de una Aplicación Móvil basada en Posicionamiento mediante la definición o modificación de sus reglas de transición, y, en caso de ser necesario, la representación de nuevos estados o acciones de usuario o nuevas condiciones de evaluación.

5. Ejemplo de uso

En este capítulo se mostrarán los resultados de poner en práctica los prototipos descriptos en el Capítulo 4. Para las pruebas realizadas se utilizó un teléfono celular Samsung I9070 (*Galaxy S Advance*), con acceso a Internet, un explorador Web y un lector de códigos QR.

5.1 Prototipo 1 – Funcionalidad lineal

Siguiendo la funcionalidad descripta en el Diagrama de Transición de Estados de la Figura 4.3.2 (Capitulo 4), el usuario, al leer el código QR del primer punto de interés (o simular su lectura mediante el enlace *Comenzar experiencia* –ver Figura 4.2.2, Capítulo 4-), se convertirá en participante de la experiencia, y pasará a visualizar el primer contenido de la misma.

Cuando el usuario comienza la experiencia, pasa a ser un participante de la misma y a visualizar el primer contenido. En este momento, su estado será *Visualizando un contenido*, y habrá llegado a él mediante la aplicación de la Regla 1 (Código 4.3.1, Capítulo 4). En la Figura 5.1.1 se puede apreciar el contexto de la regla de transición aplicada cuando un usuario da comienzo a la experiencia.

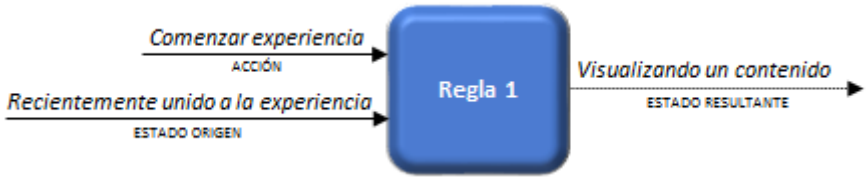

Figura 5.1.1 – Contexto de la regla de transición al comenzar la experiencia.

En la Figura 5.1.2 se muestra la pantalla del prototipo mostrada al usuario luego de ejecutar la Regla 1.

Figura 5.1.2 – Visualización del primer contenido de la experiencia.

El participante, para indicar que ha terminado de visualizar el contenido, deberá presionar el botón *"Siguiente"*. En este caso, como la funcionalidad es lineal, existirá a lo sumo un único destino posible; en el contexto de este prototipo, como existe un siguiente (el Contenido 2), se le mostrará el mapa indicado para llegar a la posición correspondiente. En la Figura 5.1.3 se muestra el contexto de la Regla 3, Caso 1 (Código 4.3.3, Capítulo 4), la cual se aplica cuando un usuario finaliza correctamente la visualización de un contenido y existe un único siguiente.

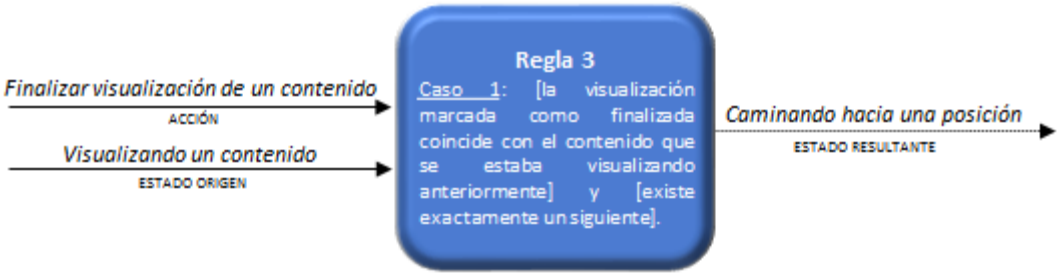

Figura 5.1.3 – Contexto de la regla de transición al finalizar la visualización un contenido.

En la Figura 5.1.4 se muestra la pantalla del prototipo mostrada al usuario luego de ejecutar la Regla 3, Caso 1.

Figura 5.1.4 – Visualización de la guía hacia el siguiente destino, luego de interactuar con el Contenido 1.

Para indicar que llegó a un punto de interés/posición, el participante deberá leer el código de QR correspondiente (o utilizar la página inicial del prototipo, mostrada en la ver Figura 4.2.2, Capítulo 4, para simularlo), y, en caso ser el destino correcto, se le mostrará el contenido correspondiente. En este caso, se aplica la Regla 2, Caso 1 (Código 4.3.2, Capítulo 4) en donde se contempla la llegada del participante a la posición correcta, en cuyo caso pasará al estado *Visualizando un contenido*. En la Figura 5.1.5 se muestra el contexto aplicado a la regla de transición mencionada.

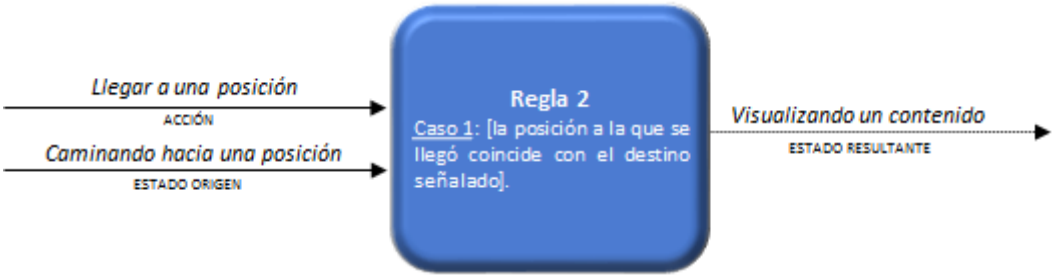

Figura 5.1.5 – Contexto de la regla de transición al llegar al destino correcto.

En la Figura 5.1.6 se muestra la pantalla del prototipo mostrada al usuario luego de ejecutar la Regla 2, Caso 1.

Figura 5.1.6 – Visualización del segundo contenido, una vez que se ha llegado a la posición adecuada.

De la misma manera que en el caso anterior, una vez que el participante finalice de visualizar el Contenido 2 (presionando el botón "*Siguiente"*), se le mostrará un mapa para guiarlo hacia la siguiente posición. En este caso, se aplica nuevamente la Regla 3 (Código 4.3.3, Capítulo 4), y el contexto coincide con el descripto anteriormente en la Figura 5.1.3. La pantalla que visualiza el participante como resultado de esta interacción se muestra en la Figura 5.1.7.

Figura 5.1.7 – Visualización de la guía hacia la posición del Contenido 3.

Si en algún momento el participante llegara a una posición que no corresponde con el destino indicado, la aplicación le indicará nuevamente que debe dirigirse al destino señalado, mostrándole la guía para llegar a él. En este caso, se aplicaría la Regla 2, Caso 2 (Código 4.3.2, Capítulo 4), la cual contempla la situación en la que el participante se encuentra *Caminando hacia una posición*, pero llega a un destino incorrecto. En la Figura 5.1.8 se muestra el contexto de la regla aplicada.

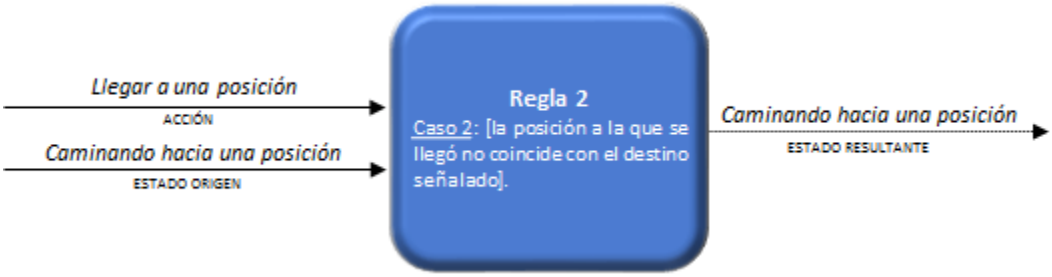

Figura 5.1.8 – Contexto de la regla de transición al llegar a un destino incorrecto.

En la Figura 5.1.9 se muestra pantalla del participante al llegar a un punto de interés que no coincide con el destino indicado, luego de aplicar la Regla 2, Caso 2.

Figura 5.1.9 – Al llegar a una posición incorrecta, la aplicación le indica nuevamente el camino al participante.

Esta secuencia de *caminar, leer el código QR,* y *visualizar el contenido* se repite hasta que el usuario haya recorrido todos los contenidos definidos para esta experiencia. En el caso del prototipo lineal, el usuario deberá visualizar el Contenido 3, dirigirse a la siguiente posición, y finalmente visualizar el Contenido 4. En las Figuras 5.1.10, 5.1.11 y 5.1.12 se muestra la secuencia de pantallas del participante hasta llegar al Contenido 4, asumiendo que se leyeron todos los códigos QR adecuados para cada caso.

Figura 5.1.10 – Visualizando el Contenido 3.

Figura 5.1.11 – Guía para la Posición 4.

Figura 5.1.12 – Visualizando el Contenido 4.

Cuando el participante finaliza la visualización del último contenido de la experiencia (en este caso, el Contenido 4), se le informa que ha finalizado la experiencia. Al mostrar esta pantalla, el participante se encontrará en el estado *Experiencia finalizada*, y habrá llegado a él mediante la aplicación del Caso 3 de la Regla 3 (Código 4.3.3, Capítulo 4). En la Figura 5.1.13 se muestra le contexto de la regla aplicada.

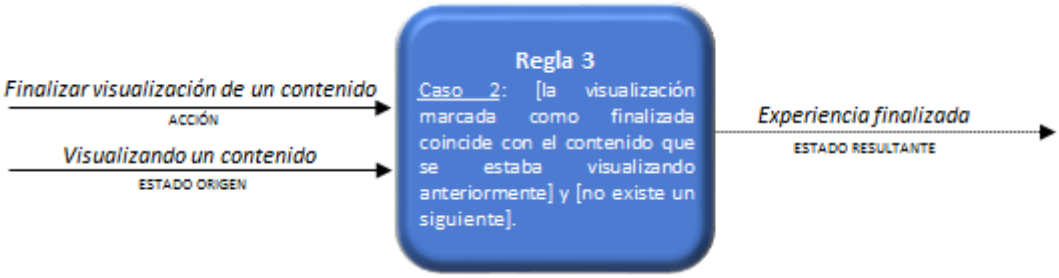

Figura 5.1.13 – Contexto de la regla de transición al visualizar el último contenido de la experiencia.

En la Figura 5.1.14 se muestra pantalla del participante al finalizar la visualización del último contenido de la experiencia, luego de aplicar la Regla 3, Caso 2.

Figura 5.1.14 – Mensaje mostrado cuando el participante finaliza de visualizar el último contenido.

Hasta aquí se ha mostrado en marcha el Prototipo 1 (correspondiente a la funcionalidad lineal), junto con los posibles flujos seguidos por los participantes. En la sección siguiente, se mostrará de la misma manera el Prototipo 2, correspondiente a la funcionalidad de grafo.

5.2 Prototipo 2 – Funcionalidad de grafo

En esta sección se mostrará el Prototipo 2, correspondiente a la funcionalidad de grafo, en funcionamiento. Dicha funcionalidad fue creada de acuerdo a las especificaciones realizadas anteriormente en la Sección 4.4 del Capítulo 4.

Al inicio de la experiencia, el Prototipo 2 se comporta igual al Prototipo 1: el usuario se une como participante de la experiencia, y luego pasa a visualizar el Contenido 1. En este caso, al igual que en el Prototipo 1, se aplica la Regla 1 (Código 4.3.1, Capítulo 4), la cual se mantiene sin modificaciones para ambas funcionalidades. En la Figura 5.2.1 se muestra el contexto de dicha regla al momento de ser aplicada.

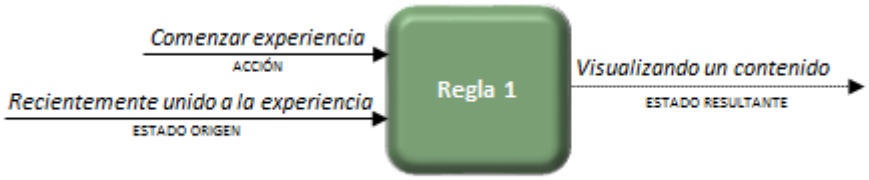

Figura 5.2.1 – Contexto de la regla de transición al comenzar la experiencia.

En la Figura 5.2.2 se muestra la visualización del Contenido 1, luego de dar comienzo a la experiencia, aplicando la Regla 1.

Figura 5.2.2 – Visualización del Contenido 1.

Como se mencionó anteriormente, luego de visualizar el Contenido 1, el participante podría continuar por varios caminos, por lo que, en lugar de visualizar la guía hacia el próximo destino como lo hacía con la funcionalidad lineal, se le presentará una pantalla de selección de su próximo destino, donde deberá indicar hacia dónde desea dirigirse; durante esta etapa, el participante se encontrará en el estado *Seleccionando próximo destino*. Esta situación se contempla en el Caso 4 de la Regla 3 (Código 4.4.1, Capítulo 4), el cual fue agregado específicamente para representar esta característica particular de la funcionalidad de grafo. En la Figura 5.2.3 se muestra el contexto de dicha regla al momento de aplicarla.

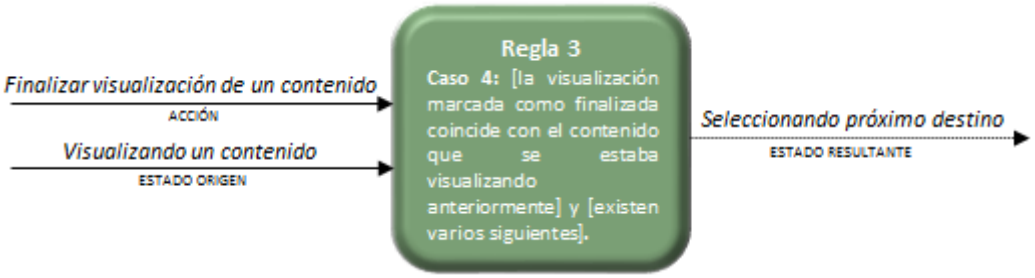

Figura 5.2.3 – Contexto de la regla de transición cuando el usuario debe seleccionar su próximo destino.

Cabe destacar que la forma en que el usuario selecciona su próximo destino puede ser modificada; en el contexto de este prototipo se optó por mostrar simplemente una lista con los nombres de las posiciones (*Posición 2* y *Posición 3*) y un elemento HTML de selección única (*radio-button*). Esto no implica que pueda optarse por mostrar, por ejemplo, los mapas hacia cada uno de los destinos posibles, y que el participante indique cuál desea mediante un clic en el destino elegido. En la Figura 5.2.4 se muestra la pantalla de selección de próximo destino visualizada luego de aplicar la Regla 3, Caso 4.

Figura 5.2.4 – Al finalizar la visualización de un contenido, si existe más de un posible destino, se le muestran al participante las opciones, para que seleccione una de ellas.

Como se mencionó anteriormente, la forma de interacción para este prototipo requiere que el participante seleccione la opción que desea como próximo destino, y luego presione el botón *Seleccionar destino*. A partir de aquí, la experiencia del participante se desarrollará según su selección, ya que los contenidos mostrados dependerán de las posiciones que visite. En primer lugar, se mostrará el flujo de la experiencia cuando el usuario selecciona la Posición 2 como su próximo destino. Al final de esta sección se mostrará el flujo análogo a la selección de la Posición 3. En la Figura 5.2.5 se muestra entonces la opción *Posición 2* seleccionada, justo antes de presionar el botón *Seleccionar destino* para confirmar la elección.

Figura 5.2.5 – Pantalla de selección de próximo destino, con la Posición 2 seleccionada.

Una vez que el participante confirma su elección del próximo destino, se le mostrará una guía con el mapa para llegar a la posición correspondiente, y pasará al estado *Caminando hacia una posición*. Esta interacción está contemplada por la Regla 6 (Código 4.4.2, Capítulo 4), la cual fue introducida específicamente para representar la funcionalidad de

grafo. En la Figura 5.2.6 se muestra el contexto al momento de aplicar dicha regla.

Figura 5.2.6 – Contexto de la regla de transición cuando el usuario selecciona su próximo destino.

En la Figura 5.2.7 se muestra la pantalla del usuario en donde se le indica cómo llegar al destino seleccionado, luego de aplicar la Regla 6.

Figura 5.2.7 – Guía mostrada al seleccionar la Posición 2 como próximo destino.

Una vez que el participante llega a la posición indicada en el mapa, se procede normalmente a visualizar el contenido correspondiente, tal como se hacía en el Prototipo 1. El participante pasará nuevamente al estado *Visualizando un contenido*, como resultado de aplicar el Caso 1 de la Regla 2 (Código 4.3.2, Capítulo 4), la cual no necesitó modificaciones para ser aplicada en ambas funcionalidades. En la Figura 5.2.8 se muestra el contexto de dicha regla al momento de ser aplicada.

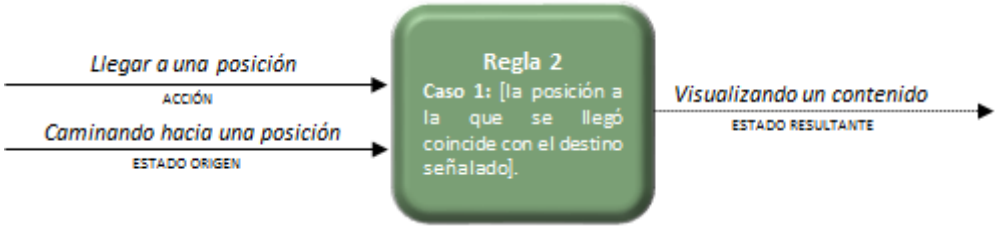

Figura 5.2.8 – Contexto de la regla de transición cuando el usuario selecciona su próximo destino.

En la Figura 5.2.9 se muestra la pantalla en donde el usuario se encuentra visualizando el contenido correspondiente luego de llegar al destino seleccionando anteriormente, luego

de aplicar la Regla 2, Caso 1.

Figura 5.2.9 – Visualización de Contenido 2.

Como se detalló en la Figura 4.4.1 (Capítulo 4), la estructura del Prototipo 2 indica que desde el Contenido 2, el participante debe dirigirse hacia el Contenido 4. Esto significa que, una vez finalizada la visualización del Contenido 2, se le mostrará la guía para llegar al Contenido 4, pasando al estado *Caminando hacia una posición*. Esta situación está contemplada por el Caso 1 de la Regla 3 (Código 4.4.1, Capítulo 4), la cual se aplica cuando existe un único siguiente. En la Figura 5.2.10 se muestra el contexto al momento de aplicar dicha regla.

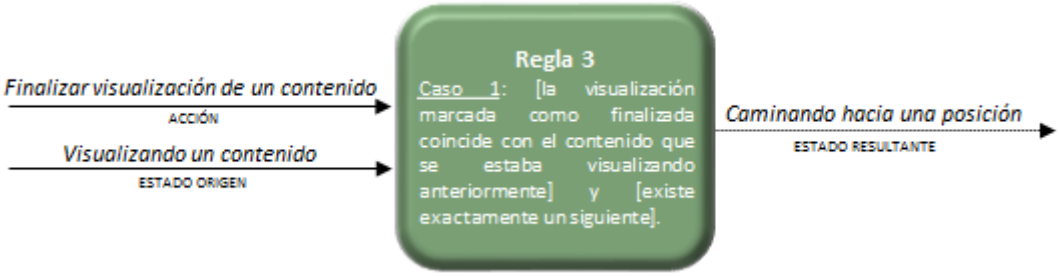

Figura 5.2.10 – Contexto de la regla de transición cuando al finalizar la visualización de un contenido, existe exactamente un siguiente.

En la Figura 5.2.11 se muestra la pantalla en donde se le indica al usuario cómo dirigirse hacia el Contenido 4, luego de aplicar la Regla 3, Caso 1.

Figura 5.2.11 – Guía para llegar desde la Posición 2 a la Posición 4.

Al llegar a la posición indicada, el participante pasará nuevamente al estado *Visualizando un contenido*, como resultado de aplicar el Caso 1 de la Regla 2 (Código 4.3.2, Capítulo 4), en el contexto ya descripto en la Figura 5.2.8. En la Figura 5.2.12 se muestra la pantalla del usuario visualizando el último contenido de la experiencia.

Figura 5.2.12 – Visualización del último contenido de la experiencia.

Por último, cuando el participante marque como finalizada la visualización del Contenido 4, al ser éste el último de la experiencia, pasará al estado *Experiencia finalizada*. Esta situación es contemplada (al igual que en el Prototipo 1), por el Caso 2 de la Regla 3 (Código 4.4.1, Capítulo 4); el contexto de aplicación de dicha regla se muestra en la Figura 5.2.13.

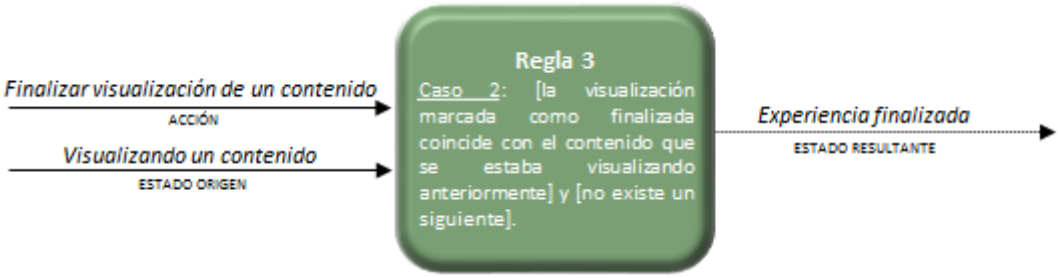

Figura 5.2.13 – Contexto de la regla de transición al visualizar el último contenido de la experiencia.

En la Figura 5.2.14 se muestra la pantalla con el mensaje visualizado por el participante, luego de finalizar la visualización del último contenido de la experiencia, luego de aplicar la Regla 3, Caso 2.

Figura 5.2.14 – Mensaje de fin de experiencia, mostrado luego de visualizar todos los contenidos.

Si bien no aporta nuevos casos o utilizaciones de reglas diferentes, se mostrará el flujo de la experiencia si el participante, en el momento en que debe seleccionar su próximo destino (luego de visualizar el Contenido 1), selecciona la Posición 3 en lugar de la Posición 2. Las Figuras 5.2.15, 5.2.16, 5.2.17, 5.2.18, 5.2.19 y 5.2.20 describen la misma secuencia que fue detallada a lo largo de esta sección, pero siguiendo el camino de la Posición 3.

Figura 5.2.15 – Pantalla de selección de próximo destino, con la Posición 3 seleccionada.

Figura 5.2.16 – Guía mostrada al seleccionar la Posición 3 como próximo destino.

Figura 5.2.17 – Visualización de Contenido 3.

Figura 5.2.18 – Guía para llegar desde la Posición 3 a la Posición 4.

Figura 5.2.19 – Visualización del último contenido de la experiencia.

Figura 5.2.20 – Mensaje de fin de experiencia, mostrado luego de visualizar todos los contenidos.

6. Publicaciones realizadas

En relación a los conceptos introducidos en este trabajo, se han realizado dos publicaciones. Las mismas serán descriptas brevemente en este capítulo, y podrán ser consultadas en su documento completo en el Anexo I.

- **Authoring Tool for Location-Aware Experiences** ([Alconada et al., 2015a]).
  En esta publicación presenta una herramienta visual para la generación *in situ* de contenidos y posiciones de una Aplicación Móvil basada en Posicionamiento. En relación a esta tesis, el enfoque de esta herramienta se puede combinar con los conceptos de *funcionalidad* y *reglas de transición*, para lograr así una posible generación de todos los aspectos de una Aplicación Móvil basada en Posicionamiento mediante el uso de una herramienta visual, y sin los conocimientos técnicos necesarios en el enfoque de esta tesis (como el formato XML o el lenguaje Java). Esta generación de funcionalidad y reglas de transición *in situ* forma parte de los trabajos futuros (ver Sección 7.2, Capítulo 7).

  Abstract de la publicación (traducción del original): *En este paper presentamos un enfoque para crear experiencias 'location-based'. Utilizamos el concepto de 'separation of concerns' para representar tanto la capa de contenidos y la capa de posicionamiento. Esta separación permite reusar las capas independientemente una de otro. El foco de este paper es proveer una herramienta de usuario final para crear aplicaciones 'location-based' in-situ. A través del uso de esta herramienta, las experiencias 'location-aware' pueden ser definidas tanto en espacios indoor como en espacios outdoor. Presentamos también un ejemplo de cómo nuestra herramienta se utiliza y describimos algunos puntos de discusión que surgieron al definir la herramienta con estas características.*

- **Combing Location-Aware Applications with in-situ Actors Performances** ([Alconada et al., 2015b]).
  Esta publicación introduce los conceptos básicos que fueron abordados en detalle en esta tesis, como los conceptos de *contenidos*, *posiciones*, *puntos de interés*, etcétera, así como la necesidad de desacoplamiento entre ellos. También se presenta un ejemplo de una aplicación desarrollada siguiendo dichos conceptos, la cual fue aplicada en el contexto del TEDx Diagonal73[11].

  Abstract de la publicación (traducción del original): *En este paper se presenta un modelo para definir experiencias 'location-aware' teniendo sus diferentes aspectos (contenidos, posiciones, etc.) desacoplados unos de otros. Creamos una aplicación siguiendo este modelo que se combinó con actuaciones de actores reales. Esta aplicación define tres posiciones relevantes dentro de un edificio específico. Al leer un código QR, el usuario indica que ha llegado a una de esas posiciones. En ese momento, la aplicación muestra algunas preguntas relacionadas a las actuaciones de los actores. Entonces, el usuario debe ver todas las actuaciones para poder*

[11] http://tedxdiagonal73.com.ar/

contestar las preguntas. La aplicación asiste al usuario para moverse a través de las diferentes posiciones. Presentamos también la evaluación de esta aplicación y luego discutimos algunos aspectos interesantes que surgieron como resultado de la evaluación in-situ de la aplicación.

7. Conclusiones y Trabajos Futuros

7.1 Conclusiones

A lo largo de este trabajo se ha presentado un enfoque para la creación de funcionalidad dinámica de Experiencias Móviles basadas en Posicionamiento, utilizando como base el diseño de un modelo orientado a objetos que sea lo suficientemente genérico y escalable como para dar soporte a nuevos tipos de funcionalidad que puedan surgir en cualquier dominio de este tipo de aplicaciones.

En particular, el concepto de *regla de transición* introducido en esta tesis permite dar forma a la funcionalidad de una Aplicación Móvil basada en Posicionamiento, definiendo cada una cómo debe actuar la aplicación ante determinada acción de un usuario, y teniendo en cuenta su estado actual y las condiciones de contexto.

Para facilitar la definición de reglas de transición, el modelo propuesto provee un conjunto predefinido de *estados de usuario*, *acciones de usuario* y *condiciones de contexto*, los cuales son los más representativos en la mayoría de las aplicaciones estudiadas. Esto permite la creación rápida de reglas, haciendo uso de los conjuntos predefinidos por el modelo; sin embargo, esto no quita la posibilidad que tiene un desarrollador de *extender* el modelo para agregar nuevos *estados, acciones,* o *condiciones* no contemplados en el conjunto preestablecido. Para realizar extensiones de este tipo, se utiliza el concepto de herencia del modelo orientado a objetos, por lo que bastaría con extender las clases correspondientes para agregar las nuevas alternativas.

Los prototipos implementados sirven como prueba empírica de que el modelo propuesto satisface las características mencionadas, permitiendo modificar la funcionalidad de una Experiencia Móvil basada en Posicionamiento según sea necesario para cada situación particular, simplemente creando las *reglas de transición* (o modificando las existentes) adecuadas para su representación.

Los conceptos abarcados en este trabajo fueron puestos en práctica con varios prototipos y pruebas internas; quizás la aplicación más destacable de estos conceptos fue la Aplicación Móvil basada en Posicionamiento *"Caminos Alternativos"*, la cual fue utilizada en el contexto del evento TEDx Diagonal73[12].

Como conclusión final puede decirse que el modelo de solución y los conceptos abarcados en este trabajo servirán como base para el desarrollo de futuras Aplicaciones Móviles basadas en Posicionamiento, y eventualmente permitir la evolución e inclusión de nuevos aspectos de su funcionalidad.

[12] http://tedxdiagonal73.com.ar/

7.2 Trabajos Futuros

Como trabajo futuro queda pendiente también la inclusión de un conjunto predefinido de reglas de transición, y la posibilidad de utilizarlas mediante una etiqueta especial que simplifique la necesidad de definir las combinaciones comunes de *estado, acción*, y *condiciones*. Esto surge de la experiencia obtenida a lo largo de este trabajo, en donde se detectó que existen reglas que son recurrentes en varios tipos de funcionalidad. De manera análoga, sería de interés proveer *templates* o *moldes* de funcionalidad. Estos *templates* definirían un conjunto base de reglas de transición que definen una funcionalidad o parte de ella, a partir de la cual los usuarios puedan construir o modificar según lo requieran, agilizando aún más la creación de Aplicaciones Móviles basadas en Posicionamiento.

Queda pendiente también la creación de nuevos prototipos de Aplicaciones Móviles basadas en Posicionamiento en dominios específicos, en particular aquellos que introduzcan combinaciones interesantes de *estado, acción,* y *condición,* con el objetivo de ampliar el soporte del modelo propuesto incluso en aquellos casos no contemplados en el estado actual.

Otro posible trabajo futuro es la implementación de una herramienta visual que permita crear la funcionalidad de una Aplicación Móvil basada en Posicionamiento, destinada a usuarios no expertos. Para esto, la forma de representación de reglas de transición utilizada en este trabajo podría servir como punto de partida para lograr una representación visual de dichas reglas, y eliminar así la necesidad de conocimiento especializado requerido en el estado actual, es decir, dominio del formato XML o el lenguaje Java para poder crear o modificar funcionalidad.

Relacionado al punto anterior, otra rama de investigación posible, es aquella relacionada a la creación de funcionalidad de Aplicaciones Móviles basadas en Posicionamiento orientada a usuarios no expertos; sería deseable también que la creación de reglas de transición para dar forma a una experiencia pueda ser realizada *in situ*, es decir, en el mismo lugar en donde tendrá lugar la aplicación final. De esta manera, los usuarios podrán realizar pruebas en el mismo lugar donde se llevará a cabo la experiencia a la que la aplicación dará soporte, y realizar ajustes según sea necesario. Cabe destacar que los usuarios necesitarán además de una herramienta visual de soporte para la definición de los contenidos de estas aplicaciones, por ejemplo, la ya introducida en [Alconada et al., 2015a]. Para esto es necesario que el enfoque adoptado sea capaz de integrarse a una herramienta de definición de contenidos existente.

Referencias bibliográficas

[Alconada et al., 2015a] Alconada Verzini, F., Tonelli, J., Challiol, C., Lliteras, A., Gordillo, S.: *Authoring Tool for Location-Aware Experiences* – Proceedings of the 2015 Workshop on Narrative & Hypertext, pp. 21-25, ACM.

[Alconada et al., 2015b] Alconada Verzini, F., Tonelli, J. I., Challiol, C., Lliteras, A., Gordillo, S.: *Combining Location-Aware Applications with in-situ Actors Performances* – Proceedings of the 2015 Workshop on Narrative & Hypertext, pp. 27-31, ACM.

[Arsanjani, 2001] Arsanjani, A.: *Rule Object 2001: A pattern language for adaptable and scalable business rule construction* – 8th Pattern Languages of Programs Conference, Illinois, USA, 2001.

[Bouvin et al., 2003] Bouvin, N., Christensen, B., Grønbæk, K., Hansen, F.: *HyCon: a framework for context-aware mobile hypermedia* – New review of hypermedia and multimedia, 2003, Vol. 9, No. 1, pp. 59-88.

[Callaway et al., 2012] Callaway, C., Stock, O., Dekoven, E., Noy, K., Citron, Y., Dobrin, Y.: Mobile drama in an instrumented museum: inducing group conversation via coordinated narratives. - New Review of Hypermedia and Multimedia, 2012, Vol. 18, Nos. 1-2, pp. 37-61.

[DeRose et al., 2001] DeRose, S., Maler, E., Orchard, D.: *XML Linking Language (XLink)* - W3C Recommendation 27 June 2001, W3C, 2001, [http://www.w3.org/TR/xlink/].

[Emmanouilidis et al., 2013] Emmanouilidis, C., Aris Koutsiamanis, R., Tasidou, A.: *Mobile Guides: Taxonomy of Architectures, Context Awareness, Technologies and Applications.* - Journal of Network and Computer Applications, 2013, Vol. 36, pp. 103-125.

[Fowler et al., 2002] Fowler, M., Rice, D., Foemmel, M, Hieatt, E., Mee, R., Stafford, R.: *Patterns of enterprise application architecture* – Editorial Addison-Wesley, Longman Publishing Co., Inc., ISBN-13: 007-6092019909, 2002.

[Gamma et al., 1995] Gamma, E., Helm, R., Johnson R., Vlissides J.: *Design Patterns: Elements of Reusable Object-Oriented Software* - Primera Edición en español, Editorial Addison Wesley, ISBN: 84-7829-059-1, 1995.

[Hansen et al., 2008] Hansen, F., Kortbek, K., Grønbæk, K.: *Mobile Urban Drama – Setting the stage with location based technologies.* - Interactive Storytelling, 2008, pp. 20-31, Springer Berlin Heidelberg.

[Johnson et al., 1996] Johnson, R., Woolf, B.: *The Type Object Pattern - http://www.cs.ox.ac.uk/jeremy.gibbons/dpa/typeobject.pdf*, 1996 [Consulta 30 de junio de 2015].

[Kato et al., 2010] Kato, H., Tan, K., Chai, D.: *Barcodes for Mobile Devices* – ISBN-13: 978-0-521-88839-4, Editorial Cambridge University Press, 2010.

[Kjeldskov et al., 2007] Kjedlskov, J., Paay, J.: *Augmenting the City with Fiction: Fictional Requeriments for Mobile Guides* – Proceedings of Workshop on Mobile Guides, Mobile HCI, 2007.

[Lehman, 1996] Lehman, M.: *Laws of software evolution revisited* - Proceedings of the 5th European Workshop on Software Process Technology, 1996, LNCS 1149, pp. 108-124, Springer-Verlag.

[Lliteras et al., 2012] Lliteras, A., Challiol, C., Gordillo, S.: *Juegos Educativos Móviles Basados en Posicionamiento: Una Guía para su Conceptualización*. – 13th Argentine Symposium on Software Engineering, 41 JAIIO, La Plata, 2012.

[Metro Paris Subway] Metro Paris Subway iPhone and iPod Touch Application. - http://www.metroparisiphone.com [Consulta 2015, 28 de mayo].

[Millard et al., 2013] Millard, D., Hargood, C., Jewell, M., Weal, M.: *Canyons, Deltas and Plains: Towards a Unified Sculptural Model of Laction-Based Hypertext*. – Proceedings of the 24th ACM Conference on Hypertext and Social Media, HT '13, pp. 109-118, ACM, 2013.

[Pittarello et al., 2012] Pittarello, F., Bertani, L.: *CASTOR: learning to create context-sensitive and emotionally engaging narrations in-situ.* - Proceedings of the 11th International Conference of Interaction Design and Children, 2012, pp. 1-10, ACM.

[Roy et al., 2003] Roy, N., Scheepers, H., Kendall, E.: *Mapping the Road for Mobile Systems Development* - Proccedings of Pacific Asia Conference on Information Systems 2003, paper 94, pp. 1358-1371, 2003.

Anexo I: Publicaciones

A continuación se anexarán los documentos originales de las publicaciones realizadas, por estar relacionadas con la temática de este trabajo. Los documentos corresponden a las siguientes publicaciones:

- **Authoring Tool for Location-Aware Experiences** ([Alconada et al., 2015a]).
- **Combing Location-Aware Applications with in-situ Actors Performances** ([Alconada et al., 2015b]).

Notar que las páginas siguientes están fuera de la numeración de este trabajo por ser los documentos originales de las publicaciones, los cuales fueron anexados sin modificaciones.

Authoring Tool for Location-Aware Experiences

Federico M. Alconada Verzini
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
falconada@lifia.info.unlp.edu.ar

Juan I. Tonelli
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
jtonelli@lifia.info.unlp.edu.ar

Cecilia Challiol[‡]
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
ceciliac@lifia.info.unlp.edu.ar

Alejandra B. Lliteras
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
lliteras@lifia.info.unlp.edu.ar

Silvia E. Gordillo[§]
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
gordillo@lifia.info.unlp.edu.ar

ABSTRACT
In this paper, we present an approach to create location-aware experiences. We use the concept of separation of concerns to represent the content layer and the location layer. This separation allows reusing the layers independently one of each other. The focus of this paper is to provide an end-user tool to create location-aware applications in-situ. By using our tool, the location-aware experiences can be defined in both indoor and outdoor spaces. We present an example of how our tool is used and we describe some discussion points that have occurred to us while defining a tool with these characteristics.

Categories and Subject Descriptors
D.2.2 [Software Engineering]: Design Tools and Techniques – *Object-oriented design methods, User interfaces*.
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces - *User-centered design*

General Terms
Design

Keywords
In-situ Authoring Tool; Location-Aware Experiences; Indoor-Outdoor Space; Separation of Concern; Mobile Applications

1. INTRODUCTION
The increasing capabilities of mobile devices (e.g. GPS, NFC, etc.) are leading to a new way of creating mobile applications in which end-users are getting involved. Nowadays, users not only create contents (e.g. taking pictures, creating comments, etc.) but also mobile applications; in particular Location-Aware applications [13].

Location-Aware applications can be used for different purposes and no matter which domain they have, the user of the application receives information based on his physical location. These domains can vary, going from education to games (more examples are mentioned in [7]).

Up until the last few years this kind of applications has been being created by developers. But this has changed with the growing use use of mobile devices and users have become part of these creations.

According to [12], end-users have two different ways to design location-aware applications: digital or physical. In the first case (digital design), a map is used as a reference to define the relevant locations and contents but users are not physically located there. For example, [10] and [12] approaches provide authoring tools that use maps to define relevant locations (and its associated content). In the second case (physical design), the user is physically located in the place where the relevant locations and contents will be defined. In the literature, this last way of designing is generally known as "in-situ creation" ([4] and [9]). As mentioned in [11] and [13], being in-situ enables users to identify relevant environmental aspects of the real world. This can be important not only to choose a more suitable location but also to define a more accurate content associated with each place.

In general, in-situ tools create a content coupled with a location and are only focused on outdoor applications. For example, CASTOR [9] associates a GPS location with each content. Another similar example is TOTEM [4], which allows choosing a GPS or an NFC location. Both approaches provide support to create only outdoor mobile applications.

In this paper, we present an authoring tool to create location-aware applications not only for outdoor spaces but also for indoor ones. This tool has been defined in a generic way so it can be used for any domain (education, entertainment, etc.). In addition, we have focused on providing a solution to decouple contents

[‡] Another affiliation for Cecilia Challiol: CONICET, Argentina.
[§] Another affiliation for Silvia E. Gordillo: CIC, Buenos Aires, Argentina.

NHT'15, September 1, 2015, Guzelyurt, Northern Cyprus.

DOI: http://dx.doi.org/10.1145/2804565.2804570

from locations by using the concept of separation of concerns which we have been analysing and modelling for mobile applications in our previous works ([1] and [2]). Furthermore, we present an example of how our tool is used and we describe some issues that have sprung to our minds while defining a tool with these characteristics.

This paper is organised as follows. In Sections 2 we describe some related works. Our approach is presented in Section 3. Our authoring tool is shown in Section 4. Conclusions and future work are mentioned in Section 5.

2. RELATED WORK

In [3], methods and tools for producing location-based Mobile Urban Dramas are presented. In this kind of applications, users become the main characters in a play. The authors define a framework to create these dramas. In particular, the authors present six dramas with different purposes, for example, tourism and learning. Note that these dramas are created by developers.

Structuring location-aware narratives is described in [7]. The authors present three patterns characterised using [5] and, based on this, they provide a sculptural location-based hypertext model. This model defines the concept of location-query as an attribute of different classes. Until now, this approach is focused on modelling aspects of these kinds of applications and not to provide an end-user tool.

As it was previously mentioned, end-users can design location-aware applications in two ways, digital or physical [12]. The former uses a map as a reference to define the relevant locations and contents but users are not physically located there. The latter, requires the user to be physically located in the place in which the application will be used later. Concrete examples of both approaches are described next.

In [12], a digital design tool called QuesTInSitu is presented. This tool enables users to define learning activities in specific outdoor locations. The activities defined can be run (by a student) using the mobile web application's user interface. The tool provides a map that uses a web map service (e.g. Google Maps), which is used by teachers, to create in-situ activities in relevant geolocations. The concept of in-situ activities means that, when the students reach one of these particular geolocations (using their mobile device's GPS), they receive the corresponding activity that is automatically triggered by the tool.

Another digital design is presented in [10]. The authors describe the IVO platform, which allows end-users to build and deploy mobile context-aware applications. To support this kind of applications, IVO platform uses an event-driven model. As part of the IVO platform, the authors also provide the IVO Builder and the IVO Client (Android application). In [10] is shown a tourist guide application created by using this platform.

Regarding to the physical design, two approaches are presented. In [9] the author describes CASTOR, (Context Aware STORytelling) an authoring system oriented to children that enables them to create structured stories in-situ. This system supports structuring the content both with a sequential logic or using the concept of branching. This system uses the mobile device's GPS to retrieve a specific geolocation that is associated to different contents (text, pictures, etc.). Moreover, in each location students specify contextual conditions used to trigger the content when the final mobile application is running.

The other similar in-situ authoring tool is presented in [4]. This is an Android application called TOTEM.Scout that allows users to create location-based content (image, video, etc.). The authors provide the concept of Shapes that are templates users can use to define contents (associated to relevant locations). These templates provide, for example, the possibility to associate a GPS or NFC location with the content. In addition, the authors provide the TOTEM.Design (a web-based tool) that uses the created content to build location-aware experiences. Up to now, the authors only support Google Maps and they mentioned as the future work to add floor plants to their tool.

In conclusion, all these mentioned approaches are focused on the production of different kinds of outdoor mobile applications. Additionally, the contents defined on these approaches are coupled with a specific location (generally a GPS location). Taking into account the contribution of this paper, we present a tool to define indoor-outdoor location-aware experiences, which models contents and locations in a decoupled way (providing reusability of both contents and locations).

3. OUR APPROACH

In Figure 1, a general schema of our approach is shown. We provide a responsive web-based interface used to define location-aware experiences. Our framework is used to provide the functionality required to build an experience (Figure 1 shows that the framework returns a final mobile experience). Web Services are used to communicate both parts (interface and framework). Our interface is capable of communicating both with map services (e.g. for outdoor representations) and sensing mechanisms such as GPS.

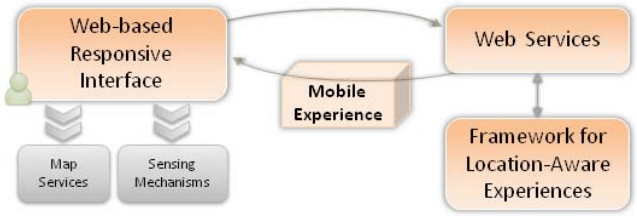

Figure 1. General schema of our approach.

Our approach defines general concepts used by our end-user tool and framework. We have identified two different layers (or concerns), one with contents (text, image, video, multiple choice question, etc.) and its relationships, and the other one that defines relevant locations (related with an indoor or outdoor space). Locations can be defined in different ways, such as in a symbolic or geometric way, as mentioned in [6]. The relationship between contents and locations define a point of interest (PoI). These are shown in the Figure 2.

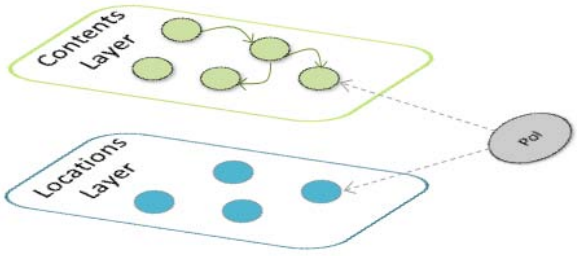

Figure 2. General concepts used in our approach.

Using the concepts defined in Figure 2, contents and locations can be part of different PoI, allowing reuse. This can be interesting and useful for different situations. For instance, if a museum used QR-codes associated to each object with our approach, the location layer could be reused to provide different location-aware experiences. Each experience could define different contents (related with their locations) with different purposes, such as learning activities or tour guides.

4. AUTHORING TOOL

In this section, we present our visual tool to create location-aware experiences. This tool is defined by using a responsive web-based interface so it can be used in any web browser (mobile or desktop). Up to now, our tool provides two different ways of defining locations, using GPS or QR-code. But its flexible design supports the addition of any other location sensing mechanism. Our tool enables users to define:

- Contents Layers
- Locations Layers (relative to an indoor or outdoor space)
- PoI (a content within a location forms a point of interest)

In this paper, we only focus on the creation of PoI. To obtain the location, a user can configure the location sensing mechanism. By default, GPS is chosen but he can switch into QR-codes whenever he wants. In the case of GPS, the behaviour is similar to the related work presented in Section 2. If QR-codes are used, a new code is generated for each content (in a future approach of the tool, we will consider the possibility of reading existing QR-codes). Each code is related to an indoor-outdoor location inside the tool's map. In the final application, QR-codes are used to obtain the contents associated to them. Either using GPS or QR-codes we create a separation layer with these locations.

In the following sections, we show an example of the usage of our tool and some discussion points.

4.1 Using our tool

A first version of our tool is presented in this section. The tool supports defining PoI in an indoor, outdoor or a mixed space (indoor-outdoor). The most interesting example is the latter, as it uses both indoor and outdoor spatial representations. In this first version, we have decided to use an image of the indoor space (the museum's plan) to represent the indoor space. Figure 3 shows how the image is overlaid over the outdoor map.

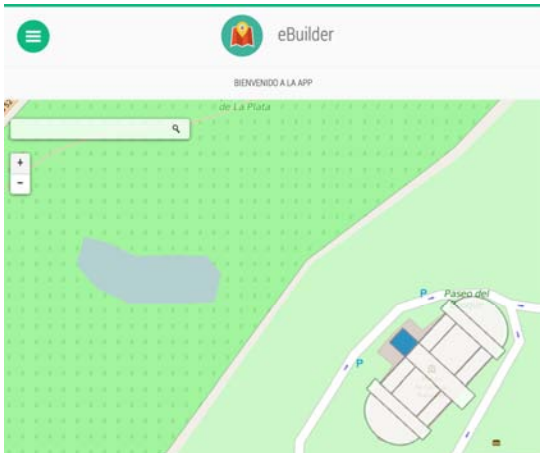

Figure 3. Indoor-Outdoor Map using our tool.

In Figure 4, the different types of content provided by our tool are shown, each one with one icon.

Figure 4. Icons representing contents in our tool.

Suppose a user is interested in creating a guided tour showing different mammal animals. In La Plata city (Argentina), the Zoo (outdoor) and the Natural Science Museum (indoor) are very close. The user wants to show some extinct animals (exhibited in the Museum) and some non-extinct animals (which are in the Zoo).

Using our tool the user can visit the Zoo and the Museum, and in each place define PoI. At the Zoo, the user takes advantage of the GPS sensing mechanism to create the different PoI. As he walks through the different places, he creates the different PoI and the markers representing each one of them are automatically placed on his/her current position.

Figure 5 shows some PoI created in the outdoor space (the zoo) and a linear relationship between them. For simplicity, assume that for each PoI only a text is defined.

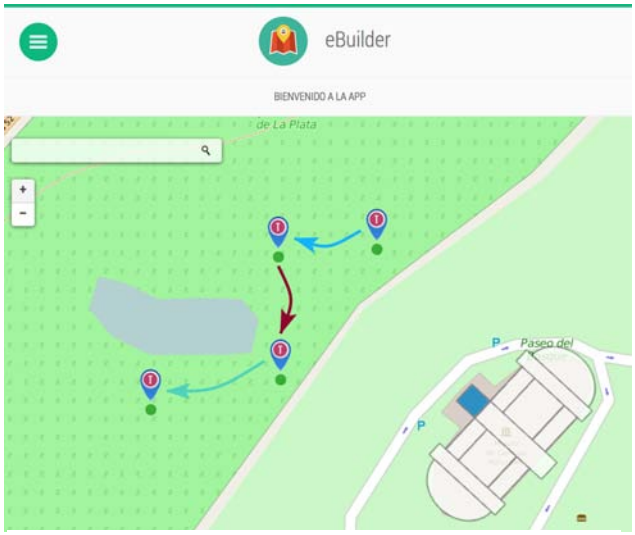

Figure 5. PoI defined in the outdoor map.

When the user is inside the Museum, he decides to switch to the QR-code sensing mechanism. As in this case there is no GPS coordinate representing the user's current position, the tool randomly places the PoI anywhere in the map. And this is why the user must manually drag the markers to the correct places on the tool's map. Every time the user creates a new PoI, a QR-code is generated as shown in Figure 6.

Figure 6. QR-code associated to a PoI.

Figure 7 shows all the PoI created by the user. Note that one of the outdoor PoI is connected with one of the indoor PoI.

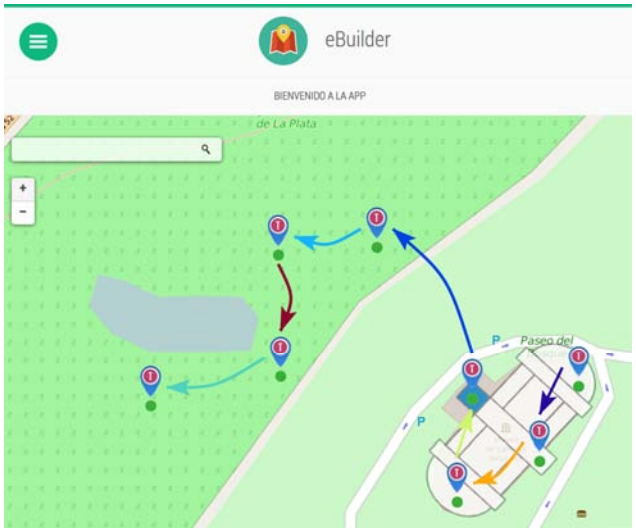

Figure 7. An indoor-outdoor tour defined with our tool.

As part of the final application, all QR-codes (if there are) are given to the user, who will be in charge of placing those codes in the correct physical locations. The framework builds the experiences as web applications with the possibility of using Phonegap [8] to create hybrid applications.

4.2 Discussions

Nowadays, most existing authoring tools do not provide any location sensing mechanism for indoor spaces. Consequently, not having this location assistance is the reason why creating mobile applications by end-users is more complex. The use of QR-codes is a first approach to define relevant locations. But this also depends on settling that code in the correct place.

The indoor representation is a topic by itself. In our first version of the tool, we only consider a referenced image to simplify it. But we know that this is not scalable in order to build an indoor space with more than one floor. In these cases, the tool will should be defined with a more complex representation as shown in [14].

The complexity of the tool increases when context features are defined as part of each PoI, as presented in [9]. Up to now, the first version of our tool only allows defining simple contents such as: text, figure, audio, video, multiple-choice questions or a composite of them.

Another interesting point to mention is the layer's reusability. In this case, location layer's reusability is possible because they just define relevant locations in an indoor-outdoor space. But the reuse of contents is not always possible if it is location-dependent (as is presented in [13]). In other words, if the content mentions any characteristic of the environment, the reuse is limited.

It is important to note that, with our tool, it is possible to define the three structure patterns mentioned in [7]. These structures are dynamically defined according to the relationships that users define between the contents.

5. CONCLUSION AND FUTURE WORK

We have presented an approach to create location-aware experiences. As mentioned before, we have used the concept of separation of concerns to represent the content layer and the location layer. To relate content and location, we have used the concept of PoI. We have presented a first version of our tool, which creates location-aware applications in-situ. Our tool uses indoor-outdoor spaces to define PoI and relationships between them. These are shown in the example presented in this paper. Some discussion points have been presented to show that this is the starting point of an area that requires much more research and evaluation.

Up to the moment, the tool has only been evaluated by the developers involved in its definition. We know that it is crucial to evaluate it with end-users. To achieve this, we are currently organising an experiment to create a specific location-aware application that involves indoor-outdoor places to be evaluated by end-users.

We are also working on providing more complex indoor representations as it is defined in [14]. According to that, we have evaluated different kinds of interfaces to ease the definition of PoI when the indoor space has more than one floor.

As future work, the tool will support more sensing mechanisms such as NFC. Furthermore, the tool will define more complex contents. For example, users will be able to record videos or take pictures (as is described in [4]).

6. REFERENCES

[1] Fortier, A., Challiol, C., Fernández, J. L., Robles, S., Rossi, G., and Gordillo, S. 2014. Exploiting personal web servers for mobile context-aware applications. *The Knowledge Engineering Review* 29, 2, (Mar. 2014), 134-153. DOI=http://dx.doi.org/10.1017/S0269888914000022

[2] Fortier, A., Rossi, G., Gordillo, S. E., and Challiol, C. 2010. Dealing with variability in context-aware mobile software. *Journal of Systems and Software*83, 6, (June 2010) 915-936. DOI= http://www.sciencedirect.com/science/article/pii/S0164121209002830

[3] Hansen, F. A., Kortbek, K. J., and Grønbæk, K. 2012. Mobile urban drama: interactive storytelling in real world environments. *New Review of Hypermedia and Multimedia*18, 1-2, (Mar.-June 2012), 63-89. DOI= http://www.tandfonline.com/doi/abs/10.1080/13614568.2012.617842

[4] Jurgelionis, A., Wetzel, R., Blum, L., and Oppermann, L. 2013.TOTEM.Scout: A mobile tool for in-situ creation of location-based content. In *Proceedings of Games Innovation Conference 2013* (Vancouver, BC, September 23-25, 2013). IGIC 2013. IEEE*,* 89-96.

[5] Kjeldskov, J., and Paay, J. 2007. Augmenting the City with Fiction: Fictional Requirements for Mobile Guides. In *Proceedings of Mobile Interaction with the Real World 2007/5th Workshop on HCI in Mobile Guides* (Singapore, September 9, 2007). MIRW 2007. ACM, New York, NY, 1-6.

[6] Leonhardt, U. 1998. *Supporting location-awareness in open distributed systems*. Doctoral dissertation, Imperial College.

[7] Millard, D. E., Hargood, C., Jewell, M. O., and Weal, M. J. 2013. Canyons, deltas and plains: towards a unified sculptural model of location-based hypertext. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media* (Paris, France, May 01-03, 2013). Hypertext 2013. ACM, New York, NY, 109-118. DOI= http://dl.acm.org/citation.cfm?doid=2481492.2481504

[8] Phonegap Home Page: http://phonegap.com/

[9] Pittarello, F., and Bertani, L. 2012. CASTOR: learning to create context-sensitive and emotionally engaging narrations in-situ. In *Proceedings of the 11th International Conference on Interaction Design and Children* (Bremen, Germany, June 12-15, 2012). IDC 2012. ACM, New York, NY, 1-10. DOI= http://dl.acm.org/citation.cfm?doid=2307096.2307098

[10] Realinho, V., Romão, T., Birra, F., & Dias, A. E. 2011.Building mobile context-aware applications for leisure and entertainment. In *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology* (Lisbon, Portugal, 8-11 November, 2011). ACE 2006. ACM, New York, NY, Article No. 29. DOI=http://dl.acm.org/citation.cfm?doid=2071423.2071459

[11] Rogers, Y., Connelly, K., Tedesco, L., Hazlewood, W., Kurtz, A., Hall, R. E., and Toscos, T. 2007. Why it's worth the hassle: The value of in-situ studies when designing ubicomp. In *Proceeding of the 9th international conference on Ubiquitous Computing (Innsbruck,* Austria, September 16-19, 2007). UbiComp 2007. Springer Berlin Heidelberg, 336-353.

[12] Santos, P., Hernández-Leo, D., and Blat, J. 2014. To be or not to be in situ outdoors, and other implications for design and implementation, in geolocated mobile learning. *Pervasive and Mobile Computing* 14, (Oct. 2014), 17-30.

[13] Weal, M. J., Hornecker, E., Cruickshank, D. G., Michaelides, D. T., Millard, D. E., Halloran, J., and Fitzpatrick, G. 2006. Requirements for in-situ authoring of location based experiences. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services* (Helsinki, Finland, September 12-15, 2006). Mobile HCI 2006. ACM, New York, NY, 121-128. DOI= http://dl.acm.org/citation.cfm?doid=1152215.1152241

[14] Zlatanova, S., Liu, L., and Sithole, G. 2013. A conceptual framework of space subdivision for indoor navigation. In *Proceedings of the Fifth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness* (Orlando, Florida, USA, November 5-8, 2013). ISA 2013. ACM, New York, NY, 37-41. DOI= http://dl.acm.org/citation.cfm?doid=2533810.2533819

Combing Location-Aware Applications with in-situ Actors Performances

Federico M. Alconada Verzini
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
falconada@lifia.info.unlp.edu.ar

Juan I. Tonelli
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
jtonelli@lifia.info.unlp.edu.ar

Cecilia Challiol‡
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
ceciliac@lifia.info.unlp.edu.ar

Alejandra B. Lliteras
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
lliteras@lifia.info.unlp.edu.ar

Silvia E. Gordillo§
LIFIA, Fac. de Informática, UNLP.
Calle 50 y 120, La Plata,
Buenos Aires, Argentina.
+54 221 422-8252
gordillo@lifia.info.unlp.edu.ar

ABSTRACT
In this paper we present a model to define location-aware experiences having its different aspects (contents, location, etc) decoupled one from another. We have created an application following this model, which is also combined with actors' performances. This application defines three relevant locations inside a specific building. By reading a QR-code the user indicates that he/she has arrived at one of those locations. In that moment, the application triggers some questions related to the actors' performances. Hence, the user is required to see all the plays in order to be able to answer such questions. The application assists the user in moving throughout the different locations. We present the evaluation of this application and then we discuss interesting aspects that have shown up as a result of the in-situ evaluation of this application.

Categories and Subject Descriptors
D.2.13 [**Software Engineering**]: Reusable Software - *Reuse models*
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces - *Prototyping*

General Terms: Design, Experimentation

Keywords: Location-Aware Applications, In-situ Actors Performance, In-situ evaluation, Prototype, Modelling Location-Aware Applications, Separation of Concerns, Mobile Computing

1. INTRODUCTION
Mobile Applications have been increasing in the last few years covering different domains such as education, entertainment, tourist guides, etc. In particular, those applications that reference some relevant aspects of the environment (e.g. location-aware application) have only been tested in an in-situ fashion. Different authors ([2], [3], [14] and [16]) have come to the conclusion that evaluating prototypes in-situ is essential. In [12], the authors reinforce this idea by explaining that the way in which a person perceives a physical space can change according to the time of the day (morning, afternoon, evening), for instance.

There are different ways of creating mobile applications. In particular, location-aware applications can be defined by developers (e.g. [8], [10]) or by end-user ([13] and [15]). In both cases, according to what was mentioned before, the final mobile application is required to be tested in-situ to obtain real evaluation results.

Mobile Urban Dramas are a particular type of mobile application introduced in [7] and defined as Location-Aware Narratives. In each location defined within a narrative, the user receives a multimedia content, such as an audio or an image, and then he/she continues to the next location. Recent works of these authors present six different dramas [8], which are defined with specific domains (e.g. education or entertainment). Inspired by these dramas we came up with the idea of creating a location-aware application combined with plays performed by real actors. For each relevant location, the application shows a specific content and several actors perform a play. The same play can be performed in different locations (the content received by the user remains the same but location and physical space change). For a rapid prototype of this kind of applications, it is important to represent the contents independently of each location.

We have been working on different modelling aspects of mobile applications ([5] and [6]). In particular, we have used the concept of separation of concerns to handle them in an independent way. For instance, having content and location concerns decoupled one another. This experiences that combine location-aware applications with actors' performances are faster to prototype if we use the concept of separation of concerns.

In this paper, we present a model to define location-aware experiences where content aspects are decoupled from location

NHT'15, September 1, 2015, Guzelyurt, Northern Cyprus.
 ISBN 978-1-4503-3797-7/15/09…$15.00DOI: http://dx.doi.org/10.1145/2804565.2804571

ones. To test our model, we create a location-aware application prototype. We describe an in-situ evaluation of this application, which combines location-aware content with actors' performances. We present various results and discussions points of this evaluation.

This paper is organised as follows. In Section 2 we describe some related work. A modelling approach for location-aware experiences is presented in Section 3. In Section 4, we describe a prototype which was created by using our model. An evaluation of this prototype is presented in Section 5. Conclusions and future work are mentioned in Section 6.

2. RELATED WORK

In [4], the authors analyse different kinds of Mobile Guides and give the taxonomy of relevant contexts in these kinds of applications, such as the location, that is one of the most useful contexts. The authors describe different location techniques and environment types. In [3], the authors mention the importance of evaluating prototypes in an in-situ way.

An event-driven model is used in [13] to create mobile context aware-applications. The authors build a mobile tourist guide using IVO platform (retrieving locations using GPS), which is tested in-situ.

In [10], the authors present a sculptural location-based hypertext model. This model is defined through a characterisation of three patterns (Canyons, Deltas and Plains) that authors have detected in this kind of applications. This model defines the concept of location-query as an attribute of different classes. To test their model, the authors have created GeoYarn, a web service application for Android implemented with this model which provides stories in different locations. Motivated by this idea, we have tested our model in the same way.

Six Mobile Urban Dramas are described in [8]. The authors of this paper present a conceptual framework that includes two specific models (user and environment) to define dramas whose plot is described by means of graphs with branches. The framework detects the user's locations through 2D barcodes, RFID or GPS and, when the user reaches a specific location, he/she receives a multimedia content (audios, images, animations or videos). Each drama is deployed as a mobile application both in iOS and Android platforms.

The authors in [8], [10] and [13] use different models to represent the content and the relevant locations. But in all of them, these concepts are coupled. We use these approaches as an inspiration and as a starting point to define a new model, where the content and location aspects are decoupled.

3. MODELLING LOCATION-AWARE EXPERIENCES

In Figure 1, we present a general schema of our model that allows representing location-aware experiences. The abstract class *Content* represents a general content. For simplicity we have decided not to show each concrete subclass (as Text, Image, Video, etc). We have decoupled the content from the structure that contains it. To do that, we have created the concept of Element. This enables us to reuse a content more than once in different structures (or patterns as mentioned in [10]). The location is defined as an *Interface* (following the idea of [9] in which a location can be represented, for example, in a symbolic or geometric way). The class PoI acts as a weaving between elements and locations. Note that the same content or position could be used by different PoI.

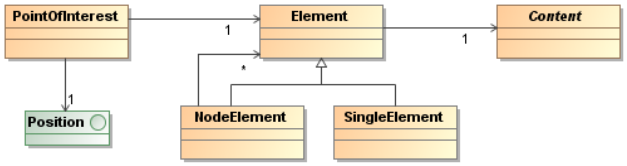

Figure 1. Decoupled contents, structures and locations.

Other relevant classes are required to be able to define the experience itself. To achieve that, we have created the class *MobileExperience* that contains all the relevant PoI. To show how we achieve the decoupling of some relevant aspects, we point out here the classes *SensingManager* and *SpaceRepresentationManager*. These classes are shown in Figure 2. This is a simplified model which aims at presenting the general idea.

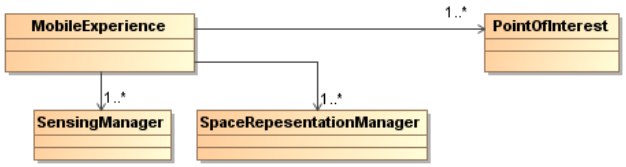

Figure 2. Simplified model to define location-aware experiences.

4. OUR PROTOTYPE

In 2014, we were invited by one of the local TED talks (TEDxDiagonales73) to create a mobile application to be used among the participants. Using the model depicted in Section 3, we decided to create a location-aware application prototype called CaminosAlternativos (Alternative Paths). We defined the indoor space by drawing a plan of the building, including some relevant points such as the elevator and the stairs.

When we first thought about this prototype, we agreed to make a web application and hosted it in a server, so that the participants could access it through the Internet. Later, when we tested the WiFi connection of the building where the TED talk was going to be held, we found out that it was not working properly. Therefore, we decided that the fewer HTTP connections the prototype required, the better performance it would be able to achieve. This is why we decided to use the framework Phonegap [11] to create a hybrid mobile web application for Android. In this way, users would install the mobile application in their phones and the prototype would only have one HTTP connection to send the result generated by each user.

In order to create a more interesting application we thought about combing this application with acting performances in each location. To do so, we contacted Lorena Velazquez, headmaster of the ECAE School of Performing Arts, who helped us to define each acting performance and coordinate the actors' participation. Lorena proposed us doing an experience in which three performances would be defined, each one based on a popular TED talk phrase taken from previous TED events (these phrases were selected by the organisers of the TEDxDiagonales73 talk). A

user would see the plays and answer, in one word, what has he/she felt while seeing it. Once the user finishes seeing the three plays, he/she would have to create a phrase trying to use the three words he/she has answered. The user with the most creative phrase would be the winner of a prize at the end of the TEDxDiagonales73 talk.

To achieve this, we visited the building (in which the talk was going to be held) and defined three relevant locations (corresponding to three rooms or scenes), two on the first floor and the other one on the second floor. The way to get around the three locations was without a predetermined order. The first QR-code selected by the user determined the first play he/she would see, which is carried by an actor as shown in Figure 3 (so, there are three actors, one for each scene). Even though this first-scene selection could have been achieved by a technical solution, we decided to do it in this way in order to make a more exciting experience.

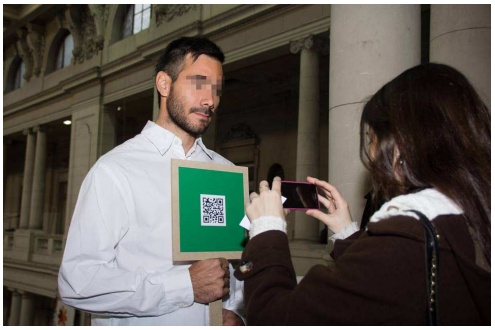

Figure 3. Deciding the first place.

After reading the first QR-code (Figure 3), the application shows a map to the user indicating the path to reach the specific room where one of the performances will take place. The user walks to that room and finds another QR-code, as shown in Figure 4. When the user reads this code a question appears and the user is intended to answer a word regarding to his/her feelings about the performance.

Figure 4. QR-code of a scene.

Once the play is over (all of them last 5 minutes) and the user has answered the question, the application shows the map of the building with the other two scenes that the user still has to see. If the two remaining plays are the ones on the first floor, a map only of this floor is displayed to the user. Otherwise, the user has to choose which floor he/she wishes to go next (see Figure 5).

Figure 5. Two options to follow with the experience.

After choosing one of these options (Figure 5), the user receives a map with the path to its destination. If the path involves moving to the other floor, he/she receives the relevant instructions through the maps shown in Figure 6. The map on the left indicates the path to the stairs. Finally, the map on the right assists the user to reach the correct room.

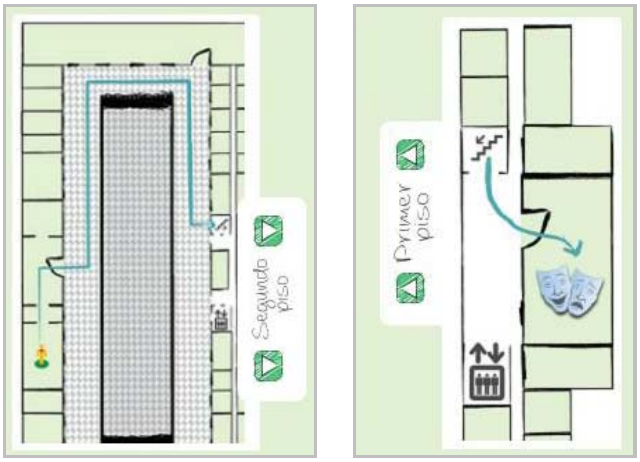

Figure 6. Instruction to move from one floor to the other.

The same process which took place in the first scene is repeated. Since the application was defined with only three scenes, the path to the last scene is automatically provided when the second question is answered. When the user completes the last question, he/she receives a final question to generate a phrase about how the performances have made him/her feel.

Note that the pictures shown in Figure 3 and 4 are taken during the in-situ evaluation with TEDxDiagonales73 participants.

5. EVALUATION

In this section we present the in-situ evaluation of the prototype described in Section 4. The participants of TEDxDiagonales73 that used the application were 34 (ages ranged from 20 to 47). All participants used their own mobile devices to install the prototype.

5.1 Analysis and Design

We created a document to describe general characteristics of the prototype and what the installation process in their mobile devices was like. This information was available one day before the TEDxDiagonales73 talk. In this way, the user had the chance to download it and have it installed by the time of the talk.

We made identification cards with a QR-code and gave them to the users when the experience started. To start running the prototype users were required to read the code of their cards so as to be able to receive the map to go to the place where the three actors were. This identification step was made both to prevent users from running the prototype before the time the evaluation began as well as to create more expectations. Figure 7 shows users reading their identification codes.

Figure 7. QR-code for identification.

We defined a SUS questionnaire [1] to evaluate the usability of the system. In addition, we defined a general form with two questions:

- Mark the grade of complexity on the use of the application (scores 1 to 5, 1 means low difficulty and 5 high difficulty)
- The assistance provided by the application was enough to move from one place to another (scores 1 to 5, 1 means strongly disagree and 5 strongly agree)

These forms were completed by the participants at the end of the experience.

5.2 Result and Discussions

The 34 participants completed both forms mentioned in the previous section. The result of the SUS was 76,0 (over 100), which is an acceptable score for a prototype and considering that the indoor space was represented only with and static image that limited the usability of the user.

The result of the general form is described below. The average score for the question "*Mark the grade of complexity on the use of the application*" was 2,14 (SD=1,23). This means that the complexity to use the prototype was low. The other question "*The assistance provided by the application was enough to move from place to another*" had an average score of 3,79 (SD=1,27). According with these values the assistance is more open to discussion because it was useful to most of the users but for others it was not. This point requires further improvement in future prototypes.

In this section we discuss some interesting points. To prevent all the users from going to the same play, each actor (of the three initial actors shown in Figure 3) counted the number of participants that were reading his QR-code. When this number reached the number of 12 users, the actor hid the code so that no other user could read it. This tactic was thought to balance the number of users in each room (or scene) preventing each room from getting overcrowded.

Another issue came up at the time of choosing the second play to see. Participants that were on the first floor (where two performances were being held), preferred their next choice to be the one of the first floor. In this situation, there were fewer people on the second floor than on the first one, where most participants were.

All three performances required a clockwork synchronisation to start simultaneously. To solve this, in each scene there were assistants who saw to this and decided when to start each performance, so as to do it all at the same time. The assistants' mission was also guaranteeing that there were no participants in the corridor or stairs.

The QR-code identifications were general, so there was no need to give any personal information. This was an advantage due to the fact that being anonymous increased the chances that users would answer the questions.

Both the content and the structure can be reused as the application was built following the model presented in the Section 3. To reuse them, it will only be necessary to define a new physical space to be able to test this prototype.

6. CONCLUSION AND FUTURE WORK

In this paper, we have presented a model to location-aware experiences. We described the main classes and focused on providing decoupled aspects of these kinds of applications, such as contents, structure and locations.

We have described a novel prototype, which was combined with actors' performances. We have presented and analysed the result of the evaluation in-situ. The discussion points mentioned in this paper are part of the learning process in evaluation in-situ. This type of evaluations (with real end-user people) are the best way to understand how these kind of applications behave.

We have presented the evaluation of this application and then we discussed some interesting aspects that were learnt as part of the in-situ evaluation.

We are working to enrich our model and create more complex applications. One of the aspects to improve is the map the application provides. In indoor spaces it is essential to be precise in the assistance to ensure that the user arrives at his/her destination as easily and straightforward as possible. Moreover, based on this model, we are currently developing a tool that enables end-users to create these location-aware experiences.

As a future work, we will evaluate this model with other prototypes to consider other relevant aspects. For example, the collaboration between users to create contents.

7. ACKNOWLEDGMENTS

We extend our thanks to Lorena Velazquez for creating and preparing each acting performance, and to the actors of ECAE (*Espacio Creativo de las Artes Escénicas*). The authors thank to Agustina Zimbello and Leandro Vilas for their collaboration on the organisation of this experience.

8. REFERENCES

[1] Brooke, J. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189,194, 4-7.

[2] Consolvo, S., Harrison, B., Smith, I., Chen, M. Y., Everitt, K., Froehlich, J., and Landay, J. A. 2007. Conducting in situ evaluations for and with ubiquitous computing technologies. *International Journal of Human-Computer Interaction* 22, 1-2, (Dec. 2007), 103-118.

[3] Crabtree, A., Chamberlain, A., Davies, M., Glover, K., Reeves, S., Rodden, T., and Jones, M. 2013. Doing innovation in the wild. In *Proceedings of the Biannual Conference of the Italian Chapter of SIGCHI* (Trento, Italy, September16-19, 2013). CHItaly '13. ACM, New York, NY, Article No. 25. DOI= http://dl.acm.org/citation.cfm?doid=2499149.2499150

[4] Emmanouilidis, C., Koutsiamanis, R. A., and Tasidou, A. 2013. Mobile guides: Taxonomy of architectures, context awareness, technologies and applications. *Journal of Network and Computer Applications* 36, 1, (Jan. 2013), 103-125.

[5] Fortier, A., Challiol, C., Fernández, J. L., Robles, S., Rossi, G., and Gordillo, S. 2014. Exploiting personal web servers for mobile context-aware applications. *The Knowledge Engineering Review* 29, 2, (Mar. 2014), 134-153. DOI=http://dx.doi.org/10.1017/S0269888914000022

[6] Fortier, A., Rossi, G., Gordillo, S. E., and Challiol, C. 2010. Dealing with variability in context-aware mobile software. *Journal of Systems and Software* 83, 6, (June 2010) 915-936. DOI= http://www.sciencedirect.com/science/article/pii/S0164121209002830

[7] Hansen, F. A., Kortbek, K. J., and Grønbæk, K. 2008. Mobile Urban Drama–Setting the Stage with Location Based Technologies. In *Proceeding Proceedings of the 1st Joint International Conference on Interactive Digital Storytelling: Interactive Storytelling* (Erfurt, Germany, November 26-29, 2008). ICIDS '08. Springer-Verlag Berlin, Heidelberg, 20-31.

[8] Hansen, F. A., Kortbek, K. J., and Grønbæk, K. 2012. Mobile urban drama: interactive storytelling in real world environments. *New Review of Hypermedia and Multimedia* 18, 1-2, (Mar.-June 2012) 63-89. DOI= http://www.tandfonline.com/doi/abs/10.1080/13614568.2012.617842

[9] Leonhardt, U. 1998. *Supporting location-awareness in open distributed systems*. Doctoral dissertation, Imperial College.

[10] Millard, D. E., Hargood, C., Jewell, M. O., and Weal, M. J. 2013. Canyons, deltas and plains: towards a unified sculptural model of location-based hypertext. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media* (Paris, France, May 01-03, 2013). Hypertext 2013. ACM, New York, NY, 109-118. DOI= http://dl.acm.org/citation.cfm?doid=2481492.2481504

[11] Phonegap Home Page: http://phonegap.com/

[12] Pittarello, F. 2011. Designing a context-aware architecture for emotionally engaging mobile storytelling. In *Proceeding Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction* (Lisbon, Portugal, September 5-9, 2011). INTERACT '11. Springer-Verlag Berlin, Heidelberg, 144-151.

[13] Realinho, V., Romão, T., Birra, F., and Dias, A. E. 2011.Building mobile context-aware applications for leisure and entertainment. In *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology* (Lisbon, Portugal, 8-11 November, 2011). ACE 2006. ACM, New York, NY, Article No. 29. DOI=http://dl.acm.org/citation.cfm?doid=2071423.2071459

[14] Rogers, Y., Connelly, K., Tedesco, L., Hazlewood, W., Kurtz, A., Hall, R. E., and Toscos, T. 2007. Why it's worth the hassle: The value of in-situ studies when designing ubicomp. In *Proceeding of the 9th international conference on Ubiquitous Computing* (Innsbruck, Austria, September 16-19, 2007). UbiComp 2007. Springer Berlin Heidelberg, 336-353.

[15] Santos, P., Hernández-Leo, D., and Blat, J. 2014. To be or not to be in situ outdoors, and other implications for design and implementation, in geolocated mobile learning. *Pervasive and Mobile Computing* 14, (October, 2014), 17-30.

[16] Weal, M. J., Hornecker, E., Cruickshank, D. G., Michaelides, D. T., Millard, D. E., Halloran, J., and Fitzpatrick, G. 2006. Requirements for in-situ authoring of location based experiences. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services* (Helsinki, Finland, September 12-15, 2006). Mobile HCI 2006. ACM, New York, NY, 121-128. DOI= http://dl.acm.org/citation.cfm?doid=1152215.1152241

[‡] Another affiliation for Cecilia Challiol: CONICET, Argentina.

[§] Another affiliation for Silvia E. Gordillo: CIC, Buenos Aires, Argentina.